Mobility management in RINA networks: experimental validation of architectural properties

Edu Grasa*, Leo Bergesio*, Miquel Tarzan*, Diego Lopez†, Sven van der Meer‡, John Day§ and Lou Chitkushev§
*Internet Architecture and Services, Fundacio i2CAT, {eduard.grasa, leonardo.bergesio, miquel.tarzan}@i2cat.net
†Telefonica Investigation y Desarrollo S.A., diego.r.lopez@telefonica.com
‡Ericsson Network Management Labs., sven.van.der.meer@ericsson.com
§Computer Science Department, Metropolitan College, Boston University, {day, ltc}@bu.edu

***Abstract*—Mobility management is a challenging problem in current networks, typically requiring dedicated, specialised protocols that manage the lifetime of a series of tunnels that follow mobile hosts as they roam through the network. The fundamental issue that complicates the mobility management problem is the lack of a complete naming and addressing schema in the current Internet architecture. This paper analyses what properties such schema needs to have, and discusses how Internet mobility solutions are missing parts of it. Then it looks at RINA, a network architecture with a complete naming scheme. Theoretical analysis backed up by experimental validation of the main properties for mobility support shows that managing mobility in RINA networks not only is simpler and easier to scale compared to the Internet situation, but also that no special protocols or mechanisms need to be added to RINA in order to support mobility.**

I. Mobility management: a naming and addressing perspective

Mobility is one of the most consolidated trends in the ICT industry. A large portion of the traffic that is transported by our current networks is generated or consumed by applications running in mobile devices, therefore mobility support is a key aspect of modern network design. Network mobility management has traditionally been hard to implement at scale, usually requiring multiple specific solutions for different environments, involving the design, deployment and operation of specialised protocols. However, by revisiting the fundamentals of network architecture - specially naming and addressing - this paper proposes a simpler, general framework for managing mobility in any form of computer network.

What is the fundamental problem with mobility management? Applications in mobile devices must be able to keep sending and receiving data through the network, even if they keep changing their points of attachment to the network as they move (the usually so called service continuity requirement). The degradation in the quality of service (packet loss, delay) perceived by the application while the mobile device changes its physical point of attachment (the handover procedure) should also be minimized.

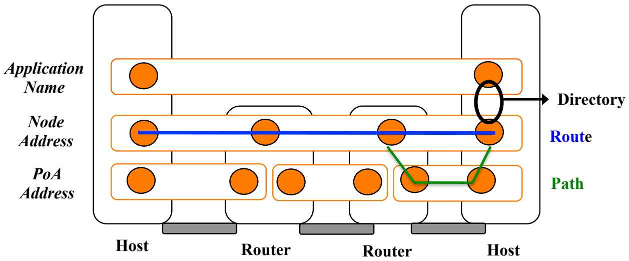

Fig. 1. Abstract view of entities to be named in a network

Figure 1 depicts an abstract view of the entities to be named in a network, according to Saltzer [1]. Applications communicate with each other via flows provided by nodes. Nodes execute the protocol machines to provide transport services and route application packets to its destination. Nodes can be connected to each other by one or multiple points of attachment, allowing one or multiple paths between nodes. Applications are assigned names that indicate what is communicating (application names identify the source/destination of communication flows). Since devices can move (or the same application can be moved to different servers), application names must remain stable and continue identifying the application endpoint regardless of the device's point of attachment; hence application names must be location-independent. Nodes must route application data through the network. In order for routing to scale to large network sizes, it is desirable that the name assigned to a node reflects its position in the graph of the network. This way multiple addresses can be aggregated into a single entry in a router's forwarding table. In other words, nodes are assigned addresses, which are location dependent names. Directories record bindings of application names to node addresses (the what is communicating to where it is). Finally points of attachment (PoA) provide a node with different paths to reach the next hop, indicating how to get there. If PoAs are point to point, PoA names just need to have local significance, otherwise they also require addresses.

