Simple and Backwards Compatible Layer-2 Routing for Multi-technology Personal Area Networks

Rui Campos and Manuel Ricardo
INESC TEC (formerly INESC Porto) and Faculty of Engineering, University of Porto
Rua Dr. Roberto Frias, 378 – 4200-465 Porto
Email: rcampos@inescporto.pt

***Abstract*—Users increasingly own multiple devices with a number of network interfaces. These devices support different communications technologies that enable the creation of multi-technology Personal Area Networks (PANs). Yet, without a routing solution, connectivity between any two devices in the PAN is not guaranteed. State of the art PAN routing solutions are complex, do not guarantee shortest routes, and are not fully backwards compatible, precluding the plug & play integration of legacy devices in the PAN.**

We propose GLUE, a simple and backwards compatible PAN routing approach, based on transparent bridging and a novel spanning tree mechanism. Simulation and experimental results show the feasibility of our solution, a reduction of one order of magnitude in the route configuration delay, and its good performance in terms of data throughput and delay, when compared against state of the art solutions.

I. Introduction

The cost-effective access to portable and mobile devices is leading to a change in the personal communications paradigm. There is an ongoing migration to a scenario where users own several personal devices with (1) a number of network interfaces supporting different communications technologies, e.g., IEEE 802.11n/ac/ad [1], Wireless USB [2], and Bluetooth [3], (2) increasing computation and storage capabilities, and (3) support of networking functionalities, which were only present in dedicated network devices in the past; this includes devices such as wristwatches[1] and glasses[2]. In addition, thanks to their high storage capacity, huge amounts of data, such as videos, photos, and audio files, can be stored in personal devices. The access to all this data, regardless of the device used, demands easy data transfer among personal devices. The creation of multi-technology, IP-based PANs is envisioned as the enabler to make this happen and bring into the PAN domain the applications and services running over the Internet [4][5]. Moreover, it can enable the deployment of new IP-based applications and services that take advantage of the complementary capabilities of the personal devices in benefit of the user; for instance, a portable gaming console may take advantage of the bigger screen and loudspeakers available at home to enhance the user experience while playing.

Fig. 1 illustrates a multi-technology PAN, which includes existing and envisioned personal devices and wireless PAN

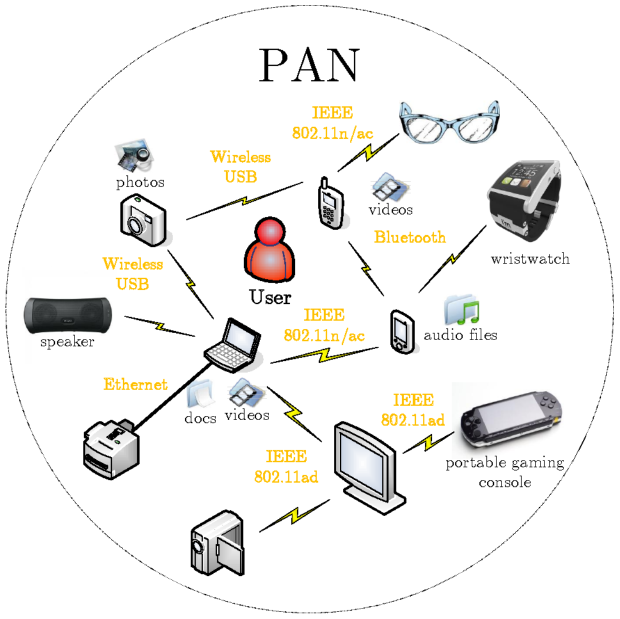


Figure 1: Multi-technology Personal Area Network.

technologies. The communications technologies are heterogeneous with respect to the data rates they provide, but are rather homogeneous when it comes to the support of IP-based applications, with most of them supporting Ethernet emulation [2], paving the way to an all-Ethernet paradigm. Yet, the communications technologies enabling single-hop communications between PAN devices are not enough to create multi-technology PANs; such technologies alone simply enable a set of communications islands, where only devices supporting the same MAC and PHY layers can communicate. Also, the dynamics associated to a PAN, due to the user movement, may trigger membership changes and induce changes in the underlying topology. Thus, two problems arise: (1) the configuration of any-to-any routes within the multi-technology PAN; (2) the reconfiguration of the any-to-any routes when topology changes occur.

