Design and Implementation of the Node Identity Internetworking Architecture

Simon Schütz[2], Henrik Abrahamsson[1], Bengt Ahlgren[1], and Marcus Brunner[2]

[1]SICS, Box 1263, 164 29 Kista, Sweden, e-mail: [henrik | bengta]@sics.se
[2]NEC Europe Ltd, NEC Laboratories, Heidelberg, Germany, e-mail [Schuetz | brunner]@nw.neclab.eu

SICS Technical Report T2008:13
ISSN 1100-3154

2008-12-05

Abstract

The Internet Protocol (IP) has been proven very flexible, being able to accommodate all kinds of link technologies and supporting a broad range of applications. The basic principles of the original Internet architecture include end-to-end addressing, global routeability and a single namespace of IP addresses that unintentionally serves both as locators and host identifiers. The commercial success and widespread use of the Internet have lead to new requirements, which include internetworking over business boundaries, mobility and multi-homing in an untrusted environment. Our approach to satisfy these new requirements is to introduce a new internetworking layer, the *node identity layer*. Such a layer runs on top of the different versions of IP, but could also run directly on top of other kinds of network technologies, such as MPLS and 2G/3G PDP contexts. This approach enables connectivity across different communication technologies, supports mobility, multi-homing, and security from ground up. This paper describes the Node Identity Architecture in detail and discusses the experiences from implementing and running a prototype.

Design and Implementation of the Node Identity Internetworking Architecture

Simon Schütz[2], Henrik Abrahamsson[1], Bengt Ahlgren[1], and Marcus Brunner[2]

[1]SICS, Box 1263, 164 29 Kista, Sweden, e-mail: [henrik | bengta]@sics.se

[2]NEC Europe Ltd, NEC Laboratories, Heidelberg, Germany, e-mail [Schuetz | brunner]@nw.neclab.eu

ABSTRACT

The Internet Protocol (IP) has been proven very flexible, being able to accommodate all kinds of link technologies and supporting a broad range of applications. The basic principles of the original Internet architecture include end-to-end addressing, global routeability and a single namespace of IP addresses that unintentionally serves both as locators and host identifiers. The commercial success and widespread use of the Internet have lead to new requirements, which include internetworking over business boundaries, mobility and multi-homing in an untrusted environment. Our approach to satisfy these new requirements is to introduce a new internetworking layer, the *node identity layer*. Such a layer runs on top of the different versions of IP, but could also run directly on top of other kinds of network technologies, such as MPLS and 2G/3G PDP contexts. This approach enables connectivity across different communication technologies, supports mobility, multi-homing, and security from ground up. This paper describes the Node Identity Architecture in detail and discusses the experiences from implementing and running a prototype.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols – *protocol architecture, protocol verification*

Keywords

Communication systems, Future Internet, mobile communication, Internet Architecture.

1. INTRODUCTION

The basic principles of the original Internet architecture include end-to-end addressing, global routeability and a single address space of IP addresses that act as locators and node identifiers at the same time. These principles are suitable for static and well-managed network hierarchies. However, since the Internet has evolved from a small research network to a worldwide information exchange network, a growing diversity of commercial, social, ethnic, and governmental interests have led to increasingly conflicting requirements among the competing stakeholders. These conflicts create tensions that the original Internet architecture struggles to withstand.

The commercial success and widespread use of the Internet have lead to new requirements for a future Internet, which include internetworking over business boundaries, mobility, multi-homing, and security for untrusted environments. Concurrently with this research into new internetworking architectures, a demand for private, autonomous networks is growing. Although still connected to the global Internet, these autonomous networks offer local features and capabilities that are independent from the public Internet. The today's solution to achieve more autonomy are Network Address Translators (NAT) [11], which is a popular method for reusing address space and decoupling routing in the private network from routing in the public Internet. Although these capabilities of NATs mitigate many immediate problems, NATs are not a clean solution [12].

The fundamental problems of the Internet Protocol stems from overloading two separate functionalities onto the same bit string of the IP address. One is its use as a locator, i.e., as an address that denotes a location in the topology of the network and specifies a network attachment point (interface). The second one is that of an identifier that describes the identity of a node.

The problem with the NAT approach is that it translates between internal and external addresses and with that also implicitly translates between the associated identities. This causes applications and protocols that exchange IP addresses in their payloads, such as FTP, to break.

The problem with addressing a network attachment point is that today most hosts have more than one communication capability, and with it the possibility to attach to the network through several interfaces. This multi-homing causes the host to show up with multiple interface addresses, and thus multiple identities. Furthermore, the Internet is an untrusted network, and more emphasis is required to secure the communication across the network.

Finally, designing and implementing a new Internetworking architecture always poses the question on the feasibility of deploying it globally. Therefore, measures must be taken to design for gradual deployment, such that early adopters benefit, without others having to migrate. Also the changes to the network required, should be limited and management of the network should be minimal. Finally, the architecture needs to scale up to global deployment.

The Node Identity Internetworking Architecture [1][2] is designed to address the above described challenges architecturally, and not as an after thought or patch of the today's Internet. Its key characteristics are the separation of a node's identity from its location, and the notion of locator domains. Multiple locator domains, which may use different technology, addressing and/or routing schemes, are intrinsically supported, where at locator domain borders a process similar to twice-NAT is performed. Mobility and host multi-homing are supported by decoupling the Node Identity from the topological meaning of the addresses. Finally, security is built-in through the usage of cryptographic Node Identities.

This paper describes the Node Identity Architecture in detail and discusses the experiences from implementing a prototype. The remainder of this document is organized as follows. In section 2 we describe the Node Identity Architecture including assumptions, principles and key features. Section 3 introduces some specific implementation options and section 4 presents the prototype implementation used for the experimental evaluation and its results. A discussion on the proposed architecture is provided in section 5. Section 6 presents some related work and the differ-

ences to the Node Identity Architecture. Finally, the paper is concluded in section 7.

2. NODE IDENTITY ARCHITECTURE

2.1 Assumptions and Principles

The Node Identity Internetworking Architecture, NodeID in short, is based on two main ideas. The first idea is the notion of a *node identity layer* directly on top of existing networking technology. This layer provides unique cryptographic identifiers for nodes, called *node identities* (NIDs), which are independent of the node's current location and network address. The second idea is to perform routing in the node identity layer using the node identifiers with the purpose of providing mobility and multi-homing, as well as bridging heterogeneous address domains. The first of these ideas is shared with HIP, the Host Identity Protocol architecture [4].

