

# **HAIR: Hierarchical Architecture for Internet Routing**

Anja Feldmann	TU Berlin/Deutsche Telekom Labs
Randy Bush	Internet Initiative Japan
Luca Cittadini	Università Roma Tre
Olaf Maennel	TU Berlin/Deutsche Telekom Labs
Wolfgang Mühlbauer	TU Berlin/Deutsche Telekom Labs

Bericht-Nummer: 2008-14

ISSN-Nummer: 1436-9915

# HAIR: Hierarchical Architecture for Internet Routing

Anja Feldmann  
T-Labs/TU Berlin  
anja@net.t-labs.tu-berlin.de

Randy Bush  
IJJ  
randy@psg.com

Luca Cittadini  
Università Roma Tre  
luca.cittadini@gmail.com

Olaf Maennel  
T-Labs/TU Berlin  
olaf@maennel.net

Wolfgang Mühlbauer  
T-Labs/TU Berlin  
wolfgang@net.t-labs.tu-berlin.de

## ABSTRACT

Super-linear routing table growth, high update churn, lack of mobility and security, insufficient support for multi-homing and traffic engineering are some of the significant deficiencies of today's Internet. More and more researchers are convinced that these shortcomings cannot be resolved by incremental and band-aid solutions.

In this paper, we introduce HAIR, a scalable routing architecture for the future Internet. It addresses many of the problems the Internet is facing today. The focus is on limiting routing table size and update churn while supporting legacy hosts and avoiding unnecessary burden for transit providers. The key idea is to combine a hierarchical routing approach with locator/identifier separation: The routing as well as the mapping system are organized in a hierarchical manner where updates to both systems are not globally visible as far as possible.

First experiences with a prototype implementation are promising and demonstrate a potential migration path where legacy devices are supported as well.

## 1. INTRODUCTION

Scalability is the most direct problem that the routing system is facing today. Routing tables in the default-free zone (DFZ) of the Internet contain more than 300,000 prefixes and continue to grow super-linearly. Aligned with this is a steady increase in the rate of changes that have to be integrated into the routing and forwarding tables as well as in the time that is required until the routing converges. According to [12] peak rates in the magnitude of 1,000 prefix updates per second across the whole network are not an exception.

Unfortunately, scalability is not the only problem we are facing at the moment. Aspects or features that are essential today, such as mobility, have not been appropriately considered at the time the Internet was designed. Insufficient support for traffic engineering and multi-homing as well as provider lock-in are further examples frequently mentioned in this context. Given these shortcomings, a wide range of research projects [1, 13, 21] believes that those problems cannot be resolved by the conventional incremental and “backward-compatible” style of research. For a more detailed overview of the problem space and of the “clean-slate” re-

search paradigm we refer the reader to [10].

A plethora of proposals have been made to remedy the problems of the current routing architecture. Backward-compatible techniques, such as NAT, address the shortage of host IP addresses within (sub)networks, but they apparently do not eliminate the need for more globally-routable IP addresses. For this reason, many researchers argue that designing a routing system for the future Internet requires the more “radical” approach of a locator-identifier split (e.g., [9, 14, 19, 28]). The idea is to have different namespaces for *identifiers* and *locators*. Currently both functionalities are mangled within IP addresses that identify end points for example in the transport layer and that reflect as well the position of network devices within the network. Proposals made in this area either adopt a host-based (e.g., [19]) or network-based approach (e.g., [9]). While host-based approach can self-deploy without the need for cooperation by the network operators, network-based approaches are capable of transparently supporting legacy hosts.

In spite of the multitude of suggested solutions and of the discussion that has been ongoing for many years now, there is still no consensus on a new routing architecture. We believe that this is due to many reasons. First, a large fraction of the proposals only address a subset of the requirements for a new routing architecture. Take as an example *shim6* [19] that does not directly mitigate the problem of routing table growth. Second, the majority of previous approaches put the burden of deploying new architectural components in the core of the network. By contrast, we believe that such workload should be pushed as far as possible at the “edge” of the Internet, possibly having ISPs sharing the costs with access networks. This requirement conflicts, e.g., with LISP [9], which needs mapping devices at the ingress and egress points of transit networks. Third, existing proposals sometimes do not come up with feasible migration paths and neglect the support for legacy hosts. Last but not least, most existing solutions do not directly address the update churn, while in our opinion this is a primary issue: As far as possible, we should prevent updates to the routing *and* mapping system from being globally visible.

We, in this paper, introduce HAIR, a scalable routing architecture for the future Internet. The principal design requirements of HAIR are (i) scalability (in particular restrict-

ing routing table size and update churn), (ii) native support for multi-homing, traffic engineering and mobility, (iii) a feasible migration path supporting legacy devices and finally (iv) moving as much functionality as possible to the “edge” of the Internet (i.e. hosts and access networks), relieving the core from additional workload.

There are two key ideas that guide the design of our architecture. First, we adopt a *hierarchical* approach. Findings from theory [11] suggest that hierarchical routing is very effective in terms of scalability as long as the number of “separators” between different hierarchical entities can be kept minimal. Fortunately, it is widely agreed that the Internet is composed of a stable “core” formed by the transit providers and a more dynamic “edge”. Hence, we believe that separators are appropriate at the intersection between the following architectural entities: Transit networks, access networks and layer-2 networks at the bottom of the hierarchy. However, our approach supports even more separators, in order to also cope with the scalability issues within a single network.

Second, we decide to *decouple locators from identifiers* in order to allow for (end-host) mobility and to avoid issues such as provider lock-in. Before sending a packet, a host asks a mapping service for the corresponding locator(s) of a given identifier. Locators encode a loose source route from a host over its access network, the transit networks and finally over the destination access network to the destination host. Packets are forwarded solely based on locators by potentially making use of encapsulation. Note that, due to our hierarchical design, we allow for different routing protocols in different access networks and keep inter-network communication as minimal as possible. The same applies for the mapping system that is organized in a hierarchical manner as well. Each access network is responsible for mapping the identifiers of its attached hosts to locators. In addition, there exists a single central mapping service implemented via DHT that knows for all identifiers the access network in which the corresponding host is currently located. We show that the update churn to the mapping system can be kept low even in mobility and traffic engineering scenarios.

