A Framework for Mobility and Flat Addressing in Heterogeneous Domains

Walter Wong[1], Rafael Pasquini[1], Rodolfo Villaça[1], Luciano B. de Paula[1], Fábio L. Verdi[1] and Maurício F. Magalhães[1]

[1]Departament of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
State University of Campinas (Unicamp)
C. P. 6.101 – 13.083-970 – Campinas – SP – Brazil

{wong,pasquini,villaca,luciano,verdi,mauricio}@dca.fee.unicamp.br

Abstract. *Support for new features like mobility, multi-homing, security and internetworking among heterogeneous network domains are challenging the current (ossified) Internet architecture. Basically, the service offered nowadays provides end-to-end communication using a global and hierarchical addressing space, i.e., a unique address (IP) for both localization and identification functions. A common view present in the current literature is that an homogeneous internetworking protocol as represented today by the IP protocol is no more convenient for supporting the introduction of new services. The today's requirements for mobility and security emphasize the need of a new internetworking architecture to permit the interconnection of independent domains. The main points to make a new approach for the Internet architecture viable are the separation of identity and localization, the creation of overlay networks for introducing new control plane functionalities and the adoption of new routing strategies for these overlay networks. This paper contributes in this direction by discussing a framework that instantiates a new interconnection architecture for autonomous and heterogeneous domains as proposed in the context of the Ambient Network Project [Ahlgren et al. 2006].*

1. Introduction

The Internet has been constantly updated to support the new requirements of applications and services. Normally, these updates are done through patches applied to the Internet infra-structure to support application specific requirements. The main problem with this approach is related to the increase of the Internet ossification due to the difficulty in implementing new services. The Network, as it is today, uses a single name space (IP) as locator and node identifier in a rigid hierarchical structure of domains. This unique address is used to global routeability, i.e., end-to-end addressing.

With the popularity of wireless networks, new concepts such as Personal Area Networks (PANs), Vehicular Area Networks (VANs) and Domain Mobility are becoming more and more common. These new features are challenging the current Internet since a novel architecture should be designed to support this new scenario. The present paper assumes that the interconnection of heterogeneous autonomous domains and the identity and localization splitting are key concepts for the future of the Internet. The architecture is based on a global identity space and does not require global addressing or a shared internetworking protocol.

The framework presented in this paper is based on the NODE ID Architecture [Ahlgren et al. 2006] proposed in the European Ambient Networks Project [Ambient Networks Project 2006]. The main functionalities offered by the NODE ID Architecture are the support to node and domain mobility, the decoupling of locators from identifiers, the support for network domains composition and the internetworking between heterogeneous network domains.

The paper presents the NODE ID Architecture characteristics and proposes an operational framework to instantiate this architecture. Presently, such issues include the support of legacy applications, a transparent integration with DNS, the interconnection of heterogeneous domains, the mobility inside one domain (changing only its layer 3 address) and the mobility across heterogeneous domains (changing its layer 3 address and also the layer 3 technology).

The remainder of this paper is organized as follows: Section 2 briefly presents the objectives of the Ambient Network Project and Section 3 discusses the related work. Section 4 introduces the NODE ID Architecture and Section 5 outlines our proposed framework towards the instantiation of the NODE ID Architecture. Section 6 describes the implemented prototype and the results obtained. The final section of this paper brings the conclusions and traces the next steps of our work.

2. Ambient Networks

The Ambient Network project addresses the creation of network solutions for mobile and wireless systems beyond 3G. The Project is based on three principles: 1) Ambient Networks build upon open connectivity and open networking functions; 2) Ambient Networks based on self-composition and self-management and 3) Ambient Network functions can be added to existing networks [Ahlgren et al. 2006].

The main concept defined by the Ambient Network project [Niebert et al.] is the ACS (Ambient Control Space), which manages the functionalities related to control and data transport through a set of interfaces for services and applications. Network entities interact with the control space through three different interfaces that are depicted in Figure 1. Higher-layer applications and services use the Ambient Service Interface (ASI) to access a subset of the control space functionality. This subset includes functions such as naming, location and context management, inter-domain management and traffic engineering. Connectivity resources interact with the control space through the Ambient Resource Interface (ARI), for example, to access multiradio resource management, mobility and trigger processing. Finally, the Ambient Network Interface (ANI) facilitates communication between the control spaces of different networks, creating the shared, common control space that enables the internetworking capabilities the Ambient Network project aims to achieve.