The NID is the public part of a randomly self-generated public/private key pair. This allows the use of the NID for authentication purposes between nodes (not between users). The NID is from a flat identifier space without topological semantics, that is, it has no location information within the network topology. In addition, a node has one or several locators. In contrast to the NID, the locators usually have topological semantics, but that depends on the specifics of the networking technology in use. The locator is used to route messages to a node within its network. It can be e.g. an IPv4 or IPv6 address. The locators of a node may be assigned to one or multiple interfaces and can be of different kinds.

The architecture defines two basic components: *locator domain* (LD) and *node identity router* (NR). A locator domain is the abstraction of a network having a consistent internal addressing and routing system. Nodes within one LD can freely communicate, only relying on internal services of the respective LD. Different LDs may employ different networking technologies, in particular IPv4, IPv6, global and private address spaces, MAC addresses, or any other technology-specific addressing like 2G and 3G PDP contexts or MPLS. This implies that an LD boundary may also be a network technology boundary. The Node Identity Internetworking architecture does not aim to unify the global network into a single, globally deployed networking technology, but accepts or even encourages the existence of different networking domains. E.g. the global IPv4 network can be seen as one LD, while private, NAT'ed networks can be seen as separate LDs. The concept is deliberately flexible in that an LD border can be defined even when there is no technology or addressing border requiring one.

The locator domains are interconnected by node identity routers (NID routers, or just NRs for short). NID routers forward packets using a NID routing table, similar to how IP routers forward packets using an IP routing table.

Connectivity between locator domains is assumed to be dynamic, especially in the edge of the global topology. That is, the existence and characteristics of connectivity between two locator domains, and between nodes and locator domains, may change dynamically on relatively short timescales, due to routing changes, mobility and multi-homing events, or provider change of nodes or networks.

2.2 NodeID Routing

Routing within the Node Identity Internetworking Architecture follows a two-level approach:

1. *Intra-LD*: Routing between nodes within an LD is solely based on the internal routing scheme of the respective LD using the locators. Each LD can therefore have a different routing scheme. Note that one node can belong to more than one LD if it has more than one locator. A locator within one LD has absolutely no meaning within another LD. As a consequence, overlapping locator/address spaces are a non-issue and locators do not need to be globally managed. They only need to be unique within a single LD.
2. *Inter-LD*: Routing across LD borders is based on the NID of the node, or more precisely on a fixed-length hash of its NID – the *NID Forwarding Tag* (NIFT).

Intra-LD routing is specific to the technology used within an LD and is independent of the routing scheme used for Inter-LD routing on the Node ID level. It will therefore not be further discussed in this paper.

2.2.1 Inter-LD Routing Approach

Inter-LD NID routing using flat NIFT labels creates a potentially very large scaling problem. Routing for individual nodes using a flat namespace would either require registration with a central lookup service, or a global routing scheme for flat labels. The former is for many reasons infeasible. The latter has been investigated to some extent [14], but there is not a clear solution that we want to depend on.

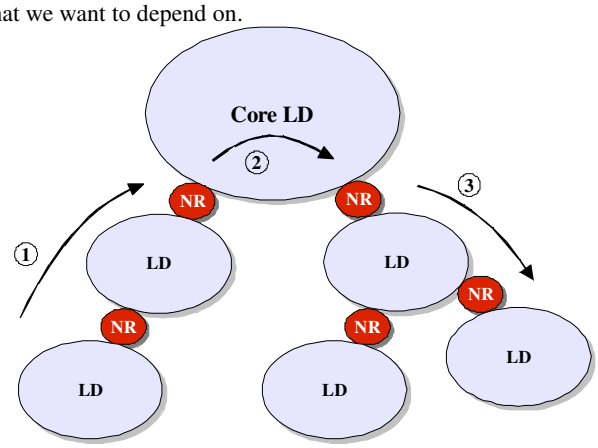

Figure 1: Topology Assumption and Routing Overview

To limit the scalability problem, we assume that there will be a *core LD*, which is rather static. The core LD can be compared to the current IPv4 backbone. Other LDs then attach in smaller topologies at the edges of this core LD, forming tree-like structures hanging off the core. This assumption is motivated by that dynamicity – such as frequent (node or network) mobility and routing changes – is mostly expected to happen at the edges of the topology.

With this assumption, inter-LD routing is then addressed by separating the global routing into three parts as illustrated in Figure 1:

1. Routing up the edge trees towards the core-LD
2. Routing through the core-LD
3. Routing from the core-LD down the trees to the edge

2.2.2 *Routing in the edges*

Part 1 – routing up the edges towards the core-LD – is using default routing. Default routes are installed when connecting networks, similar to the installation of default IP gateways in IP networks.

Part 3 – routing from the core-LD down the trees towards the edge – is solved by a registration-based approach. Every node needs to register its presence along a default route to the core-LD, thus installing the routing information for the reverse path. Assuming that edge networks are not too large, this scales as any NR only needs to hold registration information for nodes within the edge network, which it connects towards the core-LD, and not for all nodes.

This registration procedure is really a simple routing protocol, since it establishes routing state. As a side comment, more advanced routing schemes can be applied to for instance enable larger edge structures or providing short-cuts in the edge topologies [20]. Also different protocols can be deployed in different edges implementing capability awareness, multi-path routing, or other more advanced features.

2.2.3 *Routing Through the Core – The Routing Hint*

One solution to routing over the core LD is to have a global database, or lookup service, which provides the mapping from NIFT to locator. We however do not want to depend on such an assumption. Instead, we introduce the notion of a *routing hint*. It is similar to a partial source route. It has to be specified by the source node in the packets it is sending. When a NID router does not have any matching routing information for a packet, including a default route, it makes use of the routing hint.

Figure 2 illustrates how a packet is routed from node A to node B through the core locator domain. Part 1 uses default routing, and part 3 uses routing state established with registration, as described above. When the packet arrives to NR1, there is no information in its routing table for the destination (not even a default route). NR1 therefore looks at the routing hint, finds the locator of NR2, and forwards the packet to NR2.

For the return path there are two options. Either node A specifies its routing hint in the packet (source hint), or NR1 inserts itself as source hint when deciding to use the destination hint in the packet. The options do not make a big difference in this simple example, but for instance has consequences for location privacy.