Applying these naming and addressing principles to the problem of mobility management, it is possible to derive a number of properties that a general solution should incorporate:

- Address(es) that are used for routing or for points of attachment need to be able to change as the device moves. This way addresses of mobile devices can still be aggregated by the routing system and mobility does not increase the sizes of routing tables.

- A short enough time interval to detect a new point of attachment, acquire it and be able to transfer data on it, ideally before losing an old one; and
- The time to propagate routing information among the members within the layer should be small relative to the time interval between address changes. In other words, the layer should have sufficient time to reach a new stable operating state, and allow routing to converge in spite of mobility events.
- Mobility management should be done without the loss of traffic associated with running flows, or impairing the ability to create new ones.
- The applications communicating and the device should have names that are invariant with respect to changes in location in the network, so that they can be managed and remain accountable in spite of address changes.
- Solutions must scale to very large numbers of devices, sometimes moving quite fast.

The rest of the paper is structured as follows. In section II we review a number of mobility management procedures and compare its properties to the ones exposed in section I. In Section III we describe RINA and its mobility management framework, also assessing it against the properties exposed in section I. In section IV we describe a experimental validation scenario and present results and finally Section V provides concluding remarks and discusses future work.

II. Mobility management in the current Internet

Summing up the discussion of naming and addressing in section I, a good mobility management solution requires at least two sets of identifiers: i) application names that do not change as the mobile host moves, so that communication endpoints have stable identities regardless of their location; and ii) addresses that change as the host moves, to reflect the position of the device in the network and allow routing to scale. The fundamental problem of mobility management in the Internet is that the only identifier assigned to an entity in a Mobile Host (MH) that has scope greater than the system itself is the IP address. Hence there is a conflict: a single identifier cannot satisfy both properties at the same time.

There are no location-independent application names in the current Internet architecture. Communication endpoints are identified by the concatenation of the IP address and a local transport port. If the IP address changes the identity of the communication endpoint is lost and the communication flow breaks. Domain names are macros for IP addresses. Domain names are not used for routing, and do not satisfy the properties in section I as changes to the DNS name do not (and cannot in general) propagate to users of an IP address if the DNS mapping is changed to refer to a different IP address.

Consequently, the IP address can't change but the location of the device must; therefore the routing infrastructure must be cognisant of IP addresses that are mobile. If nothing was done, then every router that might see a MH address would have to have a separate routing table entry for that address, since it wouldn't be aggregateable once the MH leaves the home area where the address was originally assigned. In addition, the routing updates would have to be often enough and propagated through the entire IP routing infrastructure faster than the rate of change of points of attachment. This does not scale and several solutions have been proposed to mitigate this problem. Such mobility management solutions have been classified into two groups: i) solutions that work at the IP layer; ii) solutions that hide mobility from the IP layer.

Type II solutions manage mobility at lower layers between the mobile device and a mobile gateway, essentially creating a large IP subnet that looks like a fixed layer from the IP layer point of view. Cellular systems such as LTE (Long Term Evolution) [2], manage mobility by setting up and tearing down tunnels between the MH (called User Equipment in cellular terms) and mobile gateways that provide connectivity to the Internet or private IP networks. Tunnels have to be re-created every time the MH attaches to a different access point, incurring significant overhead. If the MH roams from a mobile network provider to another one, service continuity is lost (since the MH gets assigned an IP address from a different provider).

The analysis of this section focuses on type I solutions, which we will label IP mobility solutions. There are three primary approaches to doing IP Mobility: Mobile IPv4 (MIPv4) [3], Mobile IPv6 (MIPv6) [4], and Proxy Mobile IPv6 MIPv6) [5]. LISP [6], the Locator-Identifier Separation Protocol, has also been recently proposed as an IP mobility solution.

A. Mobile IP (v4 and v6), MIPv4,v6

The approach taken by MIPv4 [3] is to create IP tunnels to deliver packets to where the MH is attached to the network. MIPv4 designates the router the MH is nominally connected to as its Home Agent (HA). When the MH moves it attaches to a different router, its Foreign Agent (FA). Upon MH registration, the FA creates a tunnel back to the HA. Any packets sent to the MH will be intercepted by the HA and put into the tunnel, routed to the FA, which will deliver them on the interface the MH is connected. An optional optimisation in MIPv4 is to create a tunnel between the server where a MH is connected to and the FA, so that IP packets do not need to get routed through the HA. This option becomes mandatory in MIPv6 [4], in which tunnels terminate at the MH rather than the FA (this is the only difference with MIPv4).