The internetworking capability for Ambient Networks aims at providing end-to-end connectivity between nodes across a dynamically changing federation of interconnected locator domains and nodes. These domains have independent addressing and routing systems meaning that internal nodes can communicate only by relying on internal

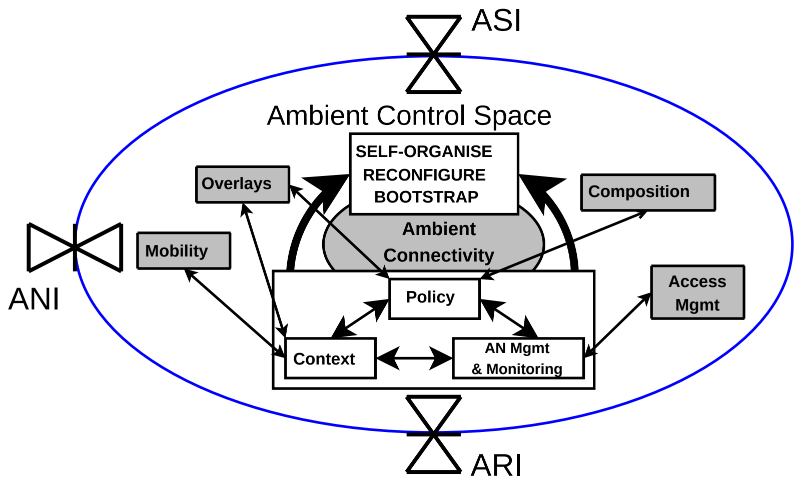

Figure 1. Ambient Network ACS modules and interfaces.

services not depending on the presence of outside internetworking mechanisms. As each domain may employ different networking technologies, they have to establish technology boundaries based on gateways. Some of the technologies demanded to support this federation of domains include cryptographic node identifiers, identity routers and localized addressing realms [Ahlgren et al. 2006].

3. Related Work

Recently, the research networking community has agreed that autonomous domains and decoupling the locator from the identifier using a totally new flat address space are key concepts to the future of the Internet. The separation of locators from identifiers has the seminal effort with PIP (Paul's IP Protocol) [Francis 1994]. PIP is a protocol developed in early 90's intended to replace the IP and introduce the forwarding directive concept that, as an effect, separates location from identifier. One of the new requirements that is challenging the today's Internet architecture is the support for mobility. Currently, the main proposal to support mobility in IP networks is the Mobile IP Protocol (MIP) [Solomon 1998] that provides transparent mobility support at the IP Layer by using a fixed IP address (called home-address) and another address while moving (called CoA - care-of-address). However, the MIP assumes an IP world, i.e., it does not have support to other protocols. At the same time, the results obtained with MIP are not as successful as desirable by a moving IP node.

The Host Identity Protocol Architecture [Nikander 2004] introduces a new namespace to fill the gap between domain naming and the IP addressing, the Host Identity namespace. This proposal separates the semantic overload of the IP address, offering mobility service by separating the transport associations from locators. Also, it offers strong security services, preventing from many security threats, such as denial of service, replay and man-in-the-middle attacks. Another proposal is the Internet Indirection Infrastructure [Stoica et al. 2002] which aims to use an indirection infrastructure to offer new services such as mobility, multicast, anycast and also preserve the privacy. IPNL (IP Next Layer) [Francis and Gummadi 2001] extends the forwarding directive concept to create a NAT-based solution for the lack of IP addresses. There is also a model called FARA

(Forwarding directive, Association, and Rendezvous Architecture) [Clark et al. 2003] that assumes the separation but does not define an architecture to support such feature.

A similar proposal to ours is the TurfNet [Schmid et al. 2004] which assumes the idea of network domains (called Turfs) organized in a hierarchical composable tree of domains attached to a core network. The traffic relaying between heterogeneous domains is done through gateways (in our framework called NID Routers) that are responsible for address and technology adaptation. One differential between our proposal and the TurfNet is that our proposal addresses the mobility scenarios inside one domain and between domains. At this moment, as mentioned in [Schmid et al. 2004], the TurfNet considers the inter-domain mobility scenario, but has not addressed it yet.

To conclude this section, the ROFL [Caesar et al. 2006] is a very interesting work that we consider important to our next steps due to its emphasis in routing directly over the flat namespace eliminating the hierarchical routing layer currently represented by the IP layer. Among other functionalities, the ROFL considers decoupling the locator from the identifier and Flat Routing through the usage of a hierarchical DHT (Distributed Hash Table). The ideas discussed by the ROFL are very prominent and are also being considered in our work.

4. NODE ID Basic Operation

Differently of many ad hoc proposals that have been developed recently in the Internet for solving only specific points (e.g. NAT, Firewall, etc.), the NODE ID Architecture permits to work across heterogeneous domains by employing cryptographic node identifiers, gateways (NID Routers), distributed hash tables (DHTs) and a control plane for integrating all of these functionalities [Ahlgren et al. 2006].