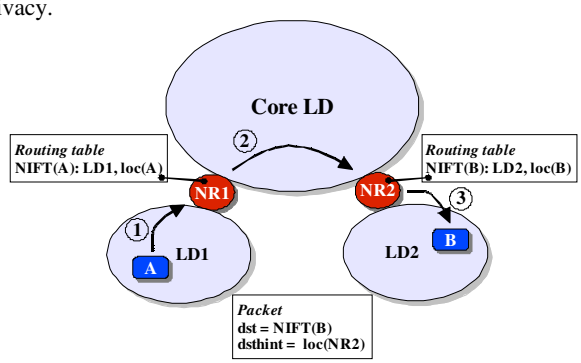

Figure 2: Routing through the core with the routing hint

It is most convenient to let the routing hint be a locator (IP address). That does not require any additional machinery. It could however be useful to let the routing hint also be a flat label with the same format as a NIFT, especially in combination with a more advanced routing protocol for edge trees.

Please note that not necessarily all three parts need to be used to route a packet from source to destination. For example, if source and destination are within the same edge network, there is no need to route the packet to the core-LD. In general, routing towards the core-LD is only necessary if a NR does not hold any more specific routing information for the destination. Otherwise, it should use the more specific information and route directly towards the destination.

The intention is that the destination routing hint is looked up in the DNS together with the destination NIFT. The DNS service therefore needs to be relayed down the edge LD trees without depending on NID routing. That can for instance be done by co-locating query forwarding DNS servers with the NID routers.

The routing hint serves as a generic indirection mechanism that can be used for many purposes. We will later see how it can be used to support mobility and multi-homing.

2.2.4 *Detailed Example*

Figure 3 presents the sample topology from Figure 2 with more details. It shows one core LD (LD1), which can be compared to the current IPv4 backbone, as explained above. There are two LDs (LD2 and LD4) that are attached to the core LD through NR2 and NR3. On a third-level in the LD-topology there is LD3 attached to LD2 through NR1 as well as LD5 and LD6 that are attached to LD2 through NR4 and NR5. The various LDs use different internal addressing schemes, e.g. IPv4 and IPv6, global and local addresses.

In addition, there is a number of nodes present that are not acting as NRs (nodes A to K). However, some of these nodes can still act as internal routers for their respective LDs. In Figure 3, the nodes C, D, E, F and G act as IP routers, but not as NRs.

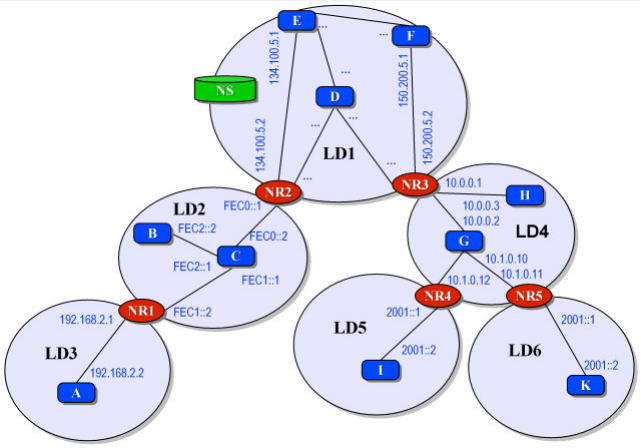

Figure 3: Detailed Sample Topology

Nodes within one LD, as described before, can directly communicate using the LD internal addressing and routing scheme. As an example, in the above figure node A can directly talk to NR1 using the LD-internal routing and addressing scheme. All nodes need to register their NIFT along a default path of NRs to the core LD. For example, node A needs to register at NR1 and NR2, while node K needs to register at NR5 and NR3. NR2 becomes the core NR responsible for node A and NR3 becomes the core NR responsible for node K.

As a complete example for communication establishment, if node A in Figure 1 wants to contact node K, the following actions need to be performed:

1. A sends a request to the global naming system (NS) to resolve the FQDN for K. This query returns K's NIFT and K's routing hint (i.e. NR3, as it connects K's edge network to the core LD).
2. As A does not know where K is located it sends the packet to its default NR (NR1) using the LD3-internal IPv4 network. In addition to the normal IP header, the packet needs to contain K's NIFT and K's routing hint.
3. NR1 as well does not hold any state about node K, re-writes the IP header according to the LD2-internal IPv6 locators and forwards the packet to its default NR (NR2).
4. NR2 does not hold state about node K. NR2 – as being a core NR – does not have a default route and therefore reads K's routing hint from the packet, which is NR3.
5. NR2 rewrites the IP header according to the LD1-internal IPv4 locators and forwards the packet to NR3.
6. NR3 knows from the registration information that K can be reached via NR5, rewrites the IP header to the LD4-internal IPv4 locators and forwards the packet to NR5.
7. NR5 knows from the registration information that node K is in its local LD, rewrites the IP header to the LD6-internal IPv6 locators and finally delivers the packet to node K.

2.3 Multi-Homing and Mobility

The NodeID architecture is designed with mobility and multi-homing in mind from the start. Mobility and multi-homing are supported directly by the routing capabilities of the NodeID overlay. No special concept of mobile IP home agent or HIP rendezvous server is needed. Any NID router can instead take on this role for a particular node or network. By organizing NID routers in a hierarchy, mobility signaling can be localized as needed, and mobility signaling can be aggregated as needed.

The already described routing hint plays an important role for mobility by providing a simple and effective mechanism for routing to a mobility anchor point for a node. The routing hint relieves the name resolution system from having to support updates due to mobility.

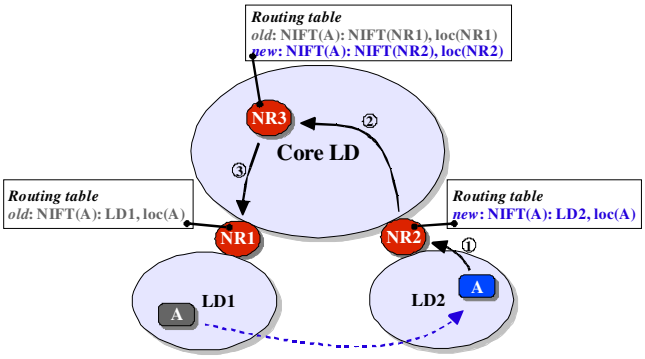

Figure 4: Node mobility using NID router as home agent

Figure 4 illustrates node mobility using a NID router as home agent. Node A moves from LD1, where it previously registered, to LD2. In LD2, A registers recursively (1 & 2) until old registration state is encountered (NR3 in this case, which acts as the home agent for A). This procedure localizes mobility signaling to the part of the network affected by the mobility event. (3) At the point where the old registration state is encountered, de-registration is done down the old registration path to remove stale routing information.