B. Proxy Mobile IPv6, PMIPv6

A PMIPv6 domain hides the mobility of a MH from the rest of the Internet through a Local Mobility Anchor (LMA), which acts as the Internet gateway for MHs. MHs are attached to one of the multiple Mobile Access Gateways (MAGs) , which are usually the first hop routers for MHs. Tunnels are created between the LMA and MAGs, and updated as the MH moves and attaches to different MAGs. The advantage to Mobile IP is that rather than have tunnels potentially spanning the Internet, the tunnels are localized to the subnet that the LMA and MH are on at the cost of creating a single point of failure. Handover between LMAs is not supported.

C. LISP

LISP [6] has been proposed as a solution for mobility [7]. LISP is based on registering endpoint IP addresses at an overlay IP layer (the identifiers) to IP addresses of LISP border routers in an underlay layer (the locators). IP packets of the overlay are routed to a LISP border router, the destination IP address in the overlay (identifier) mapped to a destination IP address in the underlay (locator), encapsulated with a UDP/IP header and routed through the underlay until it reaches the destination border LISP router, where encapsulation is removed and the packet is routed through the overlay to the destination IP endpoint. By managing the overlay/underlay IP address mappings, it is possible to support mobility of hosts in the overlay layer, subject to the restrictions that the IP address in the overlay cannot change. A study of the cost of multi-homing and mobility of LISP, mobile IP and RINA is provided in [8], showing that the RINA scheme outperforms both solutions.

D. IETF Distributed Mobility Management (DMM) Working Group

DMM protocols being considered at the IETF aim at distributing mobile Internet traffic in an optimal way while not relaying on centrally deployed mobility anchors [9]. [10] discusses how to materialize the DMM goals in practice using three different approaches: i) using an evolution of PMIPv6 distributing the forwarding amongst multiple mobility anchors but keeping a centralized tunnel database; ii) using SDN (Software Defined Networking), via a central controller that configures the forwarding tables of all mobility anchor router; and iii) using a fully routed approach with BGP and no tunnels.

E. Analysis

Almost all solutions described in this section require the use of tunnels to handle mobility. Tunnels require the addition of dedicated forwarding table entries per mobile host, as well as a protocol to signal their creation, modification and destruction. Centralised mobility anchors as in PMIPv6 cause single points of failures and require significant resources at anchor nodes, since a lot of traffic has to be forwarded through them. LISP is not capable of dealing with mobility of hosts at the overlay IP layer, and incurs in larger overheads to deal with mobility than RINA-based networks do [8].

The one solution that does not require tunnels to deal with mobility (fully routed via BGP) has two main problems: i) since the scope of the network layer is global routing convergence is too slow; and ii) since the Mobile Host IP address does not change, routing overhead is high and contributes to further increases in the size of forwarding tables tracking mobile nodes.

The fundamental reason mobility management is so complex is the lack of a full addressing complement. There is nothing to hold on to but the IP address. An added problem with all of these proposals is that they just address mobility for specific use cases. They require considerable additional mechanisms and protocols, which require additional security mechanisms, and have no other benefits.