A basic requirement for future internetworking is the support of autonomous network domains. The architecture adopted as a reference to our framework enables internetworking among a set of autonomous and heterogeneous domains. The architecture adopts a global identity name space and does not require global addressing or a shared internetworking protocol. It permits to integrate the new concept of dynamic network composition with other recent architectural concepts, such as splitting locators from identifiers. The new host identity space lies between the host name and address spaces. Instead of mapping human-readable host names directly into network addresses, as in the Internet's Domain Name System (DNS), the NODE ID Architecture adopts two common name spaces, a shared name space and a common identity space, for enabling the inter-domain communication. They are the only globally agreed state, apart from a common inter-domain control interface. In this way, the NODE ID architecture name space maps into logical host identities. A second mapping translates host identities into host addresses that are suitable for network-layer data forwarding. The architecture manages the global name and identity spaces, whereas the address space is local to each individual autonomous network. This difference to the current Internet, which uses a global address space, allows using different addressing and routing mechanisms in individual autonomous networks.

The main assumptions underlying the NODE ID Architecture are: 1) independent Locator Domains (LDs) in which nodes are free to communicate independently of any outside internetworking mechanisms; 2) the connectivity between LDs is dynamic due to routing changes, multi-homing, node mobility or network mobility; 3) there is no distinction between hosts and routers which means that hosts at the network edge may relay traffic. The main characteristics of the NODE ID Architecture are:

- All nodes have a public key named Node Identifier (NID). The correspondent private key can be interpreted as the node identity. This approach decouples the node identification from its network location;
- The domains have gateways (NID Routers) for communication with other domains. The main role of the gateway consists in mapping and translating the different locator spaces and internetworking technologies belonging to each domain;
- Stable core(s) (not movable) with its NID Routers being part of a DHT to allow the discovering of each other's locators;
- Each LD has a DNS for FQDN (Fully Qualified Domain Name) resolution and registration.

The control functions in each domain encompass the traditional control plane activities in the network, such as address allocation, routing and name resolution. It also includes new functions for the architecture as, for example, management of domain composition. As each node in the architecture has a flat identifier generated after a hashing over its public key, the NID routers (NR) in Figure 2 forms an overlay network based on the NID namespace.

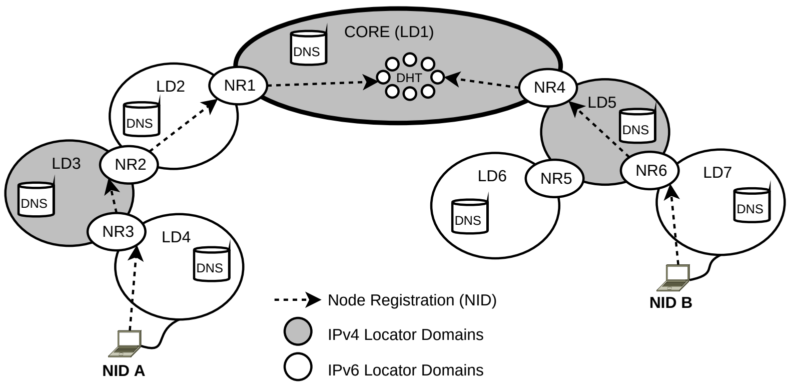

Figure 2. NODE ID Architecture example scenario.

The NID routers are special, multi-homed nodes. Since they are part of multiple domains at the same time, they need to relay traffic between these different domains. NID routers will also perform the required address and protocol translations. If two domains use the same protocols and have compatible addressing, the NID router will simply forward data packets - acting like a traditional Internet router.

The architecture assumes a hybrid approach for routing. It separates the routing into three parts:

1. The edge tree created by the edge nodes attached directly to the core or throughout other non-core domains;
2. The core routing;
3. Between the cores.

The routing in the edge tree employs a default routing towards the core and vice-versa for the opposite direction. This static route is obtained from the node registration process and eliminates the need of a global routing protocol [Ahlgren et al. 2006]. The lack of a global address space across all domains may prevent nodes belonging to different domains from communicating without prior registration. A node becomes reachable to other nodes by registering its local address with the host-identity-based lookup service. This registration propagates through the edge tree to achieve domain-external reachability. Domains always forward non-local registration messages to their vertically domain parents, resulting in a system where subsequent lookups are guaranteed to terminate at the root. For the core region, the Architecture has a DHT including only the NID routers directly connected to the core. The core DHT permits the mapping of the core NID routers into the respective core NID router locators in such a way that packet forwarding is done by using the underlying routing protocol. For the third region, corresponding to multiple cores, the global DHT indicates to which core the core NID router belongs. This information is used by the core NID routers to forward traffic to other core NID routers based on default routes to other cores and maintained in each core NID router.