HAIR is more than the sum of its parts. It is the combination of a network-based approach – mappings services are provided by access networks – and of an architecture, keeping functionality at the “edge” of the Internet, which makes HAIR distinguishable from previous work. In addition to this, the hierarchical structure of the mapping and routing system allows us to effectively restrict update churn, something which has frequently been neglected by previous works. Last but not least, HAIR can easily be modified to interact with legacy hosts.

The rest of this paper is structured as follows. In Section 2 we give an overview of related work and discuss the design space for architecturing future routing systems. Then in Section 3 we present our architecture HAIR, before we describe a potential migration path and our prototype implementation in Section 4.2. Finally, we conclude in Section 5.

## 2. DESIGN SPACE

Recently, a plethora of research projects (e.g., [13, 21, 23]) have been initiated that adopt a clean-slate approach. They are united by the conviction that the shortcomings of today’s Internet cannot be resolved by conventional incremental style of academic and industrial networking research. Fundamental problems of the Internet routing architecture include amongst others the scalability of routing tables, high update rates, inadequate support for traffic engineering or multi-homing, address shortage, lack of mobility and provider lock-in. A nice characterization of the problem space and of the clean-slate paradigm is given in [10].

It is easy to be overwhelmed by the sheer number of proposals that have been made to remedy the problems of the current routing architecture. Hence, it is vital to carefully study the design space before nailing down a new routing architecture for the future Internet. In this Section we discuss design alternatives for such a routing architecture and illustrate, based on existing solutions, the advantages and disadvantages of a certain design choice. We try to be as comprehensive as possible, although it is certainly impossible to capture the complete range of related work made in this field.

Unfortunately, a large fraction of the suggested solutions only address a subset of the problems (see above) that the Internet is currently facing. Take as an example *Mobile IP* [20] which allows for end-host mobility but was not designed to reduce the size of routing tables in the core of the network. The *Host Identity Protocol* (HIP) [18] decouples two namespaces – namely the names of a host’s networking interface and the names of a location – which are currently mangled within an IP address and thus supports easy renumbering of the inter-networking layer. However, it does not specify any mechanism to reduce the size of routing tables. Unlike HIP, the *Hybrid Link-State Path-Vector* (HLP) [18] protocol has the focus on reducing update churn and routing table size in the core, but neglects for example mobility. Simple solutions such as exclusively assigning *Provider Aggregatable* (PA) addresses cause provider lock-in and may induce frequent renumberings of addresses.

For the remainder of this section, we confine ourselves to a class of proposals that are frequently referred to as *locator/identifier separation* or *Loc/ID split* (e.g., [9, 15, 19, 24]). They decouple the identifier from the locator functionality, currently both mangled within an IP address. While these solutions inherently support mobility<sup>1</sup>, they do not necessarily resolve the scalability issues in today’s Internet.

The first decision that needs to be made when designing a new routing architecture, is whether to use single namespaces or separate namespaces for identifiers and locators. The former option is chosen by more “radical” proposals such as ROFL [4] or VRR [3] that use flat labels as iden-

---

<sup>1</sup>When separating locators and identifiers, only the identifier will be used as connection identifier in a TCP session.

tifiers and locators. However, it is also chosen by the more “conservative” approaches of *Locator/Identifier Separation Protocol* (LISP) [9] and *shim6* [19]. While relying on a single namespace seems appealing for reasons of simplicity, it violates the guideline of modular design: Entities with different functionalities and different characteristics (contrary to locators the identifiers are stable) should be separate. Introducing separate namespaces for locators and identifiers gives flexibility for future adaptations. Such an approach is taken for example by the *Internet Indirection Infrastructure* [24] or ISLAY [15].

Irrespective of whether we have single or separate namespaces, both identifiers and locators can be organized in either a hierarchical or a flat namespace. Today’s Internet has two global namespaces (DNS and IP addresses), both of which are tied to a pre-existing structure (administrative domains and network topology) and can therefore be considered as hierarchical. It has turned out that this particular choice of hierarchies actually hampers mobility and renumbering of addresses. [3, 4, 18] solve these problems by relying on flat identifiers at the cost of giving up the possibility of aggregating identifiers. The same tradeoff applies for locators, although recent work tries to reach a compromise. For example, in [7] Eriksson et al. suggest to dynamically build hierarchical locators on-demand, such that they are always aligned with the network topology.

Regarding Loc/ID splits, they are frequently distinguished between *host-based* and *network-based* solutions. For host-based approaches (e.g., [19]), each host possesses one host identifier and one or even a set of locators. The transport layer sees only the identifier which needs to be mapped onto one of the locators. Contrary to network-based approaches the end hosts themselves are responsible for the mapping and the encapsulation or translation. For this reason, solutions such as [19] require end hosts to maintain state. If Loc/ID splits rely on a network-based approach (e.g., [9]), the network stack of hosts remains unchanged. In this case, end hosts generally use a stable, non-routable IP address as identifier. This is mapped onto global-routable addresses by a mapping service that the (edge) network provides. Both Loc/ID split approaches offer advantages and disadvantages. While network-based solutions address the shortage of global-routable IP addresses and routing table size, network-based solutions allow for end-host mobility but still require a global-routable locator for each end host. There are many pros and cons for both network-based or host-based Loc/ID split approaches. In our opinion, it is particularly important to take into account the potential incentives for a network provider or end host to deploy one of the solutions and to determine whether the mapping functionality should be placed at the end host, edge or core of the Internet.