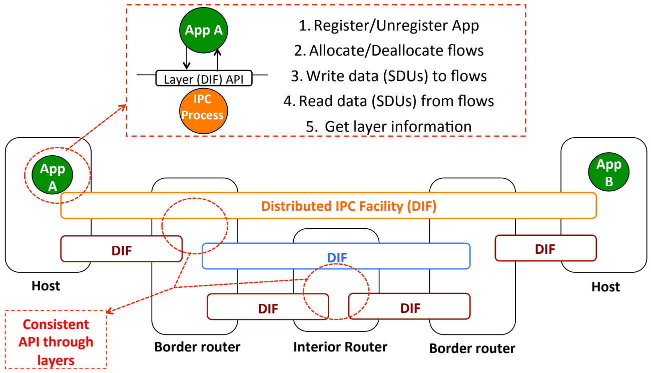
App A
Layer (DIF) API
IPC Process
1. Register/Unregister App
2. Allocate/Deallocate flows
3. Write data (SDUs) to flows
4. Read data (SDUs) from flows
5. Get layer information
App A
App B
Distributed IPC Facility (DIF)
DIF
DIF
DIF
DIF
DIF
Host
Host
Consistent API through layers
Border router
Interior Router
Border router

Fig. 2. Generic layers in RINA (down), and the common layer API for requesting IPC services (up)

III. Mobility management in RINA networks

RINA [11], the Recursive InterNetwork Architecture, is a fundamental network architecture that relies on the premise that networking is nothing more and nothing less than distributed Inter Process Communication (IPC). RINA decomposes networks into layers of generic protocols that can be configured via policies to optimally serve their operational requirements. As seen in Figure 2, in RINA there is a single type of layer - called a DIF, Distributed IPC Facility - that repeats as many times as needed by the network designer. A DIF itself is just a distributed application that performs and manages IPC. The application processes that are members of a DIF are called IPC Processes (IPCPs), and implement the network protocol state machines in a system for a given layer. A layer is a resource allocator, a container that allocates resources (memory, scheduling capacity, bandwidth) to provide communication services over a certain scope.

RINA adopts the naming and addressing structure depicted in Figure 1, generalising it to multiple layers. For any given layer N the IPCP is the node and gets assigned an address. IPCPs are application processes and have location-independent application names, which they use to register at lower layers (N-1). Therefore, for a layer N, higher layers (N+1) are applications which have names (regardless if these applications are also IPCPs or other types of applications), and the flows provided by lower layers (N-1) are the points of attachment that provide a path to the next hop IPCP at layer N. Mobility is nothing more than multihoming where the points of attachment change a bit more frequently. Multihoming in RINA does not require any special protocols, it is realised as a consequence of the naming structure [8]. The following paragraphs provide a short analysis of how mobility is managed in a RINA network.

Addresses locate the MH. Within a layer, each IPCP gets assigned an address that is location-dependent, structured to reflect the location of the IPCP with respect to the other IPCPs in its layer. If the MH which contains this IPCP moves too far (e.g. attaches to a base station in another subnetwork) the address will no longer be aggregateable causing an increase

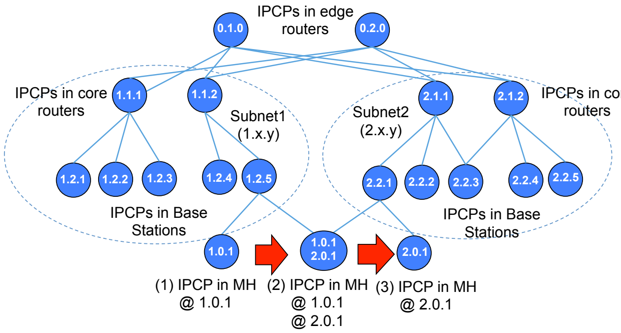

Fig. 3. IPCP in a MH doing a handover between base stations (BS) of different subnets. The MH IPCP gets a new address when attaches to the BS IPCP. For a brief period of time the MH IPCP is multihomed to both BS IPCPs.

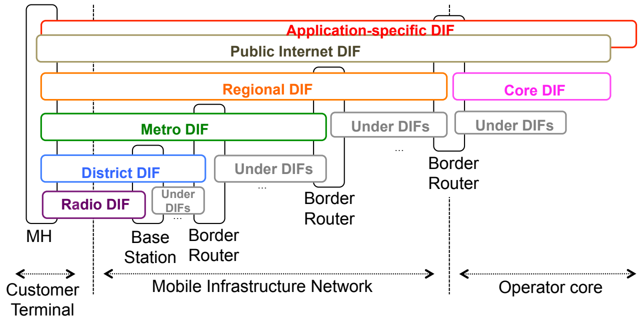

Fig. 4. The mobile network is partitioned into multiple layers of increasing scope, to bound the routing table size and mobility events seen by each layer

in router table size and potentially less efficient routing. This means that the IPCP address in the MH must change to keep router table size manageable and efficient. These address changes will not happen with every base station change, only when the MH enters a different subnetwork within the layer.