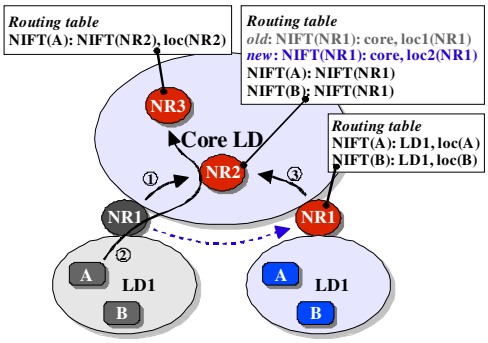

Figure 5: Network mobility

Network mobility is only slightly more complex. Figure 5 illustrates how network mobility is realized using another level of indirection. The NID router NR1 moves from one location to another as follows: (1) NR1 registers with its home agent, NR2. (2) Node A registers recursively to its home agent NR3. This path goes through NR1s home agent. When NR1 moves together with its network LD1, only NR1 needs to re-register (3) with its home agent.

A common problem of many mobility schemes using the concept of a home agent is that routing may become very inefficient. The NodeID forwarding mechanism therefore has a route redirect function. A NID router detecting that the next hop is in the same locator domain as the previous hop in the forwarding path can issue a route redirect message to the previous hop with information about a more direct route. This function is in principle very similar to ICMP route redirect for IP. The cryptographic node identities however make it possible to provide route redirect securely by a sender signature and a return routability check.

Multi-homing is supported by allowing multiple routing entries for the same destination, each entry representing one of the options for reaching the destination. The multiple routing entries can be held at one or several multi-homing anchor points, and/or show up as multiple routing hints. In the former case, the path selection control stays with the multi-homed node/network, while in the latter case, the correspondent node has the option to choose path. The route redirect function can also be useful to control the path taken.

2.4 Security Features

The use of cryptographic identifiers naturally enable a set of security features built-in from ground up in the architecture. Binding these identifiers to persons or organizations is however out of scope of this paper. The registration and de-registration is authenticated, and therefore the authorization of a node to register is based on the local polices applied in a locator domain and along the path up to the core. The authorization basically means a node is reachable through a certain NID router.

For incoming traffic at each NID router, the source is authenticated if required (again a policy decision). When the authentication process has been performed, a router can decide whether to

pass traffic or block traffic. This allows at any place to fend off unwanted traffic. Naturally, the policy engine need to know about the sources allowed on a per-routing hint or on a per-node basis. So the known prefix based firewall rules are not possible anymore, but the source of the traffic is proven and can be checked back (address spoofing attacks are not possible anymore). Rule aggregation can instead be done on a certificate authority basis. Based on those basic features more complex overlays for security management can be built and maintained.

Location privacy is enabled to a certain degree. When a node communicates within an LD, no privacy is achieved. Across LDs location privacy is given to the degree the routing hint does tell something about the location, or path to the location. When a home agent NR is used, that home agent can decide whether to conceal the location or reveal it via the route redirect function mentioned above. Since there is no restriction in the number of NIDs a node can have, location privacy can be achieved through multiple NIDs.

3. IMPLEMENTATION ISSUES

The previous section has presented the general design of the NodeID architecture. In the following, we discuss a set of implementation specific issues that should get considered when instantiating the architecture.

3.1 Stateful vs. stateless (connection/session state)

So far in this paper, the forwarding of data packets in the NodeID architecture has been described assuming a connectionless paradigm, that is, each data packet is individually forwarded by NID routers using a NID routing table. This approach introduces a quite large packet size overhead due to the additional packet headers required for NIFTs and routing hints. The advantage is that the NID routers do not need to store session state for ongoing communication.

The alternative is to introduce a session set-up step in which the NID routing takes place, and where subsequent packet forwarding then uses a more efficient session state. An example of this approach is the HIP base exchange [4] together with SPINAT [19] which use the IPsec SPI values, established during the base exchange, to forward the data packets. The advantage of this approach is that the packet header overhead is much smaller compared to the connectionless approach, since the large NIFTs and routing hints are only needed as part of the session set-up. The disadvantages are that the explicit signaling adds latency before communication commences, that the NID routers have to store the session state, resulting in a potential scalability problem, and that changes in routing, e.g., due to mobility, requires updating all session state in addition to changing the NID routing tables. However, when secure communication is required this overhead is required anyway.

3.2 Iterative vs. Recursive NID Registration

A node that wants to be globally reachable needs to register its NID along a set of NRs from its own LD towards the core LD. This registration follows a default route to the core, and builds the route from the core NR to the node in its local LD. The way NID registration is performed can be designed in a number of ways: recursive, iterative or in a mixed version. The details are discussed in the following.

In *recursive operation*, a node sends a registration request to its local (first hop) NR and waits for a response. The local NR is then in charge of registering the node further towards the core LD on behalf of the node. The NRs should wait with sending a response until they have received their responses to forwarded requests, to inform the registering node about success or failure of the registration process.

The advantage of recursive operation is that the node only needs to know or find a local NR. Additionally, the scheme minimizes message round-trips. In *recursive operation* however, the NRs need to perform registrations on behalf of other nodes. First, this implies that, the nodes cannot influence where they get registered on the path towards the core. Second, the NR must get authorized through some mechanism that it can register another node.

In *iterative operation*, a node that wants to be globally reachable registers its NID along a path towards the core LD one step at a time. It needs to send one request after the other to each involved NR and receives an appropriate response. Optionally, if there is no other lookup system for NRs available to find the right NRs, the response can include the (or a set of potential) next hop NR that needs to be contacted .

While the round-trip time for a full registration in *iterative operation* increases compared to the *recursive operation*, the *iterative operation* has different security properties and gives much more control of the path creation process to the end nodes - a node can directly decide at which NRs it wants to get registered if there are multiple available. As the node registers itself, it can directly use its own key material to register at NRs. If the security credentials of the registering nodes are to be used for the registration process in general (e.g. within organizations), then the *iterative operation* does not need an additional authorization mechanism to enable the NRs to carry out subsequent registrations on behalf of the registering nodes.

Since the registration defines the path from the core to this node to a certain degree, the *recursive operation* requires the network to make routing decisions on behalf of the end-system. On the other hand, the *iterative operation* allows nodes to influence the path to its needs. In both cases, certain capabilities of the path can be taken into account.

In *mixed operation*, the benefits of *recursive* and *iterative operation* can be weighed against each other according to the respective environment. A node may for example use *iterative operation* for the first few hops, but then the last hops could be done in *recursive operation*.