A further design choice for future routing architectures is whether to separate the “core” of the Internet from the “edge” in terms of the scope of routing protocols (e.g., [7, 26]. Take as an example HLP [26]. It routes based on AS

numbers and adopts a link-state routing approach within a given hierarchy of ASs (as specified by provider-customer relationships), but uses a path-vector protocol between hierarchies. The idea is to reduce update rates and speed up convergence while preserving global reachability. The alternative choice is to use a global flat routing space as in today’s Internet or to deploy more “radical” solutions such as ROFL [4].

Irrespective of the adopted Loc/ID split approach, there is the need to provide a mapping service inside the network for network-based solutions. In terms of how the indirection between identifiers and locators is implemented, we can distinguish between solutions with (e.g., [9]) and without tunneling (e.g., [2, 28]). While the former category encapsulates packets and adds a header with the resolved locator, the latter either performs a NAT-style translation between identifier to locator addresses (e.g., [28]) or requires addresses that are composed of an identifier as well as a locator part [2]. The obvious drawback with tunnel-based approaches is the reduced MTU size, whereas NAT-style translation techniques may require state to be kept at special boxes.

There are further aspects which need to be considered when designing the mapping system. For example, the mapping function can be either “one-to-one” or “many-to-one” (see [27]). The solution of [28] maps edge addresses one-to-one onto the transit addresses from a given provider. To preserve scalability of the routing tables, transit addresses need to be organized such that they can be easily aggregated. The alternative approach is adopted by LISP [9] where multiple endpoint IDs (EIDs) can be mapped onto multiple routing locators (RLOCs). The assumption here is that there is a limited number of RLOCs that need to be globally routable. Also, the mapping service can be implemented in different ways. Some solutions (e.g., [18]) extend DNS and rely on DNS lookups to determine locators for a given identifiers, while others use overlays [8, 17, 24]. Overlays are frequently based on DHTs such as [22, 25]. There even exist proposals that are agnostic to the specific mapping solution. For example, LISP leaves the choice of whether to use LISP-ALT [8], which is based on an overlay of BGP routers, LISP-DHT [17], relying on DHT techniques, or any other solution that understands LISP queries.

The discussion of this section reveals the challenges we face when designing a new routing architecture for the future Internet. To finally meet the design requirements, we carefully need to weigh the pros and cons for a large number of design alternatives. Note that our discussion in this section is by far not complete, although we have tried to be as comprehensive as possible. Further design choices include, for example, whether to be backward-compatible or not or which type of routing algorithm to use for locators. Possible choices here are link-state versus distance vector approaches, shortest path routing versus compact routing etc.. Discussing all them is beyond the scope of this paper.

### 3. ARCHITECTURE

HAIR proposes a scalable routing architecture for the future Internet. In this section, we start with the design requirements for HAIR, followed by an overview of the architectural components in Section 3.2. Then in Section 3.3 we illustrate how packets are forwarded from a source to a destination host, before we explain in Section 3.4 the component that maps identifiers to locators in more details.

#### 3.1 Design Requirements

The list of challenges facing today's Internet is long: Scalability, security, mobility, multi-homing, traffic engineering, provider independence to name just a few examples. The guideline for a new routing architecture is to be comprehensive and to address as many challenges as possible. Still, we identify in the following four requirements that are essential for the design of HAIR.

**Scalability:** The workload of a typical router boils down to processing routing updates, computing routing tables and forwarding packets. Currently, a considerable fraction of BGP updates are globally visible and the number of routing table entries continues to grow. This adds high memory and computational burden on routers in the core of the Internet. We envisage a routing architecture that puts an end to routing table growth in the DFZ and where updates messages are localized and infrequent.

**Multi-homing, traffic engineering and mobility:** Many of the current problems were not an issue at the time the Internet was designed. The lack of mobility stems from the design decision to mangle the identifier and locator functionality within an IP addresses. Both multi-homing as well as (inbound) traffic engineering require network administrators to advertise more-specific prefixes to the rest of the Internet. Our architecture needs to inherently support mobility and provide means for multi-homing and traffic engineering without increasing routing table size in the core of the Internet.

**Migration:** To ease deployment, it is essential to devise a smooth migration path that similar to the approach in [9] allows ISPs to change a minimal number of devices and that supports legacy routers and hosts.

**No changes to the core:** One motivation for designing a new routing architecture is to relieve routers in the DFZ from unnecessary workload. Adding new functionality (e.g., mapping services) in the core of the Internet is counter-productive. Any new components or mechanisms which need to be introduced as part of HAIR should therefore be placed close to the edge of the Internet (i.e., in access networks).

#### 3.2 Components of HAIR

The general design of our architecture follows two major *guidelines*. On the one hand locators need to be *decoupled*

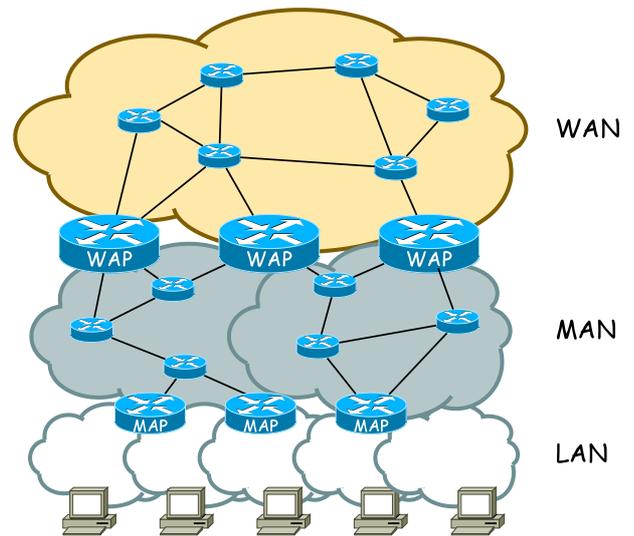


Figure 1: Network structure.

from identifiers, on the other hand we have to rely on *hierarchical routing*. Together, these two guidelines enable the design of a scalable routing system that meets the design requirements of the preceding section. In this section, we will elaborate on this, present the architectural components and provide a general overview of HAIR.