5. Proposed Framework

The Proposed Framework encompasses functionalities that must be present in all nodes participating in the network. These functionalities constitute what we call *Internal Framework Modules*. Furthermore, there are other components that constitute the domain and the core infrastructure, explicitly positioned in some nodes, offering services to all nodes that are called *External Framework Entities*.

The framework being proposed is outlined in Figure 3 in which it is possible to verify the interaction between the Internal Modules and External Entities through the Ambient Service Interface (ASI), the interaction between the Internal Modules and the Network Layer Technology through the Ambient Resource Interface (ARI) and the presence of the Ambient Network Interface (ANI) responsible for providing the communication between Ambient Network nodes. Due to space limitation, the paper will focus basically on the Internal Modules. Consequently, we concisely describe the External Entities and the NID Header inserted in all packets in order to permit the packets to be routed through flat addresses.

5.1. Internal Framework Modules

In order to facilitate the comprehension of the internal modules they were organized in six functional groups as indicated in Figure 3. The next subsections describe these modules.

5.1.1. Capture of Legacy Packets

This first group is composed by the NID Filter module. It captures the legacy application packets based on the Linux iptables[1] rules. These rules are defined to intercept the outgoing packets using a NID as the destination address, and queues them using the ip_queue[2] kernel module. A packet is classified as a NID Packet if it matches a mask (*1.0.0.0/8* for IPv4, for example). If any packet matches this rule, it is queued in the ip_queue and the NID Filter receives it in the userspace. The main feature offered by this model is that

[1]Iptables is a Linux module that executes an action whenever a given condition, stated by a rule, is matched.

[2]Ip_queue is a Linux module that allows kernel space packets to be queued and processed in the userspace.

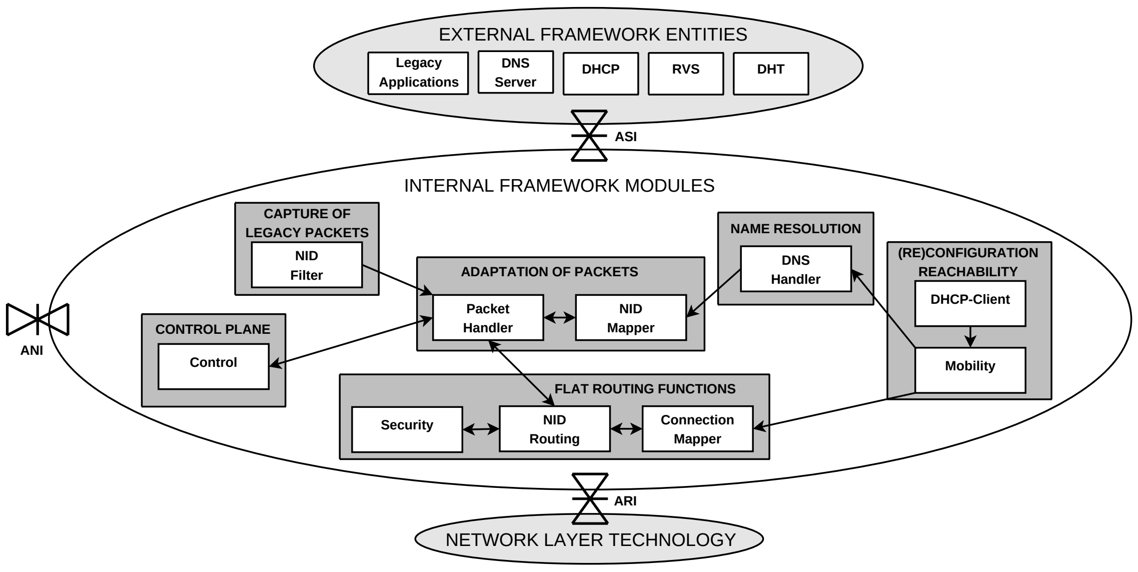

Figure 3. Interaction of the Proposed Framework Modules.

any legacy application would run over the new Flat Addressing structure without code modification.

5.1.2. Name Resolution

The second group is also formed by only one module, the DNS Handler module. It interacts with the NID Mapper module present in the Packet Adaptation group. Given a Fully Qualified Domain Name (FQDN), the DNS handler resolves it into a pair of NIDs: the destination NID and the NID of the corresponding domain router attached to the Core (called destination NID-R) which gives access to the destination node. In order to achieve that, the DNS Handler module acts as a proxy for the DNS Server. In other words, the DNS Handler uses UDP port number 53 on behalf of the DNS Server. In this way, the Legacy Applications uses the DNS Handler transparently.