The procedure for updating addresses in RINA layers is discussed in [12], showing that it can be done in real time without impacting the flows provided by the layer. Hence service continuity and QoS are preserved. When an IPCP in a MH attaches to an IPCP of a new Base Station in a different subnetwork - as depicted in Figure 3 - it will get a new address and the old one will stop being advertised and disappear. The new address will be disseminated via the routing system, and directory updates will modify the required higher-layer application names to address bindings. All these procedures are already part of the architecture and do not require any additions to handle mobility scenarios.

Responsiveness to location change. Since address changes trigger routing and sometimes directory updates (only if the IPCP that changed addresses has server applications registered), these updates must be fast enough (with respect to the rate and number of handover events) to allow the layer to converge and reach a stable operational state. This puts a limit on the scope and size of the layer, which must be properly designed to operate effectively. This is a similar problem to the one experienced by the DMM fully routed solution, with the key difference that in RINA network designers can decide the number and scope of layers in the network.

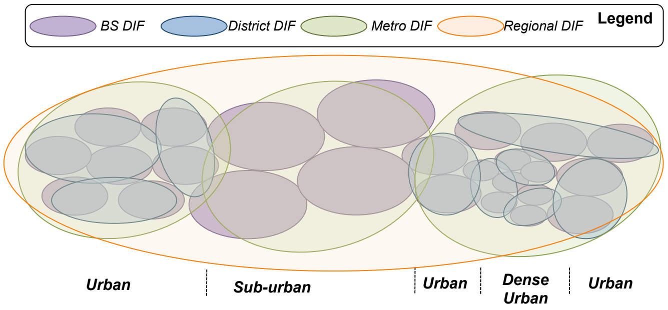

Fig. 5. Different parts of the mobile network can have a different number of layers, adapted to their requirements. The figure shows a view from above, in which the different rounded shapes are different DIFs, stacked on top of each other.

As seen in Figure 4, mobile networks can have multiple layers of increasing scope. Lower layers manage frequent mobility events for a lower number of mobile hosts, over a reduced scope (e.g. a neighbourhood in a city). Higher layers are only aware of handovers between access points belonging to different lower layers, therefore they will have to deal with less frequent mobility events and can have greater scope. This structure can be leveraged by the network designer to scale up the network and bound the size of routing tables as well as the rate of mobility events at each layer.

Moreover, the number of layers need not to be the same in the whole network. For example, urban areas with a high density of users will benefit from a higher number of layers than rural ones, as shown in Figure 5. Last but not least, the operator can dynamically create more or less layers in different regions of the network according to its needs and demand for network resources: e.g. it may create an ad-hoc structure of layers to cluster all the mobile hosts of people attending a concert, hiding their mobility events from other parts of the network.

Adding more layers increases the header overhead, but since i) every layer has the same protocols and ii) the presence and length of protocol header fields (e.g. addresses) can be customised for each layer, the overhead of adding a layer is minimum compared to today's situation in which all layers are different. Moreover, scaling up a single layer horizontally also introduces overhead (length of protocol header fields must be larger, protocol mechanisms become more complex) and makes it harder to scale (as shown by the DMM fully routed solution example).

Service continuity and QoS degradation. The same flows are in place through the lifetime of the application connection, mobility events do not disrupt existing flows since application names are stable and location independent. There are no tunnels to set up and tear down. The QoS experienced by applications is only degraded by the handover delay, packet loss can only occur if there was no physical communication with the network.