The justification for the first guideline is evident. Only if identifiers and locators are not mangled within one entity (i.e., IP addresses), we can enable mobility for end hosts or even complete (sub)networks and avoid problems such as provider lock-in. An obvious consequence of our design choice of separating these two functionalities is the need for a new architectural component that we call *mapping service*: Given a certain identifier it returns the current locator(s). Our major guidelines are intertwined in the respect that the implementation of the locator/identifier is done in a hierarchical manner. For more details on the implementation of this service we refer to Section 3.4.

The remaining architectural components are identified based on theoretical insights about hierarchical routing and on the actual structure of today's Internet. The expected benefit from adopting a hierarchical routing approach is better scalability achieved by information hiding: An entity only has aggregate not complete information about entities somewhere else in the hierarchy. While it is not yet clear whether a hierarchical scheme is the only one that can provide the required scalability, all currently known scalable addressing schemes exploit some kind of hierarchy [5]. The hierarchical approach is further motivated by theoretical results (e.g., [16]) which show that, by optimally placing *separators*, i.e., elements that connect levels in the hierarchy, tremendous gain can be achieved in terms of both routing table size and update message churn.

Our architecture HAIR proposes to place separators for a hierarchical routing scheme at the border routers of transit

ISPs. As shown in Figure 1 we define a layered, hierarchical network structure which decouples *transit* networks from *access* networks. Each layer is connected to the others via *attachment points*. In general we distinguish between the following types of networks:

**LAN:** Access network where hosts and servers are attached. For example, this can be a (switched) layer-2 network such as an Ethernet LAN.

**MAN:** A managed network that aggregates access networks and is administered by a single authority. Contrary to a LAN, a MAN contains only routers. In the current Internet, MANs may correspond to small ISPs that do not provide transit.

**WAN:** The backbone network that routes packets between MANs. In the actual Internet this part of the network may be formed by transit providers and tier-1 autonomous systems.

According to the definition of the three network layers we identify the following two separators:

**MAN Attachment Point (MAP):** Routers inside a MAN to which one or multiple LANs are attached.

**WAN Attachment Point (WAP):** Routers inside a WAN through which a MAN is connected to the backbone. Note that a single WAP can serve multiple MANs.

To achieve scalability, the hierarchical routing scheme is combined with the separation of locators and identifiers. Network nodes are bound to an identifier which is not used for routing packets and which is globally unique. In principle, any identifier namespace is possible, even self-certifying identifiers. A scalable mapping service, presented in Section 3.4, is responsible for translating identifiers to the appropriate locators, which are then used for the actual routing. Our goal of limiting routing table size is achieved by only exposing attachment points to the routing system. For this purpose, HAIR uses locators that encode a *loose source route* towards a certain host: Each locator can be seen as a sequence of attachment points that need to be traversed when sending a packet to the host (for more details see Section 3.3). Whenever a packet arrives at an attachment point, i.e., a hop that is specified in the loose source route, routing towards the next hop only requires information about a *single* MAN or the attachment points of the WAN. For this reason, individual MANs can run separate routing protocols with MAPs as routing locators. In contrast, inside the WAN all participating routers need to agree on a common routing protocol where WAPs are used as locators. Provided that the number of WAPs is limited<sup>2</sup>, the routing table size will be manageable as well. Regarding update churn, we are convinced that in the proposed hierarchical scheme the majority

<sup>2</sup>WAPs may correspond to the Points of Presence (PoPs) that are used to interconnect transit providers in today’s Internet. This number should be in the magnitude of 10,000 or 100,000 and is not expected to change considerably in the future (see [6]).

of updates will have local scope and thus only be visible inside a MAN or the WAN.

We direct the attention of the reader to two things. First, HAIR does not make any assumption on how the MAN layer is structured. In principle, it is possible that a LAN needs to traverse multiple MANs before reaching the WAN. In order to support an arbitrary number of hierarchical layers, we allow locators to have variable length. Nevertheless, for the sake of clarity, throughout the rest of this paper, we will assume a 3-layer hierarchy. Second, the structure in Figure 1 is orthogonal to the typical classification of ASs as “transit” and “stub”: in fact, even a transit AS usually has a portion of its network which is solely interested in network access. In practice, today’s transit ASs correspond to a MAN and a set of interconnected WAPs, while today’s stub ASs just map to a MAN.

The main advantage of our approach with respect to other proposals (e.g., [4, 9, 29]) is that the design of HAIR is in accordance with the structure of today’s Internet: To a certain degree firewalls, proxies and NATs can be seen as MAPs, while border routers of transit providers are similar to WAPs. Possibly, ISPs just have to upgrade such legacy equipment. Moreover, long-term trends show that the network is growing much more at the edge than in the core [6]. Since such evolution is mainly dictated by economic reasons, which are unlikely to change in the future, distinguishing between a stable network core and a fast-growing edge makes our proposal future-proof. After all, HAIR shows excellent scalability when adding MANs or LANs.

### 3.3 Packet Delivery

Sending a packet across the network consists of three parts: routing within the source MAN, within the core, and within the destination MAN. MANs act as independent routing domains, and, in principle, each one can use a separate address space as well as any arbitrary routing protocol. The only portion of the network that needs a common address space (and a common inter-domain protocol) is the WAN.

