Rodríguez Fernández, D., Martínez Yelmo, I., Munthe-Kaas, E. & Plagemann, T. 2012, "ILORIN: Identifier-LOcator Resolution for Infrastructure-less Networks", in 2012 Third International Conference on the Network of the Future (NOF), Nov. 21-23 2012, Tunis, Tunisia.

Available at http://dx.doi.org/10.1109/NOF.2012.6463992

(Article begins on next page)

ILORIN: Identifier-LOcator Resolution for Infrastructure-less Networks

Daniel Rodríguez-Fernández
Department of Informatics
University of Oslo
Email:dani@ifi.uio.no

Isaias Martinez-Yelmo
Department of Automatics
University of Alcalá
Email:isaias.martinezy@uah.es

Ellen Munthe-Kaas and Thomas Plagemann
Department of Informatics
University of Oslo
Email:{ellenmk, plageman}@ifi.uio.no

***Abstract*—In order to overcome the limitations of the current Internet addressing, it is generally accepted that the Future Internet needs a separation between identifiers and network locators. Such identifier-locator split is also needed in infrastructure-less networks, such as Mobile ad hoc Networks (MANETs) and Delay Tolerant Networks, since they are an integral part of the Future Internet. Despite the amount of work in infrastructure-based networks, only a few proposals have considered how to apply this identifier-locator split in infrastructure-less networks. The contribution of this paper is an identifier-locator resolution system that can work in sparse MANETs, which are prone to network partitions. Our approach is an identifier-locator association discovery system, which uses periodic beacons to exchange the resolution information, avoiding the establishment of shared state between nodes. Our system exploits the broadcast nature of the wireless medium, opportunistic encounters, and information replication to disseminate identifier-locator associations across the network. The results of our extensive experiments demonstrate that our solution outperforms the related work, achieving a higher identifier-locator association discovery rate.**

***Index Terms*—Identifier-Locator split, Resolution system, Mobile Ad Hoc Network, Delay Tolerant Network.**

I. Introduction

Networking has substantially changed since the original design of the Internet. Concepts as mobility, multihoming and security, which were not taken into account then, are now fundamental requirements for the Future Internet. Furthermore, the Future Internet has to integrate increasing network heterogeneity. In this context, the double role of the IP address as both node *identifier* and *network locator* is a major constraint for the evolution of the Internet [1]. Therefore, many Future Internet proposals suggest some kind of separation between these two functionalities. This separation comes with the cost of a new resolution step from identifier to its associated network locators. In the current Internet, the identifier-locator resolution is implemented using highly scalable systems such as Distributed Hash Tables or hierarchical systems based on trust domains. In this paper, we define *network locator* as a native network address, which is used for routing purposes in a network, such as IPv4 addresses and IPv6 addresses are used in IPv4 and IPv6 respectively.

Infrastructure-less networks are used when it is not desirable or impossible to rely on the communication infrastructure. Application domains for these networks include, among others, rescue and emergency response and wireless sensor networks, which are all based on Mobile ad hoc Network (MANET) or Delay Tolerant Networking (DTN) [2] solutions. In order to effectively integrate infrastructure-less networks with the Future Internet, it is necessary to apply the identifier-locator split also in the infrastructure-less networks. These networks should be able to operate independently of the availability of communication infrastructure. Hence, an identifier-locator resolution system capable to work in infrastructure-less networks is required. However, despite the amount of proposals for identifier-locator resolution systems in infrastructure-based networks as the current Internet, just a few proposals consider the additional challenges that infrastructure-less networking presents. DTN architecture [3], HENNA [4] and Haggle [5] propose identifier-locator splits in infrastructure-less networks. Neighbour discovery protocols, such as NHDP [6] and IPND [7], can also be used to discover identifier-locator associations. However, they all propose encounter-based discovery mechanisms that are non-optimal and insufficient in the presence of network partitions.

This paper contributes with the design, implementation and evaluation of ILORIN (Identifier-LOcator Resolution for Infrastructure-less Networks). We have designed ILORIN to be simple and low-cost. It takes advantage of the broadcast nature of the wireless medium and opportunistic encounters. In contrast to the related work, our approach exploits also the replication of the identifier-locator associations. Each node has a local identifier-locator repository which stores all the resolution information discovered by the node. When a node gathers an identifier-locator association it becomes a server of this association. This fact makes our solution achieving a much better identifier-locator association discovery rate than the related work, as our results demonstrate. Furthermore, our proposal requires no changes in the routing protocols and is totally independent of them. Our approach avoids the establishment of a shared state between nodes. Each node relies just on its own state. We present a complete evaluation of how ILORIN performs under different conditions, ranging from highly sparse MANETs to well-connected MANETs. Even though we originally designed our system for sparse MANETs and DTNs, our results demonstrate that it performs well in both sparse and well-connected MANETs. Finally, ILORIN has been integrated with our previous work [8], [9] on heterogeneous networks.

The rest of the paper is structured as follows. Section II introduces our previous work. Section III describes the challenges that sparse MANETs present for identifier-locator resolution mechanisms. In Section IV, we detail the design of ILORIN. Section V describes our ns-3[1] ILORIN implementation. In Section VI, we present the evaluation of our proposal. Section VII discusses the related work. Finally, Section VIII concludes the paper and points out our future work.