State of the art PAN routing approaches are mainly based on reactive routing protocols [2]. Reactive routing solutions, such as AODV [2] and LUNAR [2], do not ensure shortest routes, regardless of the routing metric used, unless a significant amount of signaling is involved in the route discovery process,

[1]http://www.imsmart.com [March, 15th 2013]
[2]http://www.google.com/glass [March, 15th 2013]

as proposed in AODV-DR [7]. Furthermore, reactive protocols introduce route discovery delays, which may be harmful namely for real-time applications. Proactive routing solutions, such as OLSR [2], enable shortest routes and have them immediately available. However, this is achieved at the cost of more signaling overhead and complexity; routes between any pair of nodes are configured independently of those routes being needed or not, leading to a waste of bandwidth, memory space, and processing time. In [6] a hybrid routing approach combining reactive (AODV-like) and proactive routing (OLSR-like) is proposed for a mobile PAN. Reactive routing is used to discover new routes when a PAN topology change occurs and proactive routing is used to maintain the routes; still, the solution inherits the problems referred above. Either reactive, proactive, or hybrid, state of the art routing approaches are not fully backwards compatible, precluding the plug & play integration of legacy devices in the PAN and making deployment hard [2]. Herein, we propose a novel routing approach towards simple, efficient, and backwards compatible PAN routing.

Our contribution is three-fold: (1) a **centralized spanning tree mechanism** based on *Campos's* algorithm [2], which enables the computation of a tree with lower routing cost than the obtained using the standard Rapid Spanning Tree Protocol [2]; (2) a **simple and backwards compatible PAN routing approach**, based on the centralized spanning tree mechanism and the well-known learning bridge algorithm, which reduces in one order of magnitude the route configuration delay, achieves good performance in terms of data throughput and delay, when compared with state of the art solutions, and enables the establishment of a single Layer-2 network on top of multiple communications technologies; 3) the **demonstration that state of the art reactive routing approaches do not ensure shortest routes**, regardless of the routing metric used.

The rest of the paper is organized as follows. Section II provides background information. Section III presents our routing approach. Section IV and Section V refer to the simulation and experimental evaluation. Finally, Section VI discusses the proposed solution overall and Section VII concludes the paper.

II. Background

Existing PAN routing approaches are mostly based on reactive ad-hoc routing protocols, in particular AODV. Thus, in what follows we describe AODV and AODV-DR, an AODV evolution using data rate as the routing metric.

A. AODV

The Ad-hoc On-demand Distance Vector (AODV) [2] protocol is a reactive routing protocol. A source S broadcasts a *Route Request* including the source and target addresses, a sequence number that uniquely identifies the request, a destination sequence number, which is used by intermediate nodes to know whether they have fresh enough information to reply on behalf of the destination, and a Time-To-Live (TTL) value, which controls the maximum number of hops the message can traverse. An AODV node receiving the *Route Request* for the first time creates a new routing table entry for S or updates an existing entry with the fresh routing information. If the current node has recently received the *Route Request*, or the TTL value is 0, the message is silently discarded. In practice, this leads to the establishment of a route between S and D that corresponds to the route taken by the first *Route Request* reaching D.

If the current node is the final destination D, or an intermediate node which has a route to D, after updating the local routing table, it sends back to S a *Route Reply*; otherwise, the current node increments the Hop Count field in the *Route Request* and retransmits the message. The *Route Reply* is then forwarded back to S along the route taken by the first *Route Request*. Each intermediate node is able to forward the *Route Reply* based on the routing table entries created while processing the *Route Request*.

Regarding route maintenance, each node involved in an active route sends out periodically a *Hello* message through all local network interfaces involved in active routes. By default, one *Hello* is sent per second per active network interface. Theoretically, link layer feedback could be used instead. Still, this is not available in practice [2]. After three *Hello* intervals, if the current node did not receive any *Hello* from the next node in the route towards D, it assumes that the link has failed. The current node may then decide to make a local repair by issuing a *Route Request* towards D. If a *Route Reply* is received, the route is re-established without involving S. Otherwise, the current node sends a *Route Error* to its precursors, which in turn forward the *Route Error* to their own precursors until eventually the message reaches S [3]. After receiving the *Route Error*, S triggers a new route discovery to find a new route to D.