In the following, we illustrate with an example how packets are forwarded from a source  $A$  (identifier  $ID_A$ ) to a destination host  $B$  (identifier  $ID_B$ ). Note that locators consist of a WAP and a MAP. While  $A$  is located in a LAN that is reachable over  $WAP_A$  and  $MAP_A$  (locator  $LOC_A = WAP_A|MAP_A$ ), the destination host has the locator  $LOC_B = WAP_B|MAP_B$ . In the following we will assume that host  $A$  knows its own identifier and locators<sup>3</sup> as well as the domain name of destination  $B$ . Figure 2 summarizes how the packet is forwarded to the destination.

Before host  $A$  can send a packet to  $B$  it has to find out the identifier for  $B$  via DNS. Then it queries the mapping system to determine for the obtained identifier  $ID_B$  the corresponding locator  $LOC_B = WAP_B|MAP_B$ . Finally, it composes the destination address ( $WAP_B|MAP_B|ID_B$ ), adds it to the packet

<sup>3</sup>If needed this information can be delivered to the host, e.g., via DHCP or routing advertisements.

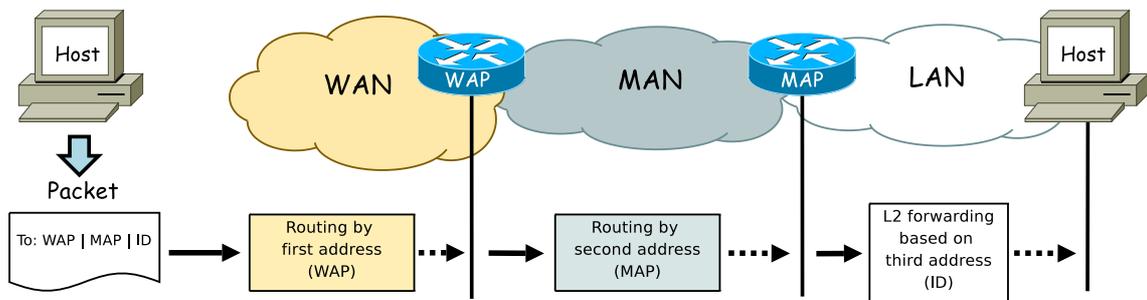


Figure 2: Packet forwarding in HAIR.

and also includes the source address  $WAP_A|MAP_A|ID_A$  to avoid that the destination has to do a locator lookup for the reply.

After generating the packet, it needs to be forwarded to the WAN as the destination  $B$  is not located in  $MAN_A$  but in  $MAN_B$ . There are two possibilities here: Either each MAN installs default routes towards the WAN or the packet uses the reverse loose source route that is encoded in the locator  $LOC_A = WAP_A|MAP_A|ID_A$  of source  $A$ . Routing within the backbone is done based on the WAP portion of the destination address, that is,  $WAP_B$ . This attachment point then forwards the packet within its MAN to  $MAP_B$  of the destination address. There, the identifier part of the destination address  $ID_B$  is bound to a layer-2 address and sent to  $B$  within the LAN. The complete process of packet delivery is summarized in Figure 2.

Contrary to many other locator/identifier split approaches, identifiers in HAIR are routable neither in the WAN nor in the MANs. For this reason, a large identifier space cannot harm the routing system. Regarding locators, the MAP portion of an address only needs to be unique within the context of a single MAN. In this respect HAIR provides flexibility to use any routing protocol inside MANs or the WAN.

### 3.4 Mapping System

In HAIR a mapping service is needed to resolve identifiers to locators. Note that, much like DNS resolution, this step is only required when a host sends the first packet to a certain destination. To avoid unnecessary lookups, a host maintains a cache that stores a certain number of mappings between identifiers and locators. In this section we will give some details about the architecture of the mapping service.

The design of the mapping system takes into account the hierarchical structure of our routing architecture and thus divides the mapping functionality into two parts: A *global DHT* inside the WAN and a *Server of Authority* (SoA), maintained by an individual MAN or a set of MANs. While the SoA keeps the mappings of all identifiers that are currently located in its associated MAN(s), the global DHT stores for all existing identifier only a pointer to the appropriate SoA.

Let us assume that a host wants to resolve a certain identifier  $ID_B$ . First it queries the global DHT which returns a pointer to the SoA  $SoA_B$  that stores the actual mapping for

identifier  $ID_B$ . In general, the  $SoA_B$  directory service will be provided by the MAN to which the LAN of  $ID_B$  is currently attached. Then, we ask  $SoA_B$  for the locator of  $ID_B$  and get as reply the locator  $LOC_B = WAP_B|MAP_B|ID_B$ . Accordingly, mapping an identifier to a locator involves two queries. Either both requests are triggered by the host or only the first query. In the second case, the host only asks the global DHT which then forwards the request to the appropriate SoA and has this SoA answer the mapping request directly back to the host. Observe that the two required steps can be easily merged, e.g. by piggybacking either the SoA or the mapping itself into a DNS reply.

While HAIR relies on a DHT to resolve identifiers to SoAs, we leave it to the responsible MAN(s) how to implement the SoA service. In principle, any directory service that returns a value for a certain key can be used. However, our proposal strongly recommends a DHT for mapping identifiers to the corresponding SoAs. Our assumption is that transit ISPs in the WAN provide nodes to participate in the DHT. To enforce this, registries could require the allocation of DHT nodes (or clusters) before assigning AS numbers to transit providers. The main reason for relying on a DHT is high scalability: The number of entries will correspond to the number of all existing identifiers/hosts. To deal with such a huge number – imagine a list of all hosts that participate in today’s Internet – DHTs are apparently a good design choice, as the number of participating nodes can be easily increased.

If people criticize that our solution simply shifts burden from the routing to the mapping system, they are right to some extent. However, there are two major reasons for doing so: First, distributing a mapping table is apparently simpler than distributing routing tables [29]. Second, the separation allows us to tune both systems separately. As already discussed, we can arbitrarily scale the mapping system by inserting additional nodes into the DHT. Furthermore, the mapping service can be offered by commodity systems at relatively low cost and does not require expensive router hardware.