II. Community Internetworking

In our previous work [8], [9], we describe Community Internetworking, which is a framework for internetworking highly heterogeneous networks. In order to cope with the potential heterogeneity of network locators in each networking substrate, i.e. in each networking domain, we propose the usage of a global cryptographic-based identifier namespace. This approach is in line with other approaches such as HIP [10], SpoVNet [11] or NodeID [12]. However, instead of a global resolution system, we propose the usage of different resolution systems specialised for each network substrate. In [8], [9], the Networking Substrate Adaptation Layer (NSAL) resolution primitives are described at the functional level. The implementation of these primitives is tailored to the particular NSAL instance that wraps a given networking substrate. The solution described in this paper is integrated with the infrastructure-less NSALs as described in Section V.

III. Identity-Location Split in Sparse MANETs

From a functional perspective, the two main functionalities provided by ILORIN are: (1) to *announce* an identifier-locator association, and (2) to *resolve* an identifier to a set of network locators. In this paper, we consider the use of cryptographic-based identifiers. These identifiers are the public part of a cryptographic public-private key pair. The size of such identifiers might vary, but they usually have a considerable size to obtain strong security. In our case, we use 2048 bit identifiers, which we consider secure enough for our purpose. Due to their size, it is common to use tags created by applying hash functions to the identifiers. This is the case of the HIT in HIP [10]. We use 128 bit tags, which we refer to as entity identifier tags (EIT) and which are generated in the same way as HITs in HIP. However, the use of EITs does not prevent us from the need of discovering the complete identifiers.

In this work we target sparse MANETs. The properties of a MANET are heavily influenced by the density of nodes, and in a sparse MANET there might be drastic variations in the node density. Some areas might have sufficient density of nodes to be well connected, while other areas might have low node density. Thus, network partitions can appear. Inside a partition there is enough connectivity to work as a classical MANET. In order to communicate between two MANET partitions, it is necessary to exploit opportunistic communication by using nodes that can act as ferries between the partitions. Therefore, the resolution system should be able to work independently of the node density and be able to cope with network disruptions.

[1] ns-3 is available at http://www.nsnam.org/

MANETs are usually composed of resource-constrained devices, which are battery driven. One important factor in battery consumption is network usage. Hence, we aim to optimise both the amount of messages transmitted and their size. We consider that the network usage is a good metric of the overall resource consumption in the nodes. The usefulness of the system can be measured as the probability of being able to resolve identifiers. Thus, the problem requires optimising two contradicting goals: (1) minimising the resources consumed by the resolution system, while (2) maximising the probability of being able to resolve identifiers.

In this paper, we aim to support OLSR and AODV routing protocols. These two routing protocols manage the effects of node mobility on the network topology at the routing level. Therefore, nodes do not need to change their network locators, i.e. their IP addresses. For this reason, we leave support for network locator updates as future work.

Finally, resolving identifiers to locators might have many security threats. Based on the cryptographic identifier associated with each entity, it is possible to include signatures to ensure that the information has really been generated by the entity that announced it originally. Security considerations are however out of the scope of this paper.

IV. ILORIN Design

Taking into consideration the characteristics and challenges of a sparse MANET, we have decided to design ILORIN as a discovery system. Nodes disseminate the identifier-locator associations that are locally known so that other nodes can gather these associations. Given that there might be network partitions and long delays, we give priority to the goal of low resource consumption. For this reason, we exchange all ILORIN messages through periodic beacons that carry the messages as payload. This approach consumes a very limited amount of resources and allows us to take advantage of the broadcast nature of the wireless medium. The downside is that the messages, such as association requests and responses, are delayed until a new beacon is sent. Nodes gather the information transmitted by the beacons from their 1-hop neighbours. Thus, the resolution information available locally increases. This is especially interesting in the presence of network partitions, because nodes become servers of identifier-locator associations gathered from other nodes. Hence, ILORIN exploits opportunistic encounters to disseminate resolution information between partitions. Another property of ILORIN is that it does not establish shared state between nodes. Each node maintains its own state independently, based on its local information and the information gathered from the beacons of other nodes. This property avoids problems related to shared state and session maintenance and makes ILORIN resilient against errors caused by lost beacons.

A. Architecture

We have divided the system in three main modules: Dissemination, Neighbourhood and Resolution. Figure 1 presents the architecture of ILORIN.