An IPv4 legacy application tries to resolve the FQDN to an IP address making a query of type A to the DNS Handler. Firstly, the DNS Handler module tries to resolve the query searching in its own cache table and, if there is a correspondent association, the DNS Handler answers to the Client Application within a 32 bits hash from the NID Address that has 128 bits. On the other hand, if there is no association in its cache table, the DNS Handler module makes a TXT query to its domain DNS Server, which, in case of success, returns the pair <NID, NID-R>. According to the answer from the DNS Server, the DNS Handler inserts two new associations in the NID Mapper module: one association using the 32 bits hash generated from the destination NID and another association using the NID of 128 bits. Finally, it answers to the Client Application sending the 32 bits hash code. If the legacy application works with IPv6 addresses, it will make an AAAA query, and then, the return from the DNS Handler to the legacy application will be the NID of 128 bits.

5.1.3. Adaptation of Packets

The third group includes the Packet Handler and the NID Mapper modules. These modules are responsible for all the necessary actions involving the adaptation of incoming and outgoing packets.

The Packet Handler module offers services related to the insertion and removal of the NID Header in all NID Packets[3]. In order to achieve the creation of the NID Header, the Packet Handler firstly invokes the NID Mapper module service, which returns the destination NID-R based on the destination NID[4]. Once the Packet Handler receives the answer from the NID Mapper module, it will add the NID Header containing the four NIDs (Source NID, Destination NID, Source NID-R, Destination NID-R), and sends the NID Packet to the NID Routing module.

Another possibility is when an incoming packet addressed to the current node is passed to the Packet Handler by the NID Routing module. In this case, the Packet Handler module is responsible for removing the NID Header present in the packet and delivers it to the corresponding legacy application.

The NID Mapper module is responsible for managing two NIDs association tables. As mentioned before, there is a table indexed by the destination NID of 128 bits and there is another table indexed by the 32 bits hash code. The first table returns only the destination NID-R and the second table returns the original 128 bits destination NID and the destination NID-R.

5.1.4. Flat Routing Functions

The fourth group encompasses three modules: Security, NID Routing and Connection Mapper. The proposed framework supposes that the end-to-end communication employs some security approach that is out of the proposed framework. So, the Security module provides security for the framework against some attacks that may threaten the network through a domain specific security mechanism. This mechanism takes advantage of the cryptographic feature of the Node identifier (NID). The proposed framework has chosen the IPSec Protocol to implement the local security requirements for those domains based on the IP technology.

The NID Routing module performs routing based on the NID Header present in all packets. The behavior of this module depends on the node type*(common node, domain NID Router or core NID Router)*. For all these cases, the NID Routing module examines the NID Header to verify if the packet is addressed to the current node. In case of being true (destination node is reached), the NID Routing module delivers the NID packet to the Packet Handler module, which in turn, removes the NID Header and sends the packet to the local legacy application. If the packet is not addressed to the current node, its next action will be to look up the routing table to verify if there is any entry corresponding to the destination NID. If the destination NID is present in the current domain or down in the edge tree, the packet is sent directly to the node or to the next domain NID router towards the destination. On the other hand, if the destination NID is not present in the routing table, there are two different possibilities as explained below:

1. NID Router not attached to the core - In this case, the NID Routing default action is to send the packet to the next domain NID Router, up to the tree (default route), based on the destination NID. It is important to say that in this case the RVS (Ren-

[3]The NID Packet is the legacy application packet plus the NID Header presented in Section 5.3.

[4]This association is the result of the Name Resolution process performed by the DNS Handler module.

dezvous server [Moskowitz and Nikander 2006]) is the external entity responsible for the NID to Locator resolution.

2. NID Router attached to the core - In this case, the NID Routing default action is to send the packet to the next domain NID Router present in the core based on the destination NID-R field. In this case, the responsible external entity for the resolution of NIDs into Locators is the DHT.

The Connection Mapper module provides services related to the maintenance of the current NID to Locator associations, i.e., this module acts as a cache for the RVS or DHT aimed at saving queries to these external entities. Consequently, it has a timeout associated to each entry in its table in order to monitor when a peer node locator changes due to a mobility event inside the domain or to another domain.

5.1.5. (Re)Configuration/Reachability and Control Plane

The (Re)Configuration/Reachability group is responsible for the (Re)Configuration and Reachability due to mobility events and is composed by the DHCP-Client and Mobility modules.

The DHCP-Client module supports basic node information regarding the node ID architecture. The set of information is received from a DHCP server extended to offer the parameters[5] needed for our prototype implementation. The Mobility module provides support for node handover. In case of handover inside the same domain, the Mobility module is responsible for updating its locator in the local RVS.

The Control Plane group is constituted by the Control Module that is responsible for the horizontal communication through the Ambient Network Interface (ANI). The Control Module interacts with the Packet Handler in order to send and receive control messages.