Manageability. The application names never change. The mapping of application name to (N)-address and of (N)-addresses to (N-1)-addresses is ensured by engineering the

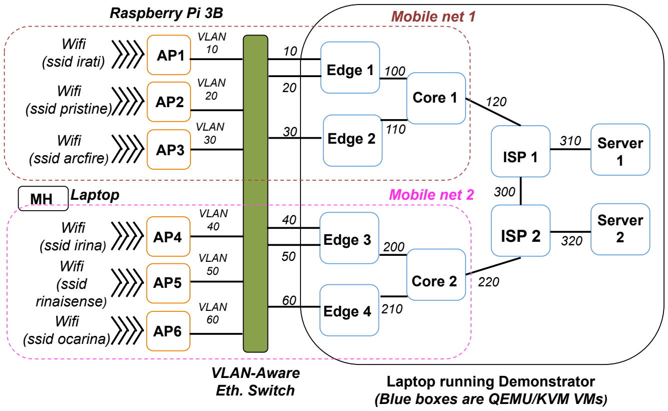

Fig. 6. Layout of the experimental scenario: physical systems

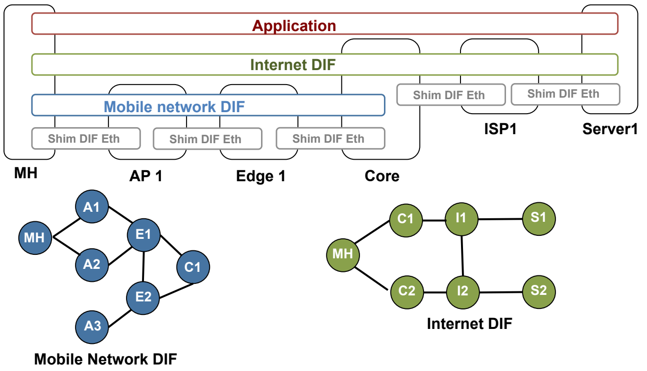

Fig. 7. Layout of the experimental scenario: Layers and connectivity graph of each layer

scope of the layers to guarantee that the update time is small compared to the rate of change of (N-1)-addresses. Mobility management does not require extra procedures that are not already present in other types of networks.

Scalability. Network designers are not limited to the use of a single layer, they can scale its network both horizontally (creating larger layers) and vertically (creating higher layers on top of each other). The number of layers in a mobile network becomes a matter of network design - not decided and frozen in standards - therefore it can be adapted and optimised dynamically while the network is running.

IV. Experimentation and validation

We have used the IRATI [13], [14] RINA implementation to experimentally validate the architectural properties of mobility management in a RINA network. IRATI is a programmable RINA implementation for Linux written in C and C++, with support for RINA over Ethernet, TCP and UDP. We have extended the IRATI codebase to support RINA over WiFi deployments, thus enabling experiments with mobility. Each WiFi interface is modelled as a shim IPCP that offers the DIF API (depicted in Figure 2) to upper layers. Calls to the DIF API are mapped to operations on the WiFi interface: reading and writing data, attaching, authenticating and detaching to WiFi access points. A detailed description of the implementation of WiFi support for IRATI is available at [15].

Figure 6 shows a detailed overview of the physical systems involved in the RINA mobility experiment. The deployment features 2 mobile networks - each one with 3 base stations, 2 edge routers and 1 core router. 6 Raspberry Pi Model 3B are used as base stations (WiFi Access Points), three for each mobile network. Each base station broadcasts a different SSID. Each mobile net is connected to a different ISP router, which in turn provide connectivity to two servers. All the edge, core and ISP routers, as well as the servers are deployed as QEMU-KVM virtual machines running within a laptop. Finally, a laptop with two WiFi interfaces acts as the Mobile Host (MH). All the physical and virtual systems are running the IRATI RINA implementation.