B. AODV-DR

The route discovery procedure used by AODV does not ensure the establishment of the shortest route. The "first" route is not guaranteed to be composed of the best links [7]. The main reason for this is that the "first" route to be found is essentially the result of a random process. The simultaneous retransmission of *Route Request* messages by neighbor nodes during a route discovery may result in packet collisions in the wireless media where they can occur; this is the usual case for the wireless technologies that can be used in a PAN. The collision of *Route Request* messages at intermediate nodes may lead to a situation where the destination fails to receive some or even all *Route Request* messages [2]. To overcome this problem, it is recommended that the retransmission of *Route Request* messages at each intermediate node is randomly delayed [2]. Although this solves the packet collision problem, it has impact on the quality of the route found. The introduction of random delays at each intermediate node implies that the first *Route Request* to reach the destination is not necessarily the one traveling across the "fastest" links.

[3]A precursor node is a neighbor of the current node sitting behind it in a route.

The authors in [7] proposed a modification to AODV to actually find the shortest route; hereafter this solution is called AODV-DR. For that purpose, they define the use of a Data Rate routing metric. The route containing the set of links with the highest data rates is selected. This implies the following modifications to the original protocol. At each intermediate node, the cost of traversing a link is considered to be equal to the inverse of the link data rate. Upon receiving a *Route Request* the current node adds to the cost in the message the cost due to the link between itself and the node from which it received the *Route Request*. This process is repeated until the *Route Request* reaches the destination. The protocol is further modified in order to (1) allow intermediate nodes to forward more than one *Route Request* and *Route Reply* messages, (2) allow the destination to accept more than one *Route Request*, and (3) allow the source to accept more than one *Route Reply*, if it leads to the establishment of a better route. In [7] the authors showed by simulation the performance gains of AODV-DR.

III. Proposed Routing Approach

GLUE assumes that PAN devices work simultaneously as terminals and transparent bridges (also known as learning bridges or IEEE 802.1D bridges), defines the transportation of all signaling messages on top of Layer-2 frames, and is based on a master-slave paradigm. The PAN master is defined statically and is kept along the time; it may be a mobile phone or other device always carried by the user. The other devices are called PAN slaves and react to the commands of the master. The PAN master manages the PAN configuration, including (1) the discovery and maintenance of the network topology and (2) the enforcement of the active tree topology required by transparent bridges. The master has global knowledge about the PAN and the slaves act according to its instructions.

In the following, we detail the centralized spanning tree mechanism used by GLUE to (re-)configure the PAN active topology, refer to the GLUE route establishment and frame forwarding procedures, and describe how GLUE is able to support legacy PAN devices.

A. Centralized Spanning Tree Mechanism

1) Topology Discovery and Maintenance: A single mechanism is used for topology discovery and topology maintenance. It consists of an adapted version of the mechanism proposed by Vasudevan *et al.* [2] for electing a leader within a Mobile Ad Hoc Network (MANET). PAN devices are identified by MAC addresses. If a PAN device has multiple interfaces, the lowest MAC address is selected as its identifier. The mechanism works as follows.

The PAN master broadcasts a *Topology Refresh (TR)* message through all its local network interfaces. The *TR* is uniquely identified by the master's MAC address and a sequence number, which is incremented each time a new *TR* is sent. When a PAN slave receives a *TR*, its behavior depends on the number of network interfaces it supports. If the PAN slave is a single interface node, it immediately sends an *Ack* message in unicast to the neighbor from which it received the *TR*. If the PAN slave has multiple network interfaces and it is the first time it receives the *TR* message, the slave retransmits the *TR* through all local network interfaces, except that from which the *TR* was received; all subsequent *TR* messages with the same sequence number are silently discarded and an *Ack* is sent to each neighbor from which a duplicate *TR* was received. In order to make the topology discovery procedure efficient, the *Ack* messages are aggregated at each intermediate slave instead of directly being sent by each slave to the master. Each multi-interface slave waits for a timeout interval, set by the PAN master using the *TR* message, for the *Ack* messages coming from its neighbors. During the timeout interval, each slave collects the topological information included in the *Ack* messages received from its neighbors. When timeout expires, the current slave aggregates all that information into an *Ack* message and sends it in unicast to the neighbor from which it received the first *TR*. Each *Ack* message includes a set of edges modeling the links between neighbor devices in the PAN topology; an edge is characterized by the MAC addresses of the connecting nodes, the local interfaces IDs, and the link metric. The data rate of the link is considered as metric, similarly to what is defined in AODV-DR. This process is repeated at each slave until eventually the master becomes aware of the whole PAN topology.