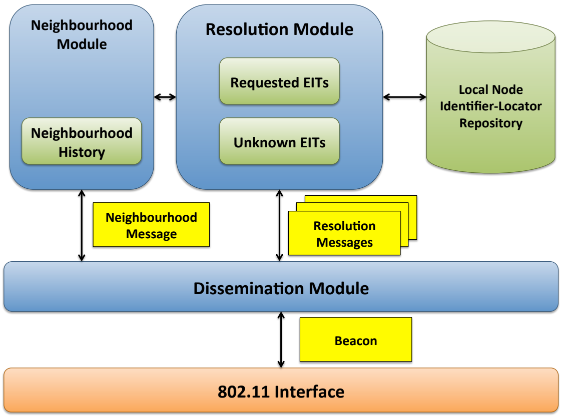


Fig. 1. ILORIN Architecture

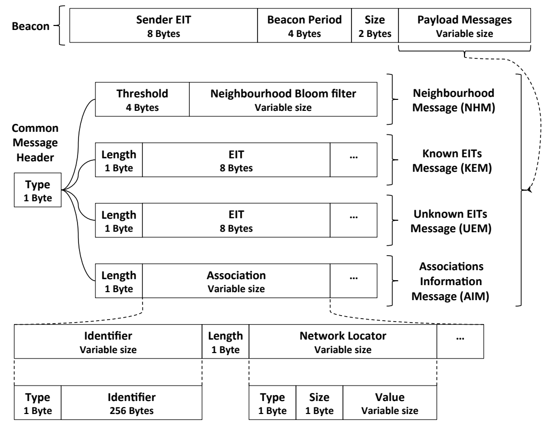


Fig. 2. Format of the beacons and their payload messages

1) Dissemination Module: This module sends periodic beacons which transport as payload a set of messages generated by the other modules. In this paper, we consider the messages generated by the Neighbourhood and Resolution modules, but the Dissemination Module might be used to send other types of messages as well. For instance, it can be used to announce services or inform about the resources present in the node. The beacons are sent as 802.11 broadcast messages, which are received by all nodes within transmission range. The frequency of the beacons (*BF*) is configurable. Preliminary studies described in Section VI indicate that two seconds is a good choice for *BF* in our evaluation scenarios. This module also receives beacons from other neighbours. The messages transported by the beacons are forwarded to the respective message handlers in other modules for processing.

2) Neighbourhood Module: This module keeps track of all nodes from which the node has received beacons. It offers methods to, for instance, check whether a node is member of the current 1-hop neighbourhood. In order to estimate the current 1-hop neighbourhood, it maintains a *Neighbourhood History* where it stores the timestamp of the last received beacon from each node. If the time since the last beacon is less than a configurable neighbour threshold (*NT*), then the node is considered a member of the current neighbourhood. Our preliminary tests indicate that the double of the neighbour's *BF* is a good *NT*. The *Neighbourhood History* stores for each neighbour its *BF*, its current 1-hop neighbourhood represented as a bloom filter, and the *NT* used to estimate its 1-hop neighbourhood. The Neighbourhood Module sends and receives *Neighbourhood* messages (*NHM*).

3) Resolution Module: Finally, this module is in charge of disseminating to and gathering from other nodes identifier-locator resolution information. For that purpose, it uses three different messages: (1) *Unknown EITs* (*UEM*), (2) *Known EITs* (*KEM*), and (3) *Associations information* (*AIM*). The first message requests the identifier-locator association for a set of EITs. The second message contains the set of EITs that are already known by the node. Finally, the third message contains a set of identifier-locator associations to give answer to requests received from other nodes. This last message includes the identifier and the network locators that can be used to communicate with it.

The Resolution Module manages two internal data structures: (1) *Requested EITs* and (2) *Unknown EITs*. The former contains a set of the EITs that have been requested by other nodes and the nodes that have requested them, while the latter contains a set of the EITs that are unknown. An EIT can be present in both data structures, because if a node eventually discovers the association related with that EIT, it already knows which nodes have requested the association. Additionally, the module has access to the local *Identifier-Locator Repository* of the node. This repository is shared with other components in the node.

B. Operation

The main operations managed by the system are the reception and transmission of beacons. Figure 2 presents the format of the beacons and messages.

1) Beacon Reception: The Dissemination Module receives beacons from nearby nodes. These beacons contain a set of messages that the Dissemination Module forwards to their respective handlers in other modules. On receiving a *NHM*, the Neighbour Module updates the entry of the beacon sender in the *Neighbourhood History* and the timestamp of the last received beacon from this node.

The Resolution Module processes *UEM*, *KEM*, and *AIM* messages. Algorithm 1 includes the pseudo-code for processing these messages and updating the data structures. The EITs contained in a *UEM* are iterated, adding to the *Requested EITs* a new request for the beacon sender. If the EIT is not in the node's local *Identifier-Locator Repository*, it is then added to the *Unknown EITs*. In case of the *KEM*, we iterate the EITs contained in the message, removing the requests made by the beacon sender concerning the EIT stored in the *Requested EITs*. Additionally, the EIT is added to the *Unknown EITs* if it is not stored in the local *Identifier-Locator Repository*. Finally, if the associations contained in the *AIM* are unknown, they are added to the node's local *Identifier-Locator Repository* and to the *newAssoc* list, which is used in the generation of *KEM*s. The association's identifier EIT is removed from the *Unknown Eits*, and the requests for this EIT made by

Algorithm 1 Processing Messages

```
1: procedure ProcessUem(hdr, msg)
2:     for each eit in msg do
3:         requestedEits.add(eit, hdr.src)
4:         if ¬(localRepository.find(eit)) then
5:             unknownEits.add(eit)
6:         end if
7:     end for
8: end procedure
9: procedure ProcessKem(hdr, msg)
10:     if ¬(localRepository.find(hdr.src)) then
11:         unknownEits.add(hdr.src)
12:     end if
13:     for each eit in msg do
14:         requestedEits.remove(eit, hdr.src)
15:         if ¬(localRepository.find(eit)) then
16:             unknownEits.add(eit)
17:         end if
18:     end for
19: end procedure
20: procedure ProcessAim(hdr, msg)
21:     for each assoc in msg do
22:         requestedEits.remove(assoc.eit, hdr.src)
23:         if ¬(localRepository.find(assoc.eit)) then
24:             unknownEits.remove(assoc.eit)
25:             localRepository.add(assoc)
26:             newAssoc.push(assoc.eit)
27:         else
28:             localRepository.update(assoc)
29:         end if
30:     end for
31:     nhModule.getNeighbour(hdr.src).lastAim ← msg
32: end procedure
```

the beacon sender are removed from the *Requested EITs*. In addition, as shown in line 31 of Algorithm 1, the message is stored in the Neighbourhood Module, because it is needed to check if an association should be included in the generation of *AIM*s, as described in Algorithm 3.