The operation mode of the registration could also be influenced by the underlying business models. As an example, a customer network might want to have explicit control on which path to register to its provider, but the provider might want to have control on how to further register the customer globally.

3.3 Usage of the Host Identity Protocol as Basis

There are several similarities between the Node Identity Architecture and The Host Identity Protocol (HIP) Architecture [4], including the separation of identifier and locator and the use of cryptographic identifiers as the basis for security. Also, since several implementations have been developed and are available HIP was a good starting point for a prototype implementation of our architecture.

The HIP Architecture decouples the dual use of IP addresses as locators and identifiers by introducing a new Host Identity namespace. Instead of using the IP address as end-point identifier each HIP-enabled host has a Host Identity(HI) on which the transport layer operates. The Host Identities a self-generated cryptographic public key. It also has a 128-bit Host Identity Tag (HIT) representation. The HIP protocol is a key exchange protocol used to establish communications context and IPsec Security Associations between hosts. However, HIP works end-to-end, which requires several changes and extensions (mentioned below) in order to implement the NodeID architecture.

4. EXPERIMENTAL EVALUATION

We have implemented a small scale prototype of the NodeID internetworking architecture. The prototype was built in order to show how the more abstract concepts and guidelines in the architecture could be implemented, to force the details of the basic internetworking mechanisms to be ironed out, and to investigate the feasibility of the architecture on a small scale.

The prototype was built based on the hip4inter.net code base [3]. We have added internetworking functionality that consists of NID registration and NID routing to bridge between locator domains.

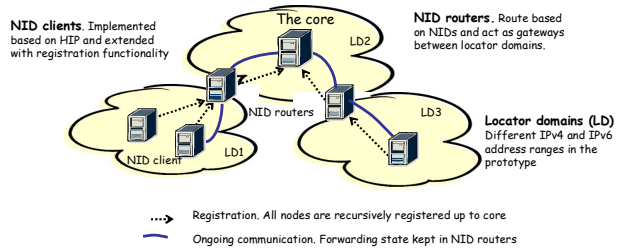

Figure 6: Prototyping Overview

As shown in Figure 6 the prototype consists of NID clients and NID routers. NID clients register to their local NID router to get reachable and the registration is done recursively up to the core. NID routers forward packets based on NIDs and act as gateways between locator domains. In the prototype we use the different IPv4 and IPv6 address ranges as locator domains. NID routers have a table of local NIDs that is populated when nodes register and they have a default route up towards the core.

In the prototype we implement the stateful version of the NodeID architecture where forwarding state is set up in all NID routers on the path between two communicating nodes. After the initial signaling, packets are forwarded based on Security Parameter Index (SPI)-labels and no NIFTs need to be included in the packets.

4.1 NID Registration Protocol

A node that bootstraps or enters a new locator domain has to register its NID with a local NR and every other NR along a default path up to the core until the registration reaches the node's home NR (HNR). In the prototype, we implemented the registration protocol in a recursive manner. A node registers its NID with the local NR and with that also orders the local NR to propagate the registration up the default path towards the HNR. The registration protocol needs two basic messages, the request for registration and a corresponding response. The prototype defines a new HIP control packet type *NID_UPDATE* to carry all NID-related parameters and reusing the HIP association. Afterwards it sends a registration request, which is recursively forwarded up to the home NID router. Once all responses have returned, the node is fully registered at all NRs up to the HNR.

4.2 NID Routing

In the prototype a NID router has tables of local NIDs that have registered and a default route up towards the core. It has a NID routing table that maps destination NID to next-hop NID and then a table that maps next-hop NID to a corresponding locator. This second step could have been done through a NID ARP-like mechanism as well (finding a router with a certain NID based on a locator), but by using HIP as the prototype basis the mapping from NID to locator is already managed by the HIP associations and can be re-used.

The routing hint is not implemented into the protocols, instead we use for the tests described below a single core LD router (NR2 in Figure 7). The algorithm on the NID router is still the same, just that in case, the NID is unknown and there is no default route (core NID router), the routing hint is used to find the next hop.

In the prototype, the NID routers set up and keep state for ongoing communication (following the stateful variant of the NodeID architecture). This can be seen as an optimization where the NIFTs only need to be included in the signaling packets, but not in the data packets. Instead, the Security Parameter Index (SPI) values are used as labels for forwarding packets in an end-to-end session. When two nodes want to communicate they first do a NID signaling exchange, similar to the HIP base exchange, that establishes the end-to-end security association. The four packets that form the HIP base exchange are forwarded in the NID routers by looking up the destination NIFT in the routing tables. In addition to looking up the destination NIFT, the NID routers on the path also look in the packets and learn the SPI values that are exchanged and use these to initialize the forwarding state. After the base exchange, no NIFTs are needed in the data packets. Instead the SPI values in the IPsec packets are used for forwarding. This forwarding functionality was implemented based on the existing SPINAT - Security Parameter Index (SPI) multiplexed Network Address Translation (NAT) [19]. The control code for SPINAT has been exchanged such that a NID client is addressing the NID router explicitly (know locator). Then the NID router uses the NID routing table for making the forwarding decision and it configures the decision what path the following packet follow into SPINAT.

4.3 Prototype performance

In order to verify that the prototype works as expected and to do a small scale evaluation of the architecture we measured the round-trip time and the control overhead in a simple experimental set-up as shown in Figure 7. Here we have two nodes A and B communicating with a varying number of NID routers in between. The NID routers are standard HP PCs equipped with 2 GHz Athlon CPUs and Broadcom BCM5750A1 Gigabit Ethernet adapters. The nodes A and B are standard DELL PCs with 2.8GHz Pentium 4 CPUs and Intel EtherExpress 82801EB/ER PRO/100 Ethernet network adapters.

Figure 8 shows the results of measuring the NID routing overhead once the forwarding state has been established and packets are forwarded based on the SPIs in the packets. Here we are sending minimal sized ICMP packets between the two nodes A

and B via a varying number of NID routers. When the number of hops is equal to one the nodes are talking directly to each other. With a number of hops equal to four there are three NID routers on the route between the two communicating hosts.