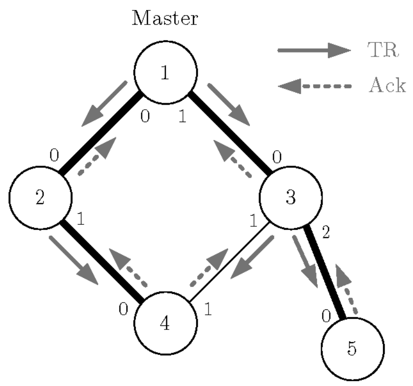


Figure 2: Illustration of the GLUE topology discovery procedure for a 5-node PAN and the control tree established (in bold).

The topology discovery procedure is implicitly used by GLUE to establish a control tree, which is then used by the PAN master to distribute configuration information to the slaves. The control tree is created as follows. Each slave selects the node from which it receives the first *TR* message as its parent node in the control tree. In the opposite direction, each slave is informed that it was selected as a parent node of a node X, upon receiving an aggregate *Ack* message from X. The control tree is dynamic and may change per topology refresh period.

The topology discovery procedure is illustrated in Fig. 2 for a 5-node PAN, where Node 1 is the PAN master and the interface IDs for each node are shown next to each edge endpoint. Node 1 broadcasts two *TR* messages, one per interface. Upon receiving the *TR* message, Node 2 and Node 3 retransmit it across interface 1 and interfaces 1 and 2, respectively. In this example, we assume that the *TR* message retransmitted by Node 2 is the first received by Node 4, and that, before Node 4 retransmits the *TR*, it receives the *TR* from Node 3. The latter is a duplicate *TR*, so Node 4 immediately sends an *Ack* to Node 3. After timeout, Node 4 sends an aggregate *Ack* including information about the edge $(4, 3)$. Node 5 is a single interface slave. When it receives the *TR* from Node 3, it immediately sends an *Ack*. After their timeouts, Node 2 and Node 3 transmit their aggregate *Ack* messages to the master. Node 3's *Ack* includes information about the edges $(3, 4)$ and $(3, 5)$, while Node 2's *Ack* includes information about the edges $(4, 3)$ and $(2, 4)$. Upon receiving these messages, the master is able to construct the topology graph. The control tree constructed during the topology discovery procedure is shown in Fig. 2 using bold lines.

In GLUE, topology maintenance is accomplished by simply running the topology discovery procedure periodically; by default, once per second. If some topology change is detected, a new active topology is configured according to the procedure described in Section III-A2. The master assumes that a topology change has occurred if it does not receive any information regarding some edge or node after three *TR* periods. During the topology discovery procedure, if after the timeout period an *Ack* with the sequence number of the current *TR* is received by a given slave, the *Ack* is directly forwarded to the master.

2) Active Topology Configuration: At the PAN master, the PAN topology is represented by a graph. The graph is then used as a basis for computing the active network topology using *Campos's* algorithm [2]. After the computation of the active spanning tree, the required configurations need to be enforced throughout the PAN. For that purpose, the PAN master issues a *Config* message with the per-slave configurations. The control tree created during topology discovery is used to forward a *Config* message. *Config* includes a set of tuples of the form (slave MAC address, set of interfaces to activate). In order to reduce the size of *Config* messages, the PAN master sends different *Config* messages per network interface; the *Config* message sent through a given local interface only includes configurations for the PAN slaves accessible through that interface. Upon receiving *Config*, each PAN slave searches for its configuration tuple, extracts the tuple from the message, so that it is not unnecessarily propagated downwards, and forwards the message to its children, if it is not a leaf node in the control tree. The *Config* message is transmitted in unicast, so that message delivery is guaranteed over the underlying wireless technologies.

In order to confirm that the active topology configuration is complete, upon receiving the *Config* message and performing the corresponding local configurations, the leaf slaves in the control tree send an *Ack* upwards. As soon as a non-leaf slave receives the *Ack* messages from all its children, it knows that the new active topology ordered by the PAN master is fully configured. It then notifies its parent in the control tree by means of an *Ack* too. This process is repeated until the master is finally notified.