The main reason for having two different types of mapping components – global DHT and SoA – is to limit update churn. Hosts may change to a different LAN which requires updating the corresponding entries in the mapping system.

Our assumption is that hosts frequently connect to another LAN that is attached to the *same* MAN. This is plausible since nodes will frequently just move to a geographically-close location and thus continue to be associated with the same MAN. In this case, the host only needs to change the MAP in the SoA that is maintained by the current MAN. No updates are required for the global DHT. Provided that a host moves to another MAN, updating the mapping system becomes slightly more complex, but is expected to be less frequent. Now, the locators of the host are transferred from the previous SoA to the new SoA and the MAP portion of the locator is changed to the new MAP. Furthermore, the global DHT is told that the locators of the host are now stored in the new MAP. At first glance, these operations seem time-expensive. However, updating a DHT basically involves a lookup to find the node where the information needs to be changed. Compared to adding or removing nodes in a DHT, this operation is fast.

We point out that the authority maintaining the SoAs have great power on the paths chosen for forwarding. After all, they return a loose source route for an identifier and thus determine along which hops packets are sent to the destination. In contrast, the global DHT in the WAN does not have a strong impact as it merely points to SoAs that store the actual locators (WAP+MAP). Various options to control the forwarding paths by leveraging the SoAs will be described in Section 4. For example, SoAs can enforce traffic engineering or load balancing, by returning multiple or different locators for a given identifier.

We finish the description of our mapping service by discussing two bootstrapping issues. First, the locators of the mapping servers – SoA or nodes participating in the global DHT – must be well-known. Second, inserting new nodes to the global DHT is a challenging problem. The difficulties stem from the fact that there is no trivial way to locate an arbitrary node in the Internet. However, in our case participation in the global DHT has to be regulated by a third party (e.g., routing registries) which can provide a list of nodes that are already participating in the DHT.

## 4. EVALUATION

The preceding section has introduced the design of HAIR. Now, in Section 4.1 we discuss whether our proposal meets the challenges of a routing system for the future Internet and the design requirements explained in Section 3.1. Then in Section 4.2 we present a simple prototype implementation and demonstrate a possible migration path that provides support for legacy hosts.

### 4.1 Discussion

In the following we briefly discuss the extent to which our architecture meets the design requirements. At the same time, we outline how shortcomings of today’s routing system are mitigated and eliminated with our solution:

**Scalability of the routing system:** The number of routing table entries for routers in the WAN corresponds to the number of existing WAPs. Thus, routing table size in the core will not be a problem, since we expect that this number will be in the magnitude of 10,000 or 100,000. After all, a WAP may match Points of Presences (PoPs) of the transit providers in today’s Internet. A nice property of our architecture is that MANs constitute isolated routing domains, where routing protocols of different MANs do not communicate with each other. Hence, routing inside access network does not pose any scalability issues and offers the advantage that routing updates have local scope. Finally note that routing inside the WAN or MAN is solely done based on WAPs and MAPs respectively. Therefore, there will be no shortage of routing addresses as with global-routable IPv4 addresses in the current Internet.

**Scalability of the mapping system:** Regarding the mapping system, we differentiate between two major components. With individual MAN(s) maintaining their own SoA, the size of the number of entries in the individual SoAs is not a problem. Scalability of the global directory service that keeps for all identifiers a pointer to the corresponding SoA, is achieved by relying on DHTs. Finally, the hierarchical design of our architecture localizes updates to the mapping system: Based on our assumption that changes between different MANs are infrequent, only rare updates are required to the global DHT.

**Mobility:** HAIR inherently supports mobility due to the separation of locators and identifiers. In Section 3.4 we explained how the mapping system needs to be updated when a host moves to a different LAN that is attached to the same or even a different MAN. Note that it is not compulsory to change the pointer in the global DHT to the new SoA whenever a host moves to a different MAN. Rather it is also possible that MANs who have an agreement to support “foreign” identifiers set up cooperating SoAs. In this case, the SoA of the previous MAN before the network change continues to store the locators of the mobile host. We point out that seamless hand-overs during network changes are not the primary objective of our architecture. However, note that session survivability is automatically achieved because packets carry both the locator and the identifier. Hence the new locator for a given identifier can be inferred by the remote endpoint as soon as the first end-to-end data packet (carrying the new locator) is received. This allows lightweight hand-overs with session survivability. New connections, on the other hand, need to wait for the mapping system to be updated<sup>4</sup>.

Network mobility, i.e., changing providers, has become

<sup>4</sup>one can easily add existing approaches, such as [20], on top of HAIR to handle new connections while the mapping system has not yet been updated

such a common task that provider independent (PI) addresses are greatly preferred to provider aggregatable (PA) addresses in today’s Internet. Let us assume a MAN wants to add a new provider in our architecture. All that needs to be done is to update the local SoA to produce locators reflecting the newly attached WAPs. These changes are easy and can be applied locally.

**Multipath:** Let us assume that the responsible SoA returns more than one locator for a given identifier. In this case, multiple locators can be used simultaneously to send traffic to a host. Leveraging multiple paths for the same session can increase throughput and availability and allows for load balancing. Note that the example of shim6 [19] demonstrates how to negotiate a set of locators and how to determine which subset of them is to be used for the communication. However, we point out that multipath within the same transport session is often debated to the overhead of handling out-of-order packets. Answering such questions is beyond the scope of this paper. Our proposal is flexible in this respect and lets end hosts autonomously choose whether to use a single locator, different locators for different transport connections or multiple locators within the same connection. Since the source locators are included in the packet header, locators can easily be switched during a session by piggybacking a different locator in the packet.