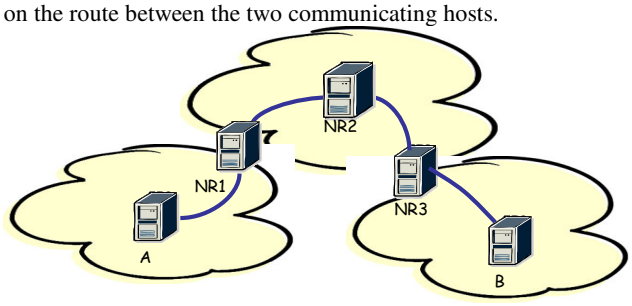

Figure 7: Experimental Setup

For reference we have included the point "ICMP 1 HOP" where the nodes are communicating directly without using HIP. As can be seen from the diagram (1 hop) HIP introduces an overhead of 0.1ms for a minimal sized packet. Each additional router contributes with an additional delay of roughly 0.25ms. As expected, the overhead grows linearly with the number of hops. All measurements were repeated 100 times and the standard deviations are shown in the figures.

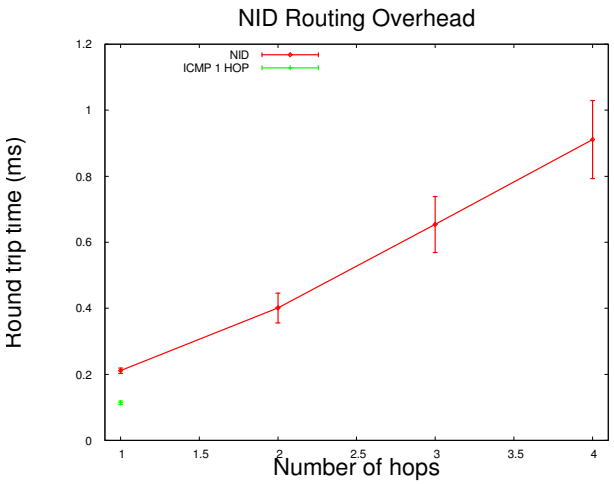

Figure 8: NID Routing Overhead

In Figure 9 we plot the same thing but for three different packet sizes, full sized Ethernet (1500 bytes), half sized Ethernet, and minimal. As can be seen from the differences in slope of the curves, the router overhead increases only marginally with full sized packets (0.1ms per router). This is to be expected since packets are not cryptographically processed by routers, only by endpoints. However, if we look at the 1 hop case we find that the cryptographic overhead goes up by 0.6ms when full sized packets are used (1 hop).

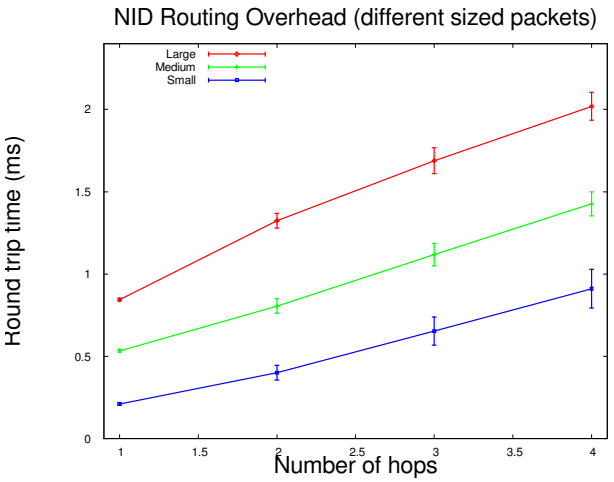

Figure 9: NID Routing Overhead for Different Packet Sizes

5. DISCUSSION

The previous sections have introduced the ingredients of the Node Identity Internetworking architecture. In this section, we discuss implications of various design choices and present some experiences gained while implementing the prototype.

5.1 Achieving the goals for a Future Internet Architecture

Bridging heterogeneous networks with middleboxes being first-order citizens

Those heterogeneous networks are represented as locator domains (LDs) in the internetworking framework. They can employ different networking technologies and are in that respect fairly autonomous. Global reachability is achieved with NID routers at LD borders, which inject routing information. These middle-boxes install state according to registrations they receive. In that respect they are not hidden from nodes as NAT-boxes are for example, introducing well-known problems. In other words, NID routers are first-order citizens that do not introduce such issues as introduced by NAT- and firewall-boxes. Note however, that after reachability is established, data plane traffic can be send without having to worry about or explicitly address these middleboxes.

Not introducing a large set of new components including managed name-spaces

The NodeID Internetworking Architecture uses existing technology when possible. For example, the technology used within an LD could be plain IPv4 or IPv6. The initial name resolution can be done using DNS after implementing appropriate resource records for NIDs and routing hints. Newly introduced is the NodeID namespace. It is however self-managed, since it is based on self-generated cryptographic material, which is statistically unique. Where there is need for, obviously the ID could be based on managed IDs, but the architecture does not mandate such management. There is only one entirely new type of component introduced using the NodeID architecture: the NID router.

Overcoming migration obstacles by allowing partial deployment while already providing sufficient benefits for early adopters

As the NodeID architecture does not mandate a global adoption of the routing functionality, partial deployment is feasible. As

has been mentioned, today's Internet could serve as the core network, requiring no change to today's Internet routing substrate. To facilitate incremental deployment, NID routers serving as home agents can also act as Mobile IP home agents. The benefits for early adopters are increased security and autonomy from other networks allowing realizing efficient multi-homing and supporting mobility for end-hosts and networks in an integrated fashion.

Always-on security including basic protection against denial of service attacks, encryption, authentication, and some form of location privacy

As node and network identities are based on cryptographic material basic privacy, authentication and encryption mechanisms can be implemented as an intrinsic part of the internetworking functionality. With the registration process in place, and additional mechanisms such as requiring end nodes to solve computationally expensive puzzles during registration, DoS attacks can be mitigated. In addition, as locators lose their global meaning a form of location privacy is implemented. The routing hint, serving to some degree as an indirection point further strengthens the location privacy property.

Native support for multi-homing, node and network mobility

Using a routing hint, the NodeID architecture has a built-in mobility anchor that removes the need for an add-on global mobility mechanism. Multi-homing is realized using NodeIDs (node multi-homing) and routing hints (network multi-homing) instead of locators, making it possible to implement multi-homing much more efficiently compared to the mechanisms at hand today.

Scalability

Unstructured NIDs or NIFTs might have scalability problems for global networks, or at least add costs to core network equipment in terms of routing table size and lookup speed. [16] shows results for a similar architecture doing routing based on NIDs only. Then work models the today's Internet with a core locator domain and 5 levels in the trees hanging off the core, and the current existing amount of nodes. In that setting, the full knowledge of all nodes in the core LD is needed, and we could expect roughly 20 million lookups per second (based on today's traffic characteristics and estimations). Therefore the routing hint was introduced, which allows distributing the core network state. The architecture is flexible enough to tradeoff the cost for mobility across different trees and the level of aggregation, depending on the specifics of the routing hint.