As soon as the active topology is configured, GLUE runs the topology discovery procedure periodically to keep track of the PAN topology.

During the PAN lifetime, one or more PAN devices may leave the network. If the node leaving the PAN is a leaf node in the active topology, no reconfiguration is needed. If the node leaving the PAN is a non-leaf node, a new active spanning tree needs to be configured. In this case, the master computes the new spanning tree using *Campos's* algorithm, disables frame forwarding, configures its new set of active interfaces, and informs the slaves about the new set of interfaces they shall activate by means of a *Config* message. Upon receiving the *Config* message the slaves know that a reconfiguration procedure is taking place. The slaves configure the new set of active interfaces included in the *Config* message and disable frame forwarding until it is guaranteed that the new loop-free active topology is configured. This is guaranteed when a PAN device, a slave, or the master receives the *Ack* messages from all its children or if it is the last hop in the control tree. At that point, the PAN device enables frame forwarding. When the PAN master has received all *Ack* messages from its children, it flushes the local ARP table and sends a *Config* message without any per-slave configuration. This message is transmitted across the control tree to inform the slaves that they can flush their ARP tables too, so that new address resolutions, and implicitly path reconfigurations, can safely take place for the active flows in the PAN.

B. Route Establishment

Every time an IP node wants to communicate with a peer it needs to know the destination MAC address related to the destination IP address. The Addresss Resolution Protocol (ARP) is used for this purpose. When the source does not know the destination MAC address, it broadcasts an *ARP REQUEST*. In GLUE, the *ARP REQUEST* is sent through the active spanning tree and enables transparent bridges to learn the path to the source of the *ARP REQUEST*; the *ARP REPLY* sent back by the destination implicitly establishes the route in the opposite direction. In order to force the use of the ARP procedure for route establishment, GLUE sets the lifetime of the ARP table entries with the same lifetime as the transparent bridges forwarding table entries.

C. Frame Forwarding using Transparent Bridges

In GLUE, the Transparent Bridge running in each PAN node may have multiple physical Network Interface Cards (NICs) associated. However, this is hidden from the upper layers by means of a logical NIC. The Transparent Bridge selects the lowest MAC address among its NICs to become the MAC address of the logical NIC presented to the upper layers[4]. In practice, the logical NIC appears to the upper layers as a regular Ethernet NIC. Upon receiving any frame whose destination

[4]This is the approach followed in mainstream operating systems, such as Linux OS and Windows XP/Vista/7, which support Transparent Bridges.

is the MAC address of the logical NIC, the Transparent Bridge delivers the frame to the upper layers through the logical NIC. In the reverse direction, any data frame received from upper layers through the logical NIC is forwarded according to the standard transparent bridge forwarding procedure [2] and considering the routes established using ARP signaling.

D. Support of Legacy Devices

In a GLUE-enabled PAN, **the integration of a legacy PAN device is "plug & play"**. The legacy device simply attaches to the PAN through any GLUE-enabled device, either through a NIC already added to the local Transparent Bridge, or even through a NIC not yet belonging to it, and can immediately take advantage of the GLUE features, namely the connectivity established among the PAN devices and the PAN-to-Internet connectivity, if available. From the legacy device standpoint, everything happens as if it was connected to a single Layer-2 network. This level of transparency is not supported by state of the art PAN routing solutions.

IV. Simulation Evaluation

GLUE was evaluated using ns-2 and considering AODV and AODV-DR as a basis for comparison. The version 2.31 of the ns-2 simulator was used as departing point. Yet, since ns-2.31 did not support several features needed for our simulation scenarios, it was modified to support (1) multi-interface nodes running on different nominal data rates, (2) AODV-DR, (3) Transparent Bridges, and (4) GLUE[5]; a description on the extension performed to ns-2.31 can be found in [2]. In what follows, we detail the simulation setup and the simulation results obtained.

A. Simulation Setup

We considered the simulation of PANs with random topologies. Ns-2.31 does not provide the means to generate random networks when multi-interface nodes are considered. Therefore, STS[6] [2] was used. STS generates random graphs to simulate spanning tree algorithms; so, it was simply extended to generate random graphs in the *Tcl* format accepted by ns-2. Such random graphs were then used as input to ns-2. Most of the wireless PAN technologies available are based on the CSMA/CA access method [2]. Therefore, we considered IEEE 802.11 as a basis and set up different nominal data rates to different network interfaces. Table I summarizes the input parameters considered in the simulations.