2) Beacon Transmission: When the Dissemination Module creates a new beacon, it indicates to the other modules the possibility of generating messages. The beacon size is limited by the MTU. Therefore, it is necessary to prioritise the information to be sent. The Neighbour Module creates a *NHM*, which is always added to the beacons. The remaining space is divided among the rest of messages, which in this paper are the messages generated by the Resolution Module. This generates for each beacon a *UEM*, a *KEM* and an *AIM*, which reflect its current state. We regard responses to request from other nodes as the most important task. Therefore, we prioritise the *AIM*, which can occupy 80% of the remaining size after the *NHM* is added. We prioritise also the *KEM*, since this can prevent other nodes from answering association requests that are already known. The *KEM* can consume 80% of the remaining space. These are two default values for space allocation in the beacon, which can be adjusted to allow other prioritization of message types. Finally the *UEM* can consume the space that is left after adding the other messages. Algorithm 2 includes the pseudo-code of the methods used to generate the messages. Given that messages have a limited space, we ensure that the data is eventually sent by iterating the data structures as circular lists. The *AIM* responds to the requests stored in the *Requested EITs*. Nodes respond just to EIT requests coming from nodes that their Neighbourhood Module considers current 1-hop neighbours. This prevents nodes from needlessly sending an

Algorithm 2 Generating Messages

```
1: procedure GenerateAim(max)
2:     msg.maxSize(max)
3:     if discoveredNewNeighbours() then
4:         assoc ← localRespository.localAssoc()
5:         if ¬(msg.hasSpace(assoc.size())) then
6:             return msg
7:         end if
8:         msg.add(assoc)
9:     end if
10:     it ← requestedEits.circularIterator()
11:     while it.hasNext() do
12:         request ← it.next()
13:         eit ← request.eit
14:         if localRepository.find(eit) ∧ needResponse(request) then
15:             assoc ← localRepository.get(eit)
16:             if ¬(msg.hasSpace(assoc.size())) then
17:                 it.previous()
18:                 return msg
19:             end if
20:             msg.add(assoc)
21:         end if
22:     end while
23:     return msg
24: end procedure
25: procedure GenerateKem(max, aimMsg)
26:     msg.maxSize(max)
27:     while ¬(newAssoc.empty()) ∧ msg.hasSpace(Eit.SIZE) do
28:         eit ← newAssoc.pop()
29:         if ¬(aimMsg.answer(eit)) then
30:             msg.add(eit)
31:         end if
32:     end while
33:     it ← localRepository.circularIterator()
34:     while it.hasNext() ∧ msg.hasSpace(Eit.SIZE) do
35:         eit ← it.next()
36:         if ¬(aimMsg.answers(eit)) then
37:             msg.add(eit)
38:         end if
39:     end while
40:     return msg
41: end procedure
42: procedure GenerateUem(max)
43:     msg.maxSize(max)
44:     it ← unknownEits.circularIterator()
45:     while it.hasNext() ∧ msg.hasSpace(Eit.SIZE) do
46:         msg.add(it.next())
47:     end while
48:     return msg
49: end procedure
```

association if the node that requested the association is no longer their 1-hop neighbour. We also consider that if a request from a node has been answered by a third node that is a common neighbour, there is a high chance that the requester has got a response. Therefore, the node considers that it is not necessary to respond to the request. These two optimisations are described in Algorithm 3. The *KEM* is created by iterating the EITs of the identifiers of the association stored in the local *Identifier-Locator Repository*. In this case, as shown in lines 27 to 32 of Algorithm 2, we give priority to EITs of the identifiers of the associations that have been recently discovered, so our neighbours will be informed that the node has received the association. To save message space, we check before adding an EIT to the *KEM* that the *AIM* that will be sent concurrently does not contain the association which identifier is related with this EIT. The *UEM* is created by iterating the *Unknown EITs*.

V. Implementation

We have implemented ILORIN as a ns-3 component. This component is integrated with the Community Internetwork-

Algorithm 3 Need Response algorithm

```
procedure NeedResponse(request)
    for each r in request.requesters do
        if nhModule.currentNeighbour(r) then
            if ¬(answered(request.eit, r)) then
                return true
            end if
        end if
    end for
    return false
end procedure
procedure Answered(eit, r)
    for each n in nhModule.currentNeighbourhood do
        if nhModule.currentNeighbours(n, r) then
            if n.lastAim.answers(eit) then
                return true
            end if
        end if
    end for
    return false
end procedure
```

ing framework described in [8], [9], in particular with the infrastructure-less NSALs. These are currently a MANET NSAL and a DTN NSAL. The MANET NSAL implementation can currently work with OLSR and AODV, and provides a classical IP end-to-end datagram service similar to UDP. The DTN NSAL provides in turn a delay tolerant hop-by-hop datagram service that can be used to intercommunicate different network partitions. The nexus between the NSALs and ILORIN is the NSAL local *Identifier-Locator Repository*. This contains the identifier-locator associations published locally in the NSAL and the associations that have been gathered from other nodes by the Resolution Module. The NSALs can also request specific EITs. These are added to the *Unknown EITs* if they are not present in the *Identifier-Locator Repository*. In the particular case of our MANET and DTN NSAL instances we use IPv4 addresses as network locators to be compatible with OLSR and AODV routing protocols. Our Community Internetwork implementation uses the generic *Address* class from ns-3 to represent the network locators. This allows the usage of different types of network locators in each NSAL. As shown in Figure 2, when a network locator is serialised, it includes as metadata its type and size. The same happens with the identifiers, which include as metadata their type. In the current implementation we consider 2048 bit Community Internetworking identifiers. This can easily be extended to allow other types of identifiers.