5.2 Experience with a HIP-based Instantiation of the Architecture

Using HIP as a building block gives the flat unstructured identity space and the separation of node identity and location prescribed in the architecture. HIP also provides the basic end-to-end security with cryptographic identifiers, host authentication and encrypted communication. Implementation-wise the HIP code base provides easy access to the NodeID prototype from all types of applications through a socket API. For example, the NodeID implementation has been used as a base networking platform for a peer-to-peer SIP prototype and for applications written using a SOAP platform.

Adding NID registration and NID routing to this as described in section 4 gives a mechanism for bridging between locator domains. In the following, we describe a few limitations and issues that arose while implementing one possible instantiation of the Node Identity Internetworking architecture.

A first limitation in the HIP-based prototype implementation compared to the architecture definition is that it does not allow neighboring locator domains to have overlapping address spaces. A NID router can only handle the attached locator domains if they have non-overlapping addresses spaces. However the same address space may be used in different locator domains if they are not directly connected to each other. To fully support the coexistence and cooperation of independent locator domains as described in the architecture more functionality is needed in a NID router to be able to differentiate IP packets that come from the same IP-address but from different locator domains through different network interfaces. Basically, a NID router would need to maintain multiple IP routing tables instead of a single one - one per connected locator domain. Note however that this is a general limitation present in most of today's operating systems and is not related to using HIP as a code base.

A second limitation with the current prototype is specific to using HIP for instantiating the architecture because HIP was designed as an end-to-end protocol. When a host moves or changes its locator in HIP it always signals the change to all its current peers, because HIP assumes an end-to-end, globally routable network layer underneath. However, in the Node Identity Internetworking architecture movements do not need necessarily need to be advertised to all peers. On the contrary, a locator has no meaning for a node in another locator domain, thus, it should not be advertised to any peer outside a node's own locator domain.

A technical consideration that came up during implementation concerns node registration. For a node to register at a NID router we defined a new HIP control packet type *NID_UPDATE* to carry all registration-related parameters. Previous work by Laganier et al. [18] already proposes a generic registration extension to HIP. Unfortunately, it is mainly designed to register a requester at a registrar and does not support *recursive operation*, i.e. the node sending the request must be the node to be registered. Therefore, the proposed extension was not suitable for the recursive NID registration. A recursive registration scheme should also be protected through some sort of control delegation, i.e. a node must authorize its NID routers to send registration requests on behalf of it. This can e.g. be done by providing authorization certificates to the NRs [17]. However, due to HIP packet size limitation, it is not possible to send the required certificates inside the HIP control packets and the mechanism is therefore not yet implemented.

Another technical issue with the current prototype implementation is the possibility of SPI collisions at NID routers. The forwarding at NID routers is based on the SPI values exchanged during the HIP base exchange as described in Section 4.2. Since SPI values are negotiated between the end-nodes only, two different communication pairs could choose the same SPI values to be used for their communication. Thus, if the communication path is overlapping, the NID routers would not be able to multiplex the communications based on the SPI values. One possible solution that was not further investigated during implementation would be to rewrite the SPI values at NID routers during the base exchange between the end-nodes – similar to the way NATs rewrite port numbers today.

6. RELATED WORK

The separation of identity and location is fundamental in the Node Identity Architecture and so also in many other proposed architectures including FARA [7], the Layered Naming Architecture [21] and DOA [22], the NAT-based architectures TRIAD [9] and IPNL [23], in TurfNet [27] and in the Split Naming/Forwarding Architecture (SNF) [24].

There are also several proposals in the IETF and IRTF that use the idea of locator/identity split. There are host-based proposals like HIP [4] and shim6 [26] and router-based solutions such as LISP [5] and Six/One [25].

The proposals differ for instance in how the identifiers are defined. HIP introduces unstructured cryptographic identifiers and is in this sense the work most similar to ours. The Layered Naming Architecture and DOA also propose the use of topology-independent endpoint identifiers from a flat namespace while in TRIAD and IPNL domain names (FQDNs) are used as identifiers.

The more incremental solutions Shim6, LISP and Six/One, don't fully separate the identifier and locator functions but use IP addresses (or parts of IP addresses) also as identifiers. LISP divides IP-addresses into endpoint identifiers and routing locators. A host is unaware of the latter which is used as transit address when tunneling between networks providers. The tunnels in LISP can in NodeID instead be implemented with NID routing and the routing hint. The "originally from/to" header option in Six/One can be compared to the NodeID routing hint, in that both provide routing information in the packet itself.

In the NodeID Architecture we also do forwarding and inter-domain routing on the NodeIDs to bridge between locator domains. Also, the NodeID Architecture provides improved security features based on cryptographic identifiers, which is not a focus of most of the works mentioned above.

Many registration and lookup mechanisms for flat name spaces have been proposed, mainly using DHTs. However, for inter-LD registration and lookup, DHTs may not be appropriate, because NodeID sets up inter-domain communication during the registration and lookup process. This can create additional dependencies between neighboring domains to use the same DHT type and protocol. On the other hand, DHTs can provide scalable and fault-tolerant intra-domain name resolution services. ROFL [14] proposes such a DHT-based inter-domain routing scheme based on a flat name space. However, it relies on a new, globally deployed internetwork protocol, whereas NodeID uses translation gateways between different local network protocols. Additionally, ROFL assumes consistency between the operation of the intra-domain and inter-domain routing schemes, whereas NodeID fully decouples intra- and inter-domain routing and forwarding schemes.

Traditional inter-domain routing mechanisms, such as BGP, operate on structured address spaces and static topologies. Because support for node and network mobility is a first-order requirement of NodeID, pre-establishment of routing paths is problematic; dynamic establishment of routing paths can have distinct advantages. In the area of geographic routing, various location service schemes have been proposed and classified [15]. NodeID is most similar to hierarchical schemes [13], except that the location service of a LD operates on non-geographic locators and defines routes during next hop inter-domain lookup. Finally, in the area of ad-hoc routing, many research proposals have concepts that are similar to NodeID. However, all of them focus on homogeneous networks that are much smaller than a global internetwork, and therefore they are typically only applicable at the edge of the network.

7. CONCLUSIONS AND FUTURE WORK

We have presented the NodeID architecture as a solution to some of the challenges faced by the Internet's infrastructure. The two-layer routing model, where the new second layer performs routing on flat node identifiers, relieves the current IP routing from dealing with node and network mobility and multi-homing, as well as basic security functions. With this separation, the IP layer can continue to focus on providing high performance packet forwarding. The more advanced functions are instead provided by the new node identity layer.