**Traffic engineering:** Knowing more than one locator for a certain destination allows to perform traffic engineering. A MAN can influence where traffic enters its network by simply changing the locators stored in the SoA. In principle, this provides them with a tuning knob to efficiently manage inbound traffic at the host granularity and to prevent hosts from autonomously interfering with network-wide TE policies. Note that end hosts can also influence how they are reachable. Provided that there exist multiple locators for a host, MAN providers can allow end hosts to tell the SoA which locators are to be favored and are to be returned as result to a query. Altogether, HAIR relieves the core network from having to cope with the consequences of inbound traffic engineering as it is the case with today’s Internet, where routers in the DFZ suffer from the injection of more-specific prefixes by multi-homed ASs.

**No changes to the core:** Any new components or mechanisms which are introduced as part of HAIR are placed as close as possible to the edge, i.e. in MANs. For example, the actual mappings between identifiers and locators are stored in SoAs, that are maintained by MANs, and they are not kept in a central directory somewhere in the WAN.

**Easy migration:** In the design of HAIR we devise a smooth migration path that allows ISPs to change a minimal number of devices and that supports legacy routers and hosts.

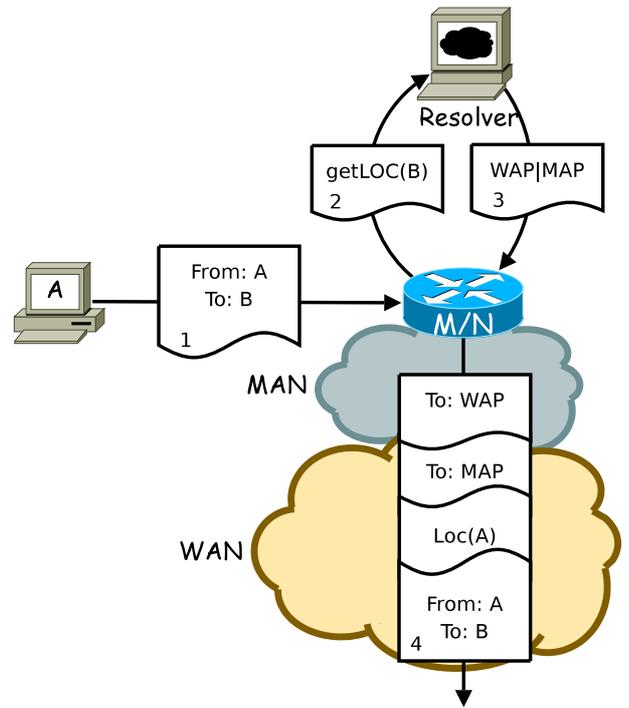


Figure 3: MAP/NAT box.

For more details we refer to the description of our prototype in Section 4.2.

## 4.2 Prototype

We now briefly present a prototype implementation for our architecture. First and foremost such a prototype has to be seen as a working proof-of-concept. For this reason, we deliberately disregard potential performance issues such as initial delay and overhead. The second motivation for our prototype is to demonstrate a possible migration path. Our proposal as presented in Section 3 requires changes to every host which will obviously hamper the deployment of HAIR. With our prototype we also sketch how to support legacy routers and hosts during the migration phase.

The approach, we take to achieve support for legacy hosts, is to introduce a middle-box, the so-called *MAP/NAT* box. Figure 3 depicts how our *MAP/NAT* box works. Upon receiving a packet from a legacy host, such a box queries the mapping system to resolve the locator of the destination address. Then it writes its own locator as source into the packet header and forwards the packet to the locator. On the other hand, when receiving a packet destined to a legacy host, the *MAP/NAT* box extracts the source locator from the header and caches the inferred mapping for increased efficiency. Finally, it transforms the packets into a standard IP packet and delivers it to the legacy host.

The implementation of our *MAP/NAT* box adopts a “map and encaps” scheme. When a *MAP/NAT* receives an IPv4 packet from a LAN, it adds an extra IP-in-IP encapsulation header for every attachment point contained in the locator.

Provided that the packet needs to be sent to locator  $LOC_X = WAP_X|MAP_X$ , it is encapsulated twice: once for  $MAP_X$  and once for  $WAP_X$ . Finally, note that we use IP addresses as locators for attachment points.

Altogether, our prototype suggests a possible migration path. To achieve incremental deployment, we basically need to provide the WAP functionality for transit providers and to deploy MAP/NAT boxes inside each LAN. For this purpose, we propose to leverage existing devices: For many of the changes it is sufficient to upgrade the firmware at the border routers of transit providers and to modify middle-boxes such as firewalls, gateways or proxies that already exist anyway in today's LANs.

More challenging is the interaction between upgraded and legacy sites if there exist LANs that do not provide a MAP/NAT service. Assume that a host inside an upgraded network wants to send traffic to a legacy host for which only an IPv4 address is known. In this case, the packet needs to be delivered to the legacy network using traditional IP routing. Therefore, the coexistence of HAIR and of today's Internet routing is required during the migration phase. With respect to incoming traffic, where traffic is sent by a legacy host to an upgraded host in a network, we identify two solutions: Either globally routable IDs or translation techniques are used. In both cases, the legacy host needs to know the identifier of the upgraded destination host. Obviously, enabling full compatibility to legacy sites without a MAP/NAT box is challenging.

However, given that such boxes are simple, cheap and easy to deploy, we believe that the deployment of MAP/NAT boxes is the migration approach that should be adopted. Furthermore, they match our design goal of keeping functionality as close to the edge of "new" Internet as possible: They are deployed at the border between a LAN and a MAN and are thus close to the "edge". Note that this is necessary to buffer packets until destination identifiers have been translated to locators. Finally, first experiences with our prototype implementation are promising and underline their importance for a smooth transition to the new routing architecture HAIR.

## 5. CONCLUSION

The routing architecture of today's Internet suffers from several shortcomings: Super-linear routing table growth, high update churn, lack of mobility and security, insufficient support for multi-homing and traffic engineering are some examples. Hence many researchers believe that the Internet's shortcomings cannot be resolved by the conventional incremental and 'backward-compatible' style of research.