The dissemination module transmits and receives using directly the 802.11 protocol. The beacons are transmitted as 802.11 broadcast messages, which are neither acknowledged nor retransmitted if problems occur during the transmission.

VI. Evaluation

The two main goals of the evaluation are: (1) to study the probability of being able to resolve an identifier in varying network conditions, and (2) to analyse the resource consumption. We have performed preliminary studies to find acceptable values for the main parameters: the beacon frequency (*BF*) and the neighbour threshold (*NT*). We also compare, when possible, the results achieved by ILORIN and the related work.

TABLE I
Experiment parameters

Simulator	ns-3.12.1
Wifi Mode	802.11b/g
Physical Model	ns3::YansWifiPhy
Propagation Loss Model	ns3::FriisPropagationLossModel
Propagation Delay Model	ns3::ConstantSpeedPropagationDelayModel
Error Model	ns3::ErrorRateModel
Broadcast bit rate	1 $Mbps$ (DsssRate1Mbps)
Data bit rate	54 $Mbps$ (ErpOfdmRate54Mbps)
Random Way Point Model	ns3::RandomWalk2dMobilityModel
Initial Position	randomly placed inside the area
Speed	Uniform(0, 1.5) m/s
Area	Whole area
Distance	150 m
Beacon Frequency (*BF*)	2.0 s
Neighbour Threshold (*NT*)	4.0 s

A. Experiments design

In order to study ILORIN we have created two sets of simulation scenarios. In Scenario 1, we aim to study how ILORIN is affected by the node density in a MANET. For this purpose, node density is modified by increasing the number of nodes whilst maintaining a constant area of 2000 $m \times 2000$ m. We study eight different node densities from 10 to 80 nodes, increasing by 10 nodes in each step. To analyse the general behaviour of our solution in unpredictable scenarios, we use the random waypoint mobility model. Table I includes the parameters for the mobility model. Nodes speed resemble human walking speed (0 to 1.5 m/s).

Scenario 2 is motivated by rescue and emergency operations. Here, we study how ILORIN performs in a scenario where there are two partitions of 1000 $m \times 1000$ m with a distance of 1000 m from each other. We study eight different node densities from 10 to 80 nodes, increasing by 10 nodes in each step. One of these nodes acts as ferry between the partitions, simulating a vehicle. It moves from the centre of one partition to the centre of the other, and then returns to the initial partition. The ferry repeats continuously this movement, stopping in the centre of the partition for 10 s. This node has a uniform speed of 20 m/s (vehicle speed). The rest of the nodes are equally divided between the partitions and have the same mobility pattern as in Scenario 1.

In both scenarios, all nodes are equipped with an 802.11b/g interface set up in ad-hoc mode, the Internet stack, and our Community Internetworking framework including ILORIN. Table I includes the models and parameters used for the configuration of the 802.11 ns-3 model. For parameters that are not specified in the table we used their default values. The ns-3 reference documentation can be consulted for further details. The Community Internetworking framework generates a node identifier that is associated to the IP address (network locator) assigned to the node's wireless interface during the simulation configuration. The only variation in the software configuration is the routing protocol, where we have tested AODV and OLSR. Both protocols imply that locators, i.e. IP addresses, are not changed in the advent of mobility. The duration of the simulations is 300 seconds. In the experiments, nodes are not sending application traffic, but just the resolution system beacons and, in case of the OLSR configuration, OLSR control messages for maintaining the routing information. We leave as future work to study the behaviour of ILORIN in presence of application traffic. The MTU of the 802.11b/g is 2296 bytes, and broadcast messages are transmitted at 1 $Mbps$. For each configuration, we have performed 24 independent runs.

ILORIN's parameters *BF* and *NT* have a big impact on the overall resource consumption. After performing a preliminary study we have found that a *BF* of two seconds and a *NT* of four seconds is an acceptable choice for the studied scenarios.

We measure the number of packets and bytes (including the 802.11 header) transmitted and received at each node by ILORIN. The probability of being able to resolve identifiers is measured by the identifier-locator association discovery rate, called for short *discovery rate*. The comparison of ILORIN with the related work presented in Section VII is difficult due to their lack of information concerning both network consumption and discovery rate. The latter has been estimated by using a simplified version of ILORIN that we called Basic Neighbour Discovery (BND). This discovers the identifier-location association corresponding to a node the first time that it receives one of its beacons. This resembles the discovery mechanism used by NHDP, IPND and HENNA. The discovery rate for ILORIN and BND is measured using the same set of experiments to have a fair comparison.