The two-level routing also enables Internetworking over multiple technologies, such as IPv4 and IPv6, and multiple addressing domains, for example private and public address spaces. Node identity routers take over the role NATs have today, making them an intrinsic part of the architecture. Work is also ongoing to support multiple core networks of different kinds, enabling seamless interoperation of the global IPv4 and IPv6 Internet.

The NodeID routing scheme based on default routing and registration described in this paper serves as a working example. The NodeID basic forwarding mechanism, including the routing hint, allows other, more advanced, routing protocols, just like the IP forwarding mechanism does. Protocols that can handle larger edge structures were for instance designed as part of the Ambient Networks project. Work was also done on replacing the current interdomain routing with a BGP-like protocol that is using locator domain identifiers instead of AS numbers.

The implementation of the prototype evaluated the feasibility through the proof of concept and showed good performance in a small network. Also it integrates well with various types of application and overlay networks running on top of the NodeID architecture. However, for evaluating the concepts in large to very large scale analytical work is required, specifically scaling the system up towards a network of information containing information elements in the range of trillions of elements accessible globally.

8. ACKNOWLEDGMENTS

Part of this work is a product of the *Ambient Networks* project supported in part by the European Commission under its *Sixth Framework Program*. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the *Ambient Networks* project or the European Commission.

9. References

[1] Bengt Ahlgren, Jari Arkko, Lars Eggert, and Jarno Rajahalme, "A Node Identity Internetworking Architecture". In Proceedings of the 9th IEEE Global Internet Symposium, Barcelona, Spain, April 28-29, 2006. In conjunction with IEEE Infocom 2006

[2] "Node Identity Internetworking Architecture", S. Schuetz, R. Winter, L. Burness, P. Eardley, B. Ahlgren. Internet Draft, draft-schuetz-nid-arch-00, Sept. 2007

[3] "HIP for inter.net Project", http://www.hip4inter.net/.

[4] “Host Identity Protocol (HIP) Architecture”, R. Moskowitz and P. Nikander, RFC 4423, May 2006.
[5] “Locator/ID Separation Protocol (LISP)”, D.Farinacci, V.Fuller and D.Oran, Internet-Draft, draft-farinacci-lisp-07, (work in progress) April 2008..
[6] D. Clark, J. Wroclawski, K. R. Sollins and R. Braden. Tussle in Cyberspace: Defining Tomorrow’s Internet. Proc. *ACM SIGCOMM*, Pittsburgh, PA, USA, August 19-23, 2002, pp. 347-356.
[7] D. Clark, R. Braden, A. Falk and V. Pingali. FARA: Reorganizing the Addressing Architecture. Proc. *ACM SIGCOMM Workshop on Future Directions in Network Architecture (FDNA)*, Karlsruhe, Germany, August 2003, pp. 313-321.
[8] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe and A. Warfield. Plutarch: An Argument for Network Pluralism. Proc. *ACM SIGCOMM Workshop on Future Directions in Network Architecture (FDNA)*, Karlsruhe, Germany, August 2003, pp. 258-266.
[9] D. R. Cheriton and M. Gritter. TRIAD: A Scalable Deployable NAT-based Internet Architecture. Stanford Computer Science Technical Report, January 2000.
[10] R. Braden, D. Clark, S. Shenker and J. Wroclawski. Developing A Next-Generation Internet Architecture. *Whitepaper* (http://www.isi.edu/newarch/DOCUMENTS/WhitePaper.ps), July 2000.
[11] P. Srisuresh and M. Holdrege. IP Network Address Translator (NAT) Terminology and Considerations. *RFC 2663*, August 1999.
[12] M. Holdrege and P. Srisuresh. Protocol Complications with the IP Network Address Translator. *RFC 3027*, January 2001.
[13] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger and R. Morris. A Scalable Location Service for Geographic Ad Hoc Routing. Proc. *ACM Mobicom*, August 2000.
[14] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, S. Shenker, ROFL: Routing on Flat Labels, To appear: Proc. *ACM SIGCOMM*, Pisa, Italy, 2006.
[15] I. Stojmenovic. Location Updates for Efficient Routing in Ad Hoc Wireless Networks. *Handbook of Wireless Networks and Mobile Computing*, Wiley, 2002.
[16] Jordi Pujol, Stefan Schmid, Lars Eggert and Marcus Brunner, Scalability Analysis of the TurfNet Internetworking Architecture, proceedings of IEEE Globecom 2007.
[17] P. Nikander, J. Arkko, *Delegation of Signalling Rights*, in Proc. of Security Protocols Workshop, Cambridge, UK, April 2002
[18] J. Laganier, T. Koponen, L. Eggert, *Host Identity Protocol (HIP) Registration Extension,* RFC5203, April 2008
[19] J. Ylitalo, P. Salmela, H. Tschofenig, *SPINAT: Integrating IPsec into Overlay Routing*, in Proc. of the First International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm'05), Athens, Greece, September 5-9, 2005
[20] A. Eriksson and B. Ohlman, *Dynamic Internetworking Based on Late Locator Construction*, In 10th IEEE Global Internet Symposium, Anchorage, Alaska, USA, May 11, 2007.
[21] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica and M. Walfish, ”A Layered Naming Architecture for the Internet”, In Proceedings of ACM SIGCOMM, Portland, USA, 2004.
[22] M. Walfish, J. Stribling, M.Krohn, H. Balakrishnan, R. Morris, and S. Shenker, “Middleboxes No Longer Considered Harmful”, In Proceedings of the OSDI, 2004.
[23] P. Francis and R. Gummadi, “IPNL: a NAT-extended Internet Architecture”, In Proceedings of ACM SIGCOMM, San Diego, USA, 2001.
[24] A. Jonsson, M. Folke, and B. Ahlgren, “The Split Naming/Forwarding Network Architecture,” In Proceedings of Swedish National Computer Networking Workshop (SNCNW), September, 2003.
[25] “Six/One: A Solution for Routing and Addressing in IPv6”, Internet-draft, draft-vogt-rrg-six-one-01.txt, November 2007.
[26] "Level 3 multihoming shim protocol for IPv6", E. Nordmark and M. Bagnulo, Internet draft, draft-ietf-shim6-proto-10.txt, February 2008.
[27] S. Schmid, L. Eggert, M. Brunner, and J. Quittek. “Towards Autonomous Network Domains,” In Proceedings of IEEE Global Internet Symposium, Miami, USA, 2005.