The set $S = \{1,\ 10,\ 100\}$[7] was considered to generate the heterogeneous graphs in STS. Based on the *Tcl* files generated by STS, the ns-2 802.11 NICs were configured with nominal data rates according to the edge weights set by STS. The base nominal data rate was 11 Mbit/s. Thus, the generated random PANs included heterogeneous wireless

Simulation Input Parameters	Values
No. of nodes forming the PAN (n)	{10, 20}
Average no. of NICs per node (A_{NICs})	{1.5, 2, 2.5, 3}
Link nominal data rates	{11,110,1100} Mbit/s
No. of pairs communicating simultaneously (np)	{1, 2, 4}
TCP flow duration	10 seconds
Packet size	1500 bytes

Table I: Summary of the input parameters considered in the ns-2 simulations.

links with nominal data rates of 11 Mbit/s, 110 Mbit/s, and 1.1 Gbit/s. PANs are envisioned to be small networks. Thereby, we simulated 10-node and 20-node PANs; for each case, 30 random topologies were generated. The average number of NICs per node, A_{NICs}, was considered as an input parameter too, so that we could vary the density of the links between PAN devices; A_{NICs} is computed for each random graph generated using STS. PANs are expected to be formed by nodes supporting up to a few NICs, so the maximum value for the average number of NICs per node was set to 3; the other values considered are shown in Table I. Finally, we considered the number of pairs of nodes communicating simultaneously, *np*, as an input parameter, in order to vary the traffic demand within the PAN; the values simulated are also shown in Table I. TCP flows between random pairs of nodes were simulated; the duration of the TCP flows was set to 10 seconds and the packet size was set to 1500 bytes. TCP was selected for two reasons. Firstly, it is the most used transport layer protocol in IP networks. Secondly, it adapts to the available bandwidth and enables easy measurement of the maximum throughput possible between two given nodes.

The following output parameters were evaluated: throughput, delay, route establishment delay, and route configuration load.

B. Simulation Results

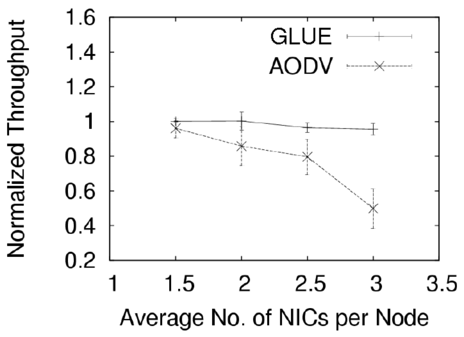
1) Throughput and Delay: The plots in Fig. 3 show the average TCP throughput for GLUE and AODV, normalized to the average TCP throughput obtained using AODV-DR, with the corresponding 95% confidence intervals; additional simulation results can be found in [2].

In general, the GLUE normalized throughput is around 1, which means that GLUE and AODV-DR have similar performance; as expected, AODV performs worse than both GLUE and AODV-DR, due to the use of the "first" route found. Still, interestingly, GLUE can in some cases outperform AODV-DR. This is explained by the increasing number of *Route Request* collisions in AODV-DR, when n, np, and A_{NICs} increase, as it has been shown in [2]. The higher number of *Route Request* collisions has the effect of preventing the discovery of the best route in terms of the data rate metric. The consequence of this effect is visible in Fig. 3, with GLUE able to outperform AODV-DR when n and A_{NICs} increase and, in general, AODV throughput becoming closer to the AODV-DR throughput; however, as A_{NICs} increases the limitation of using a single spanning tree may prevail over the problem

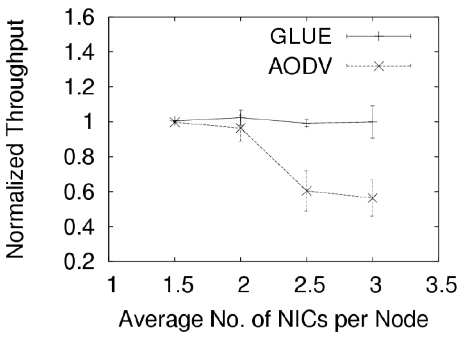
[5]The ns-2.31 simulator developed and the Tcl scripts used in the simulations are available at http://telecom.inescporto.pt/∼rcampos/software.php

[6]http://telecom.inescporto.pt/∼rcampos/STS.tar.gz

[7]S denotes the set of edge weights considered for generating random graphs in STS.