B. Results

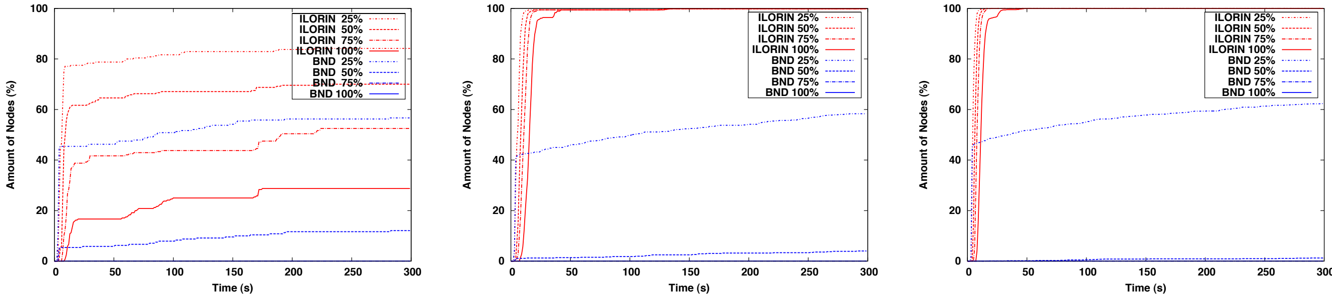
1) Scenario 1: Our experiments show that ILORIN performs equally well when using AODV and OLSR. This was expected since we are not introducing application traffic and ILORIN uses directly the Link Layer. Thus, we present just the results for AODV. Figures 3 and 4 show the performance of our system in terms of discovery rate and resource consumption respectively. We have selected the results for the configurations that we consider representative of highly sparse MANETs, sparse MANETs, and relatively well-connected MANETs.

Figure 3 presents the evolution over time of the amount of nodes that have achieved a certain discovery rate. This percentage is calculated as the average of the 24 runs. The figures present the discovery rates 25%, 50%, 75% and 100%. The red lines represent ILORIN results and the blue lines represent BND. For all configurations ILORIN achieved a much higher discovery rate than BND. For example, in Figure 3(a), after 300 s 28.75% of the nodes have a discovery rate of 100% using ILORIN, while none of them achieves this using BND. These differences in discovery rate increase with the node density, since ILORIN converges faster with higher node density. For the 30 and 50 node configurations, 95% of the nodes reach 100% of discovery rate in 22 and 17 s respectively. BND has a very poor performance anyway.

In order to measure the impart of mobility, we performed the same set of experiments increasing the maximum node speed to 20 m/s. Both ILORIN and BND obtain better results with higher mobility, since it is easier to meet other nodes. Nonetheless, ILORIN continue obtaining higher discovery rates faster than BND. For instance, for 10 nodes, all nodes achieve a 100% discovery rate after 16 s using ILORIN, while just 16.67% of the nodes obtain a discovery rate of 100% after 300 s using BND.

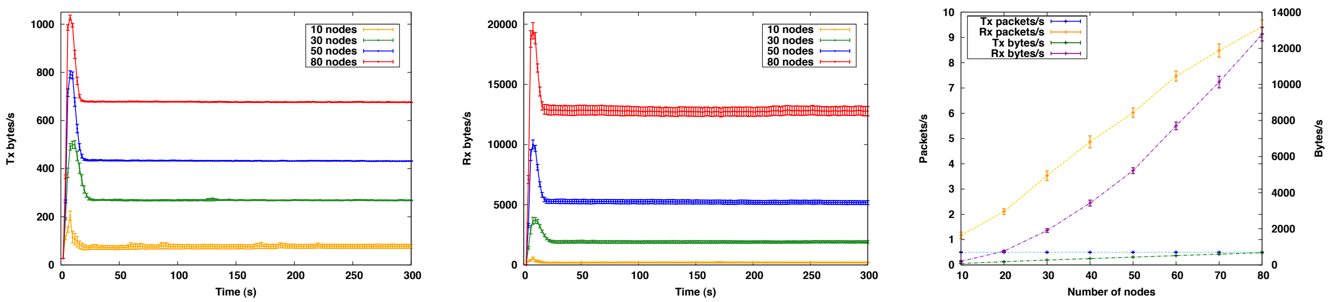
Figure 4 includes two different studies. Figures 4(a) and 4(b) show the evolution over time of the network usage at each node. Each point represents respectively the average transmitted (Tx) and received (Rx) bytes/s in a node during a beacon period (2.0 s), considering all nodes in the 24 runs of a given experiment. We have included in the figures the confidence intervals of 0.95. These two figures show that nodes receive much more bytes than they transmit, except for 10 nodes where they are similar. In the figures, we include results for 10, 30, 50, and 80 nodes. For all configurations both figures show an initial transient state where the amount of Tx and Rx bytes/s increases rapidly until reaching a peak and later on decreases and stabilises, entering a steady state. The transient state (peak) occurs when nodes start discovering many new identifier-locator associations. This happens in the start-up of the experiments, but also when information from other partitions comes available. An example of this is shown in Figures 5(b) and 5(c). When the ferry node arrives to a partition with new information it causes new peaks. The peaks in Tx and Rx bytes/s are caused by the response to associations requests for other nodes. When the steady state is reached, the messages size decrease, but they are still bigger than before the peak, since the number of known associations by the nodes increased. It is possible to see in both figures that the number of nodes has a big influence in the steady state, as it is also shown in Figure 4(c). Anyhow, the resource consumption in terms of Tx and Rx bytes/s are both low.