The architecture HAIR, we introduce in this paper, proposes a scalable routing architecture for the future Internet. The focus is on limiting routing table size and update churn while supporting legacy hosts and avoiding unnecessary burden for transit providers. The key idea is to combine a hierarchical routing approach with locator/identifier separation:

The routing as well as the mapping system are organized in a hierarchical manner where updates to both systems are not globally visible as far as possible.

First experiences with a prototype implementation are promising and demonstrate a potential migration path which supports the interaction between hosts in legacy and upgraded networks. Future work will include large-scale experiments to estimate the potential gains in terms of scalability compared to today's Internet routing system. Our proposals address the most serious problems the Internet is facing at the moment. Therefore, we are convinced that HAIR is an enticing alternative to the numerous architecture proposals, suggested in the past.

## 6. ACKNOWLEDGMENT

The research results presented herein are partly funded by a grant from Deutsche Telekom Laboratories.

## 7. REFERENCES

- [1] Global Environment for Network Innovations. <http://www.geni.net/>.
- [2] ATKINSON, R. ILNP Concept of Operations, Internet Draft, draft-rja-ilnp-intro-01.txt, 2008.
- [3] CAESAR, M., CASTRO, M., NIGHTINGALE, E., O'SHEA, G., AND ROWSTRON, A. Virtual Ring Routing: Network routing inspired by DHTs. In *Proc. ACM SIGCOMM* (2006).
- [4] CAESAR, M., CONDIE, T., KANNAN, J., LAKSHMINARAYANAN, K., AND STOICA, I. ROFL: routing on flat labels. In *Proc. ACM SIGCOMM* (2006).
- [5] DAY, J. *Patterns in Network Architecture: a Return to Fundamentals*. Prentice Hall, 2008, pp. 297–316.
- [6] DHAMDHARE, A., AND DOVROLIS, C. Ten years in the evolution of the internet ecosystem. In *Proc. ACM IMC* (2008).
- [7] ERIKSSON, A., AND OHLMANN, B. Dynamic Internetworking Based on Dynamic Locator Construction. In *Proc. IEEE Global Internet Symposium* (2007).
- [8] FARINACCI, D., FULLER, V., AND MEYER, D. LISP Alternative Topology (LISP+ALT), draft-fuller-lisp-alt-02.txt, 2008.
- [9] FARINACCI, D., FULLER, V., ORAN, D., MEYER, D., AND BRIM, S. Locator/ID separation protocol (LISP). Internet draft, draft-farinacci-lisp-08.txt, 2008.
- [10] FELDMANN, A. Internet Clean-slate Design: What and why? *SIGCOMM Comput. Commun. Rev.* (2007).
- [11] GAVOILLE, C. Routing in distributed networks: overview and open problems. *SIGACT News* 32, 1 (2001), 36–52.
- [12] HUSTON, G. BGP Routing Table Analysis Reports. <http://bgp.potaroo.net/>.
- [13] INITIATIVE, N. N. Future Internet Design. <http://www.nets-find.net>.
- [14] JEN, D., MEISEL, M., YAN, H., MASSEY, D., WANG, L., ZHANG, B., AND ZHANG, L. Towards a New Internet Routing Architecture: Arguments for Separating Edges from Transit Core. In *Seventh ACM Workshop on Hot Topics in Networks (HotNets-VII)* (2008).
- [15] KASTENHOLZ, F. A new routing and addressing architecture. IRT, Internet Draft, 2002.
- [16] KLEINROCK, L., AND KAMOUN, F. Hierarchical routing for large networks: Performance evaluation and optimization. *Computer Networks* (1977).
- [17] MATHY, L., IANNONE, L., AND BONAVENTURE, O. Lisp-dht: Towards a dht to map identifiers onto locators. February 2008.
- [18] MOSKOWITZ, R., AND NIKANDER, P. Host Identity Protocol (HIP) RFC 4423, 2006.
- [19] NORDMARK, E., AND BAGNULO, M. Shim6: Level 3 multihoming shim protocol for IPv6. Internet draft, draft-ietf-shim6-proto-10.txt, 2004.
- [20] PERKINS, C. IP Mobility Support for IPv4, RFC 3344.
- [21] PROJECT, E. F. P. . I. Trilogy. <http://www.nets-find.net>.
- [22] RATNASAMY, S., FRANCIS, P., HANDLEY, M., AND KARP, R. A Scalable Content-Addressable Network. In *Proc. ACM SIGCOMM* (2001).
- [23] RESEARCH PROJECT, D. NewArch Project: Future-Generation Internet Architecture. <http://www.isi.edu/newarch>.
- [24] STOICA, I., ADKINS, D., ZHUANG, S., SHENKER, S., AND SURANA, S. Internet Indirection Infrastructure. In *Proc. ACM SIGCOMM* (2002).
- [25] STOICA, I., MORRIS, R., KARGER, D., KAASHOEK, M., AND BALAKRISHNAN, H. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proc. ACM SIGCOMM* (2001).
- [26] SUBRAMANIAN, L., CAESAR, M., EE, C., HANDLEY, M., MAO, M., SHENKER, S., AND STOICA, I. HLP: A Next Generation Inter-domain Routing Protocol. In *Proc. ACM SIGCOMM* (2005).
- [27] VOGT, C. Design Taxonomy and Analysis for Address-Indirection-Based Routing Scalability Improvements. 2008.
- [28] VOGT, C. Six/one router: a scalable and backwards compatible solution for provider-independent addressing. In *Proc. MobiArch* (2008).
- [29] ZHANG, X., FRANCIS, P., WANG, J., AND YOSHIDA, K. Scaling IP routing with the core router-integrated overlay. In *Proc. ICNP* (2006).