(a) n = 10, np = 2



(b) n = 20, np = 2

Figure 3: Normalized GLUE and AODV average TCP throughput for 10-node and 20-node PANs.

of higher number of *Route Request* collisions and GLUE may suffer a slight performance decrease [2], as shown in Fig. 3a.

These results demonstrate that reactive routing approaches do not ensure shortest routes, regardless of the routing metric used. In practice, due to collisions, the *Route Request* message that would lead to the discovery of the shortest route may not reach the destination; when this happens a suboptimal route is established.

The average one-way delay obtained for each solution is consistent with the throughput results shown above. The GLUE normalized OWD is around 1. GLUE and AODV-DR provide similar OWD values; AODV incurs higher OWD. Due to space limitations, the actual OWD results are not shown herein. They can be found in [2].

The major conclusion is: despite limiting the active PAN topology to a single spanning tree, GLUE outperforms AODV and performs similarly to AODV-DR.

2) Route Establishment Delay (RED): The plot in Fig. 4 shows the GLUE route establishment delay normalized to the route establishment delay introduced by AODV/AODV-DR, with the corresponding 95% confidence intervals. In general, GLUE requires less than 20% of the time spent by AODV/AODV-DR to find a route. This occurs for two reasons. Firstly, AODV/AODV-DR introduces random delays at each intermediate node before re-broadcasting *Route Request* messages; in ns-2 a random delay between 0 and 10 ms is considered, by default. Secondly, at each hop, it uses ARP before sending the *Route Reply* towards the destination. Each AODV/AODV-DR node, except the source, invokes ARP to resolve the IP address of the predecessor into the corresponding MAC address, before being able to send the *Route Reply* to its predecessor in the route. GLUE implicitly uses ARP to establish the route. Therefore, the route establishment delay is simply equal to the time required by the *ARP REQUEST* to reach the destination plus the time required by the *ARP REPLY* to travel from the destination to the source.

GLUE is more efficient than AODV/AODV-DR in the establishment of routes. According to our simulation results, the reduction in the route establishment delay (1 − GLUE normalized RED) may be up to about 90%. GLUE can be this efficient because it maintains an active spanning tree, which already predefines the path between any two pairs of nodes; ARP is just used to implicitly establish the path on top of the spanning tree.

3) Route Configuration Load: Results on the route configuration load are available in [2]. The values for GLUE are similar to AODV and AODV-DR. In GLUE, the master broadcasts one *TR* message per second and the slaves reply with the *Ack* messages, regardless of the number of active flows; AODV and AODV-DR only broadcast *Hello* messages when involved in an active route. As such, they tend to be more efficient when the number of active routes is low. As np increases, the signaling overhead for AODV/AODV-DR increases, as there are more routes to maintain. At the same time, the number of data packets transported by the network increases; this contributes to reduce the GLUE route configuration load, due to its constant signaling overhead.

4) Route Reconfiguration Time: In [2] we perform a thorough analysis of the route reconfiguration time incurred by GLUE. We prove that GLUE and AODV/AODV-DR incur approximately the same delay when it comes to path reconfiguration.

The overall conclusion is: with a route configuration load similar to AODV and AODV-DR, GLUE performs similarly to AODV-DR, outperforms AODV, and reduces significantly the route establishment delay.

V. Experimental Evaluation

In order to complement the simulation analysis and demonstrate the feasibility of the proposed solution in practice, we developed a proof-of-concept prototype. The GLUE proof-of-concept prototype was implemented in Linux OS, based on the built-in Transparent Bridge. The details about the implementation can be found in [2].

Both functional and performance tests were carried out using the GLUE prototype. The scenarios considered are provided in [2]. The average throughput results obtained for intra-PAN flows confirmed the proper operation of the Linux Bridge in each PAN device and GLUE as a whole [2]. In addition, we successfully tested the GLUE ability to seamlessly integrate legacy PAN devices, namely a legacy device running a different OS [2].