Figure 4(c) presents results when the steady state is reached. It shows the average network consumption per second in each node for all eight node densities. Each value is calculated as the average amount of resources (packets or bytes) transmitted or received in each node per second for the 24 runs, without taking into account the first 50 seconds, which we consider enough to remove the impact of the transient state. For the 80 nodes configuration, ILORIN transmits and receives in a node in steady state an average of just 676.98 and 12775.98 $Bytes/s$ respectively. This has an error of only a 2.83% and 2.91% respectively with a confidence interval of 0.95. The resource consumption adding both Tx and Rx bytes/s is equivalent to only 105.10 $Kbps$. Nevertheless, the network consumption decreases substantially for lower densities. What increases then is the confidence intervals since lower node densities are more exposed to the randomness of the node mobility. Figure 4(c) shows that the amount of Rx packets/s increases linearly with the density of nodes. This is easily explained by the fact that increasing node density also increases the number of neighbours a node has. The amount of Tx packets/s is constant, since they are only influenced by the *BF*. Concerning the network consumption in term of Tx bytes/s, the figure shows a linear increase with the node density. This is caused by the fact that the beacon size increases linearly with the number of known entities. However, this increase will stop and become constant when the MTU of the 802.11 protocol is reached. On the other hand, the



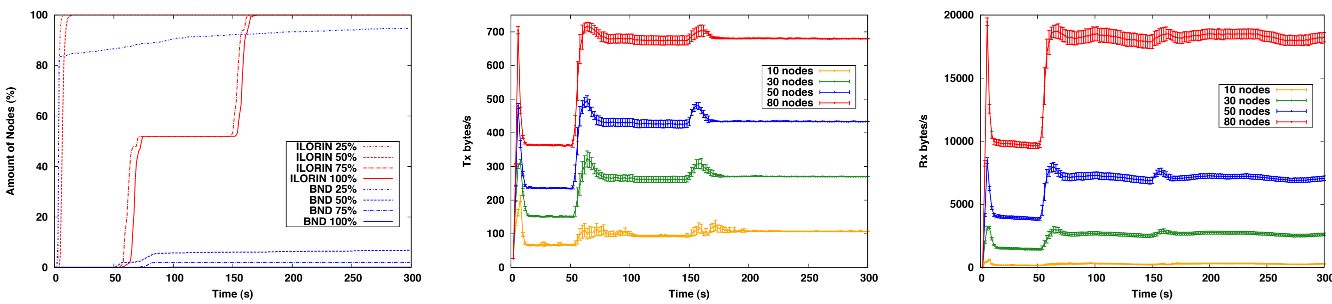
(a) Evolution of *Discovery rate* for 10 nodes (b) Evolution of *Discovery rate* for 30 nodes (c) Evolution of *Discovery rate* for 50 nodes

Fig. 3. Evolution of the *discovery rate* over time for 10, 30 and 50 nodes for the Scenario 1



(a) Evolution over time of Tx bytes/s per node (b) Evolution over time of Rx bytes/s per node (c) Average network usage in steady state

Fig. 4. Results for network usage for different densities of nodes for the Scenario 1



(a) Evolution of *Discovery rate* for 50 nodes (b) Evolution over time of Tx bytes/s per node (c) Evolution over time of Rx bytes/s per node

Fig. 5. Results for Scenario 2

amount of Rx bytes/s has a quadratic increase, since it is affected both by the increase in number of Rx packets/s and the increase in their size. This increase will nonetheless become linear when the beacons reach the MTU size, because then it will increase depending just on the number of neighbours. It is possible to reduce this resource consumption and improve ILORIN's scalability by reducing the *BF* or by limiting the beacon size. An adaptable selection of these parameters, taking into consideration the neighbour density and stability of the discovery, could optimise the overall resource consumption. We leave such adaptable selection as future work.

2) Scenario 2: Finally, Figure 5 presents results obtained for the experiment with two partitions connected by a ferry. Due to space limitations just a selection of the results is presented. Figure 5(a) shows the effect on the discovery rate when the ferry arrives to one partition around second 50 and its return to the other around second 150. ILORIN is able to achieve 100% of *discovery rate* in this DTN scenario, while BND has a very poor performance. Figures 5(b) and 5(c) present the evolution over time of the network consumption. Apart from the initial peak, two peaks around seconds 50 and 150 are caused by the ferry arrival to the partitions with new association information.

VII. Related Work

In this section, we describe proposals that apply an identifier-locator split in the presence of MANETs and DTNs. Due to space constraints we have not included related work in infrastructure-based networks, such as the current Internet. A more extensive collection can be found in [9].

NHDP [6] is a protocol for discovering 1-hop and 2-hop symmetric neighbourhoods for MANETs, which is based on the OLSR discovery process. However, in MANETs nodes are usually identified by their IP addresses as in the classical Internet. Hence, name/identifier-location resolution support has not been considered in such systems.

DTN IP neighbour discovery (IPND) [7] is a neighbour discovery mechanism for DTNs related to the DTN-BP architecture [3]. IPND is based on transmitting and receiving periodic beacons, which may include optionally the sender's canonical End Point Identifier (EID). It runs on top of IP so the IP address of the sender is included in the IP header, allowing an association between canonical EID and IP address. IPND only allows the discovery of identifier-locator associations of nodes that have been in range.

HENNA [4] presents an identifier-locator split proposal that works together with MEDEHA [13]. When HENNA operates in MANETs disconnected from the infrastructure, the nodes include their identifier-locator association in the neighbour discovery mechanism of MEDEHA, which is based on periodic beacons. Each node maintains a local mapping repository, which is restricted to nodes that have been encountered. This is problematic in DTN operation, because it is not be possible to resolve identifiers from nodes that have not been met already.