5.2. External Framework Entities

An External Framework Entity can be a legacy application running transparently over the proposed framework, or Serves complementing the framework by offering services like Name Resolution Service (DNS), Node Configuration (DHCP) and Node Locator Resolution (RVS and DHT).

The Legacy Application module has an indirect relationship with the framework since they work over the framework without any code modification. Basically, the legacy applications packets enter the framework through the NID Filter module and leave the framework through the Packet Handler module.

The DNS Server does not need to be changed because the adaptations to the name resolution service are implemented by the DNS Handler module.

The role of the DHCP Server is to support the node reconfiguration due to mobility events. The Server is contacted by the DHCP-Client module in case of a mobility event and it returns the new configuration parameters to the nodes.

[5] 1) Node Locator (Layer 3 address); 2) NID of the domain router attached to the Core; 3) NID of its Domain Router; 4) Locator of its Domain Router; 5) NID of its domain RVS; 6) Locator of its domain RVS; 7) NID of its domain DNS-Server.

The RVS is responsible for translating the destination NID into the corresponding locator and keeping on the correct mapping between these naming spaces in case of mobility. All the nodes must periodically update their locators in the RVS. The RVS is also responsible for registering the nodes into the domain NID Router through the forwarding of its updated registration information.

Similar to the RVS, the DHT is also responsible for NID resolution. The main difference between these two external entities is that the RVS is present in all domains and the DHT is located only in the core. A NID Router attached to the core must route all NID Packets using their destination NID-R field. Performing this action means that the NID Router must make a query to the DHT which is responsible for maintaining the mapping between the NID of the NID Routers attached to the core and their respective locators.

5.3. NID Header

This section presents the NID Header (see Figure 4) included in all legacy packets in order to transform them into NID Packets. The main function of the header is to allow the framework to route packets from the source node to the destination node. As explained earlier, our framework is based on the NID Architecture, i.e., the routing actions are based on the Destination NID and on the Destination NID-R. As a result, it is necessary to carry four NIDs in the NID Header (Source NID, Destination NID, Source NID-R and Destination NID-R). There is another 8 bits field used to define the Type of the messages and a Reserved 24 bits field that can be further used.

TYPE (8 bits)	Reserved (24 bits)	Source NID (128 bits)	Destination NID (128 bits)	Source NID-R (128 bits)	Destination NID-R (128 bits)

Figure 4. NID Header format.

6. Framework Validation

In order to validate our proposal, a first version of the prototype was implemented using the Linux operating system, the C language and the QEMU open source processor emulator. The prototype includes the following modules: NID Filter, Packet Handler, NID Mapper, Connection Mapper, NID Routing, DNS Handler, DHCP-Client, Mobility and Control. For the external entities, the prototype implements the following components: a) the Rendezvous Server (RVS) for each local domain; b) the DNS server (BIND 9 - Berkeley Internet Name Domain) extended to return the NIDs of the destination node and the corresponding core NID Router and c) the DHCP Server. The Security module and the DHT module at the core will be implemented in a new prototype version.

Some modules were developed as dynamic libraries in order to permit them to dynamically change their roles inside the framework. It is possible, for example, that a node becomes a NID Router by dynamically loading the Routing Module during the composition between two domains. The prototype was implemented in the userspace since the main objective for this version was to validate the operational infra-structure defined for the framework. Future versions of the prototype may be implemented in the operating system kernel-space as a way to obtain better performance during the mobility events.

Figure 5 shows the testbed used to validate the prototype. Three domains, named Domain 1, Domain 2 and Domain 3 were created based on an edge tree-like hierarchy. Each domain has the NID Router and the DHCP Server in the same Linux PC machine whereas the RVS and the DNS Server are part of a QEMU virtual machine.

The testbed includes an application server and a client node within Domain 2. The last one has two wireline interfaces to permit the test of mobility events. The first interface connects to Domain 2 (home domain) and the second one plugs to Domain 3 (visited domain). The visited domain interface is activated only when the mobility inter-domain event occurs.