The different layers that are present in the experimental scenario are depicted in Figure 7. Each mobile network is composed by a *mobile network DIF* layer, which provides communication flows between the MH and the mobile network core router. Mobility through the operator's network is managed via this layer, which operates over point-to-point Ethernet links (between base stations and edge routers, and edge routers and the core router) and multi-access WiFi links (between the MH and the base stations). Each *mobile network DIF* layer is modelled as a single subnet that runs a link-state routing policy, therefore MHs do not need to obtain a new address when attaching to different access points. On top of both *mobile network DIF*, the mobile operators and the ISPs are operating a common layer that we have labelled *Internet DIF*. This layer allows applications running in the MH to communicate with applications running at the servers. The *Internet DIF* layer is only aware of mobility events across both operators; changes of access point within the same operator remain hidden to it.

During the experiment the MH joins a mobile network, which provides the MH access to the *mobile Internet DIF*. Two client applications at the MH that measure the round trip time allocate 20 flows each one to two server applications: one running at server one and another one running at server two (there are 40 communication flows in total). The MH moves across operator's 1 network, attaching to different base stations. At some point it attaches to a base station belonging to the operator's 2 network, joins it and continues to move through it. All flows are preserved during the whole experiment, and QoS is only degraded by increased delays and occasional packet loss during some handover events (packet loss seen in experiments is due to a not yet optimal integration of the IRATI code with the WiFi Linux toolchain, particularly with the WPA Supplicant daemon).

Figure 8 illustrates how the different instances of IPCPs at the MH behave during the experiment. At TO the MH joins the network of the first mobile operator. To do so, one of the WiFi interfaces joins the *irati* WiFi network and attaches to the base station labelled *AP1* in Figure 6. Then, the dark blue IPCP joins the *mobile network 1 DIF* - which requires authentication - and is assigned the *1.1* address (from the *mobile network 1*

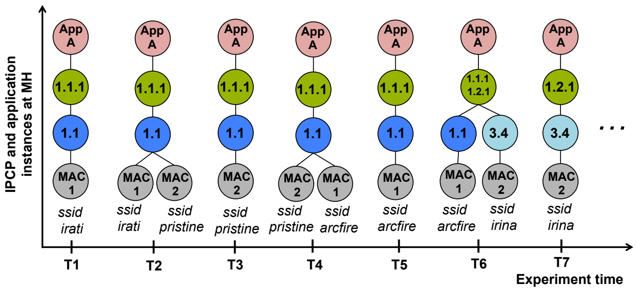

Fig. 8. Evolution of IPCPs and application instances at the Mobile Host, during the experiment time

DIF address space, entirely managed by operator 1). Then the MH requests a flow through the *mobile network 1 DIF* layer to join the *Internet DIF* layer. Once the flow is allocated, the green IPCP in the MH joins the *Internet DIF* - which in the experiment does not require authentication - and is assigned the *1.1.1* address (from the *Internet DIF* address space, managed by a coordinated effort of both mobile operators and the ISPs operating the ISP routers). After that Application A requests 20 flows to application B, which are allocated by the *Internet DIF* layer.

At T1 the MH enters in the range of *AP2* and its second WiFi interface joins the *pristine* WiFi network. Next, the dark blue IPCP requests a flow to the *mobile network 1 DIF* over the WiFi network, the flow is allocated and the dark blue IPCP becomes multi homed to both *AP1* and *AP2* (see bottom-left corner of Figure 7). At this point the MH can send and receive packets to/from both APs. The MH continues moving and at T3 it goes out of range of *AP1*. The dark blue IPCP detects it and deallocate the flow to *AP1*. At T4 and T5 the MH follows the same behaviour of T2 and T3.

At T6 the MH reaches the coverage are of *AP4*, and one of its WiFi interfaces joins the *irina* WiFi network - which belongs to the second mobile operator. The light blue IPCP at the MH allocates a flow to the *mobile network 2 DIF* layer, the flow is allocated and the light blue IPCP joins the *mobile network 2 DIF*, which requires authentication. Upon joining the light blue IPCP is assigned the address *3.4*, belonging to the *mobile network 2 DIF* address space. The green IPCP requests the *mobile network 2 DIF* a flow to the *Internet DIF* layer, which is allocated and then the green IPCP becomes multi-homed to both operator networks (see bottom-right corner of Figure 7). Then the green IPCP is assigned another address (*1.2.1*) that belongs to the subnetwork of the second core router. While the green IPCP is multi-homed, both addresses remain valid. At T7 the MH gets out of range of *AP3*, which causes the dark blue IPCP to deallocate the flow to *AP3* and the green IPCP to deallocate the flow to the *core1* router.