Haggle [5] is a data-centric networking architecture that aims to seamlessly integrate infrastructure based networking with infrastructure-less networks. In Haggle, the data is sent with associated metadata that can be used to identify its content. Haggle uses name graphs for identifying entities, e.g. users. Each node has associated a GUID and maintains a name knowledge base. When a node finds a new name, it requests to its peers information about it. This is similar to our proposal, but in our case we disseminate proactively the resolution information that has been gathered by the nodes.

VIII. Conclusions

The main contribution of this work is the design, implementation and evaluation of ILORIN, an identifier-locator resolution system for Infrastructure-less networks. Our solution exploits opportunistic encounters and replication to disseminate the identifier-location association information. This differentiates ILORIN from the related work, which only exploits encounters. Our solution is based on periodic beacons that carry as payload the resolution system messages. The beacons are sent as 802.11 broadcast messages. Additionally, the system does not require the establishment of shared state between nodes. We have implemented the system as a component in ns-3, integrating it with our Community Internetworking framework [8], [9]. However, ILORIN is flexible enough to be used by other systems.

The results of our extensive evaluation through simulation of different node densities and mobilities demonstrate that ILORIN outperforms the related work, estimated through the only encounter-based BND. ILORIN achieves higher identifier-locator association discovery rate than BND in both MANET and DTN scenarios. This increases the probability of being able to resolve the identifiers. The overall resource consumption is estimated by measuring the network consumption, in packets and bytes per second. The resource consumption caused by transmitting beacons is limited by the frequency of the beacons and the MTU. The reception of beacons depends also on the amount of neighbours, and therefore increases with node density. As described in Section VI resource consumption for both packets/s and bytes/s is low.

Finally, our future work aims to extend ILORIN with a dynamic selection of the system parameters in order to adapt to different MANET environments. We also plan to extend ILORIN to support network locator updates, and the resolution of names to identifiers. Additionally, we aim to use the dissemination system described in this paper to also disseminate service and resource information. We consider also the study the non-periodic beacon based discovery mechanisms.

Acknowledgements

This work has been funded by the VERDIKT Programme of the Norwegian Research Council through the DT-Stream project (project number 183312/S10) and the Regional Government of Madrid through the MEDIANET project (MEDIANET S2009/TIC-1468).

References

[1] D. Clark, R. Braden, A. Falk, and V. Pingali, "Fara: reorganizing the addressing architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 4, pp. 313–321, Aug. 2003.

[2] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, ser. SIGCOMM '03. ACM, 2003, pp. 27–34, ACM ID: 863960.

[3] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-tolerant networking architecture," RFC 4838, Internet Engineering Task Force, 2007.

[4] R. N. B. Rais, M. Abdelmoula, T. Turletti, and K. Obraczka, "Naming for heterogeneous networks prone to episodic connectivity," in *2011 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, Mar. 2011, pp. 1091–1096.

[5] J. Su, J. Scott, P. Hui, J. Crowcroft, E. De Lara, C. Diot, A. Goel, M. H. Lim, and E. Upton, "Haggle: seamless networking for mobile applications," in *Proceedings of the 9th international conference on Ubiquitous computing*, ser. UbiComp '07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 391–408.

[6] T. Clausen, C. Dearlove, and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)," RFC 6130, Internet Engineering Task Force, 2011.

[7] D. Ellard and D. Brown, "DTN IP neighbor discovery (IPND)," Mar. 2010.

[8] D. Rodríguez-Fernández, I. Martinez-Yelmo, E. Munthe-Kaas, and T. Plagemann, "Always best (dis-)connected: Challenges to interconnect highly heterogeneous networks," in *Wired/Wireless Internet Communications*, ser. Lecture Notes in Computer Science, X. Masip-Bruin, D. Verchere, V. Tsaoussidis, and M. Yannuzzi, Eds. Springer Berlin / Heidelberg, 2011, vol. 6649, pp. 410–421.

[9] D. Rodríguez-Fernández, I. Martinez-Yelmo, E. Munthe-Kaas, and T. Plagemann, "Always best (dis-)connected: Challenges to interconnect highly heterogeneous networks," *Journal of Internet Engineering*, vol. 5, no. 1, pp. 299–309, Jun. 2012.

[10] P. Nikander, A. Gurtov, and T. Henderson, "Host identity protocol (HIP): Connectivity, mobility, multi-homing, security, and privacy over IPv4 and IPv6 networks," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 2, pp. 186–204, 2010.

[11] R. Bless, C. Hübsch, S. Mies, and O. Waldhorst, "The underlay abstraction in the spontaneous virtual networks (SpoVNet) architecture," in *Next Generation Internet Networks, 2008. NGI 2008*, 2008, pp. 115–122.

[12] S. Schütz, H. Abrahamsson, B. Ahlgren, and M. Brunner, "Design and implementation of the node identity internetworking architecture," *Computer Networks*, vol. 54, no. 7, pp. 1142–1154, 2010.

[13] R. N. B. Rais, T. Turletti, and K. Obraczka, "Message delivery in heterogeneous networks prone to episodic connectivity," *Wirel. Netw.*, vol. 17, no. 8, pp. 1775–1794, Nov. 2011.