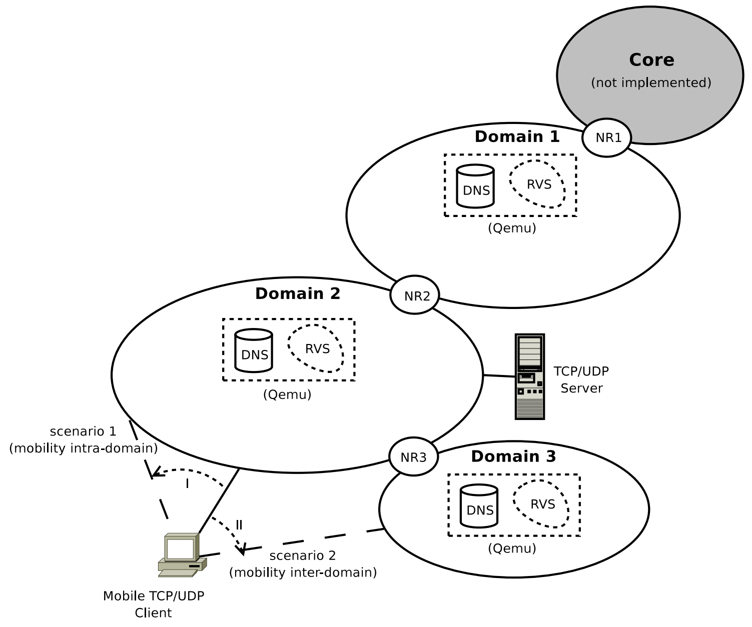

Figure 5. Framework validation scenario.

When the application server and the client start up in Domain 2, they send a DHCP_DISCOVER message and receive a DHCP_OFFER reply. So, they can send a DHCP_REQUEST message to the DHCP server in the Domain 2 to get the appropriate configuration parameters. After the (re)configuration process, the nodes must send a REGISTRY message to the Domain 2 RVS that propagates the registration to the domain NID Router (NR2 in this case). The NR2, in turn, must propagate the registration to the next NID Router upwards to the core (NR1 in the testbed).

The mobility tests were divided into two groups: 1) mobility inside the same domain and 2) mobility between two different domains. For the first scenario, the client application moves inside the Domain 2 and, to simulate its movement, the framework forces the node to invoke a new DHCP_REQUEST and obtain a new locator (IP address). For the second scenario, the movement is emulated by disabling the home domain interface and, in the sequence, the client node invokes a new DHCP_DISCOVER and DHCP_REQUEST to get the configuration parameters for the interface plugged at the Domain 3.

In a first test, just to evaluate the mobility scenarios, the movements were tested within the client application node while a SSH file transfer was in course. As expected, the connection was not dropped during the mobility and its flow resumed after the client reached its destination. The file transfer completed successful without errors for the intra-domain and the inter-domain scenarios. Other tests have been done forcing long delays for the mobility event. In all these tests, the TCP connection stayed open and after the handover conclusion the connection flow resumed without losses at the application level.

A second test, used to measure the performance of the prototype user-space implementation, the application server employed a TCP/UDP traffic generator to send packets to the client application node. The traffic generator was configured to send a CBR traffic at 75 Mbps during 30s, with 1400 Bytes in each packet. A previous test indicated that the rate of 75 Mbps was the maximum rate without packet losses using our framework without mobility events. This CBR traffic was unidirectional, so, only the client received packets generated by the server. The next steps describe the sequence applied for the tests:

1. Start a TCP/UDP flow from the server node addressed to the client node;
2. Start the client node mobility event after stabilizing the flow by 5 seconds;
3. Wait until the connection (or the flow) ends to collect the packet losses and the resume time for the mobility scenario.

The resume time was defined as the time between the last packet received by the client application before the mobility event and the first packet received by the client application after the mobility event. The TCPDUMP capture software runs in the client to generate a log containing the measured times.

The results for the resume time and for the packet loss are presented in Table 1 for the intra-domain mobility scenario and the Table 2 shows the results for the inter-domain mobility scenario.

Table 1. Resume Time and Packet Loss results for the intra-domain scenario.

Metric	Mean	Std. Dev.
UDP Resume Time	0,570s	0,156s
UDP Packet Loss	3.282 pkts. (1,89%)	911 pkts. (0,52%)
TCP Resume Time	0,369s	0,220s
TCP Packet Loss	64 pkts. (0,03%)	15 pkts. (0,007%)

Table 2. Resume Time and Packet Loss results for the inter-domain scenario.

Metric	Mean	Std. Dev.
UDP Resume Time	3,634s	1,449s
UDP Packet Loss	20.666 pkts. (12,13%)	7.820 pkts. (4,58%)
TCP Resume Time	6,47s	0,043s
TCP Packet Loss	57 pkts. (0,03%)	7 pkts. (0,004%)

In the case of the intra-domain scenario, the comparison between the resume time for the TCP and for the UDP traffics shows a lower resume time for the TCP. As the TCP connection has bidirectional control messages and due to the assumption of security communication inside each domain, the acknowledge messages sent by the client are also interpreted by the server side as redirect messages informing the new client locator. Comparing the resume time for TCP and UDP traffics in the inter-domain scenario, it shows that the TCP resume time is greater. It can be explained by the fact that the client acknowledge messages will not reach the server until the Connection Mapper times out.

The packet losses for the TCP connections in both scenarios are basically the same due to the TCP flow control.