V. Conclusions and Future work

Mobility management is a simpler problem to solve with a complete naming and addressing architecture in place. Mobility in RINA doesn't require setting up tunnels, re-writing packet headers or using special protocols: it is just achieved by utilizing the tools the architecture provides and that are used for normal operation, albeit using them a bit more frequently. Mobility is supported by a combination of routing updates, changing addresses of IPCPs and designing the number and size of layers in different parts of the network to accommodate the load, scale, and rate of change of the (in this case) mobile terminals to be supported. But nothing more than what one would do to design a network for any other purpose. All standard procedures that can be performed in any RINA network, mobility is no special case.

We plan to extend the experiments in this paper to larger networks and multiple concurrent mobile hosts, using the experimental facilities available through the FED4FIRE+ testbed federation, in particular the w-iLab.t wireless testbed. Experiments will focus on analysing the mobility management overhead in terms of the number of routing and directory updates seen by the different networked systems when using different configurations of layers in a mobile network (from a single global layer to multiple layer depths).

Acknowledgment

This work is partly funded by the European Commission through the H2020 ARCFIRE project (Grant 687871).

References

[1] J. Saltzer, "On the naming and binding of network destinations," *Local Computer Networks*, 1982.

[2] R.-H. Liou, Y.-B. Lin, and S.-C. Tsai, "An investigation on lte mobility management," *IEEE Transactions on Mobile Computing*, vol. 12, no. 1, pp. 166–175, 2013.

[3] C. Perkins, "Ip mobility support for ipv4, revised," IETF RFC 5944, November 2010.

[4] J. A. C. Perkins, D. Johnson, "Mobility support in ipv6," IETF RFC 6275, July 2011.

[5] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, "Proxy mobile ipv6," IETF RFC 5213, August 2008.

[6] D. M. D. L. D. Farinacci, V. Fuller, "The locator/id separation protocol (lisp)," IETF RFC 6830, January 2013.

[7] U. C. D. Farinacci, P. Pillay-Esnault, "Lisp for the mobile network," draft-farinacci-lisp-mobile-network-00, August 2017.

[8] V. Ishakian, J. Akinwumi, F. Esposito, and I. Matta, "On supporting mobility and multihoming in recursive internet architectures," *Comput. Commun.*, vol. 35, no. 13, pp. 1561–1573, July July, 2012.

[9] J.Lee, J. Bonnin, P. Seite, and H. A. Chan, "Distributed ip mobility management from the perspective of the ietf: motivations, requirements, approaches, comparison, and challenges," *IEEE Wireless Communications*, vol. 20, no. 5, pp. 159–168, 2013.

[10] F. Giust, L. Cominardi, and C. Bernardos, "Distributed mobility management for future 5g networks: overview and analysis of existing approaches," *IEEE Communications Magazine*, vol. 53, no. 1, pp. 142–149, 2015.

[11] J. Day, I. Matta, and K. Mattar, ""Networking is IPC": A Guiding Principle to a Better Internet," in *Proceedings of the 2008 ACM CoNEXT Conference*, 2008.

[12] E. Grasa, L. Bergesio, M. Tarzan, D. Lopez, J. Day, and L. Chitkushev, "Seamless network renumbering in rina: Automate address changes without breaking flows!" European Conference on Networks and Communications, June 2017.

[13] S. Vrijders, D. Staessens, D. Colle, F. Salvestrini, E. Grasa, M. Tarzan, and L. Bergesio, "Prototyping the recursive internetwork architecture: The irati project approach," *IEEE Network*, vol. 28, no. 2, 2014.

[14] "Irati rina implementation source code," https://github.com/IRATI/stack.

[15] A. consortium, "D3.1 integrated software ready for experiments: Rina stack, management system and measurement framework," ARCFIRE deliverable D3.1, April 2017.