As commented before, the Connection Mapper module acts as a cache for the <NID, locator> pairs resolved previously by the Rendezvous Server (RVS). For the obtained results presented in Tables 1 and 2, the Connection Mapper timeout was defined to 5 seconds, i.e., after this time an entry in this table is made invalid and a new RVS resolution for a NID to locator is necessary. For the UDP traffic this timeout permits to monitor when a peer node locator changes due to a mobility event inside the domain or to another domain. Some other tests for the UDP traffic were done by decreasing the timeout of the Connection Mapper to 3 seconds. The results showed that the resume time and the packet loss were reduced as presented in Table 3.

Table 3. Resume Time and Packet Loss for UDP traffic (CM = 3 seconds).

Metric	Intra-domain	Inter-domain
UDP Resume Time	0,405s	2,497s
UDP Packet Loss	2.249 pkts. (1,33%)	14.155 pkts. (8,29%)

7. Conclusion and Future Works

The main objective of this paper was to define an operational framework to instantiate the interconnection model, called NODE ID Architecture, proposed in the context of the Ambient Network Project. The main characteristic of the referred model is to permit the interconnection of heterogeneous mobile domains. The identifier and location splitting adopted by the NODE ID Architecture is the main approach to make viable this interconnection model. This approach has been adopted by several other proposals in the literature trying to offer new solutions for the future Internet architecture. A first version of the prototype of the operational framework was implemented and validated in the user space. The main results obtained with the prototype implementation were the maintenance of the connections during the mobility events and, also, the integration of the legacy applications without any code changing. The next steps will be concentrated in four main points: 1) a new implementation version of the prototype running in the operating system kernel-space; 2) the development of the Security Module of the architecture based on the cryptographic characteristics of the node identifiers (NIDs); 3) integration of the core DHT in the prototype; and 4) a new routing strategy based on the merging of DHTs in order to eliminate the static routing approach adopted by the NODE ID Architecture.

Acknowledgements

We would like to thank Ericsson Brazil, CAPES, CNPq, FAPESP for supporting this work. We also would like to thank Annikki Welin and Victor de Souza for their technical contributions.

References

Ahlgren, B., Arkko, J., Eggert, L., and Rajahalme, J. (2006). A Node Identity Internetworking Architecture. *In Proceedings of the 9th IEEE Global Internet Symposium realized in conjunction with IEEE INFOCOM, Barcelona, Spain, April 28-29, 2006.*

Ambient Networks Project (2006). Informations about this project can be found *online at:* `http://www.ambient-networks.org/`.

Caesar, M., Lakshminarayanan, K., Condie, T., Stoica, I., Kannan, J., and Shenker, S. (2006). ROFL: Routing on Flat Labels. *In Proceedings of the ACM SIGCOMM, Pisa, Italy, September 1-15, 2006*, pages 363 – 374.

Clark, D., Braden, R., Falk, A., and Pingali, V. (2003). FARA: Reorganizing the Addressing Architecture. *In Proceeding of the ACM SIGCOMM Workshops, Karlsruhe, Germany, August 25-29, 2003.*

Francis, P. (1994). Pip Near-term Architecture. *IETF RFC-1621 - Document available online at:* `http://www.ietf.org/`.

Francis, P. and Gummadi, R. (2001). IPNL: A NAT-Extended Internet Architecture. *In Proceeding of the ACM SIGCOMM, San Diega, CA, USA, August 27-31, 2001*, pages 69 – 80.

Moskowitz, R. and Nikander, P. (2006). Host Identity Protocol (HIP) Architecture. *IETF RFC-1621 - Document available online at:* `http://www.ietf.org/`.

Niebert, N., Flinck, H., Hancock, R., Kar, H., and Prehofer, C. Ambient networks - research for communication networks beyond 3g. *13th IST Mobile and Wireless Communications Summit, Lyon, France, June 27-30, 2004.*

Nikander, P. (2004). Applying Host Identity Protocol to the Internet Addressing Architecture. *In Proceedings of the International Symposium on Applications and the Internet SAINT'04, Tokyo, Japan, January 26-30, 2004.*

Schmid, S., Eggert, L., Brunner, M., and Quittek, J. (2004). TurfNet: An Architecture for Dinamically Composable Networks. *Workshop on Autonomic Communication (WAC), Berlin, Germany, October 18-19, 2004*, LNCS 3457:94 – 114.

Solomon, J. D. (1998). Mobile IP: The Internet Unplugged. *Prentice Hall, ISBN-10: 0138562466, ISBN-13: 9780138562465.*

Stoica, I., Atkins, D., Zhuang, S., Shenker, S., and Surana, S. (2002). Internet Indirection Infrastructure. *In Proceedings of the ACM SIGCOMM, Pittsburg, PA, USA, August 19-23, 2002.*