VI. Discussion

After comparing GLUE with the state of the art route configuration solutions, AODV and AODV-DR, in terms of throughput and delay, we showed that, in spite of using a single spanning tree, GLUE outperforms AODV and performs similarly to AODV-DR, while introducing similar route configuration load and route reconfiguration times, and avoiding the complexity and inefficiency of the route discovery procedure defined by AODV-DR. Also, we demonstrated that GLUE is more efficient when it comes to the route establishment delay. The GLUE route establishment delay is around one order of magnitude lower than that introduced by AODV/AODV-DR, making the GLUE route establishment procedure almost instantaneous. It is important to realize that, while the comparison was performed against AODV and AODV-DR, the conclusions can be applied with respect to other state of the art reactive routing protocols, such as AODVjr [2], DYMO [2], IEEE 802.11s [2], LUNAR [2], and LOADng [8], which are based on the same fundamental mechanisms.

It is important to note that the obtained results are conservative, since in our simulations we considered heterogeneous PANs including wireless links with nominal data rates whose maximum difference is up to two orders of magnitude. In practice, the differences between the wireless technologies coexisting in a single PAN may reach four to five orders of magnitude. Thus, even better results may be achieved by GLUE.

While we have focused on IPv4 herein, GLUE is IP version agnostic. If running on IPv6, the difference would be the use of the Neighbor Discovery Protocol (NDP) instead of ARP to implicitly establish the routes on top of the active spanning tree. In addition, while focused on multi-technology PANs, GLUE can be used in other types of networks, such as Home Area Networks (HANs), which may include heterogeneous wired/wireless links too and have characteristics similar to multi-technology PANs.

The evaluation results prove the GLUE feasibility and show that it is possible to define a simpler, more efficient, and backwards compatible routing approach for multi-technology PANs without compromising performance. This is accomplished by reusing legacy technology, namely Transparent Bridges, which run together with new simple complementary mechanisms that improve efficiency and keep or even improve performance with respect to state of the art routing approaches.

VII. Conclusions

Users increasingly own multiple devices with a number of network interfaces and supporting different communications technologies. This demands a routing solution to enable connectivity between any two PAN devices. Herein, we proposed a simple and backwards compatible PAN routing approach, based on transparent bridging and a novel spanning tree mechanism. Our results showed the feasibility of the proposed approach and its higher efficiency without compromising performance, simplicity, and full backwards compatibility.

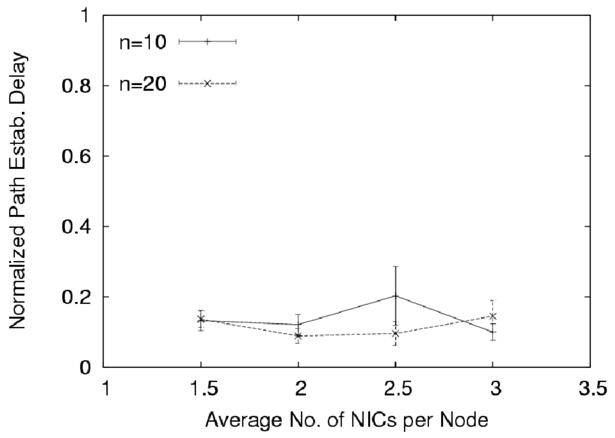


Figure 4: GLUE route establishment delay for 10-node and 20-node PANs normalized to AODV route establishment delay.

As future work, we aim at improving the topology maintenance procedure, so that the PAN master adapts the topology refresh period according to the data flow activity and the mobility patterns of the PAN devices. Also, we plan to consider other metrics, such as energy consumption, to define the active spanning tree and upgrade GLUE to decide which metric shall be used in each networking context.

Acknowledgment

This work was partially financed by the ERDF – European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project FCOMP - 01-0124-FEDER-022701. The authors also thank the support from the FCT under the fellowship SFRH/BD/19429/2004.

References

[1] IEEE 802.11 Work Group, *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, Jun. 2007.

[2] R. Campos, *Joint Path and Address Auto-configuration: an Approach to Multi-technology Personal Area Networks and 802.11-based Stub Wireless Mesh Networks*, PhD, University of Porto, Porto, 2011.

[3] Bluetooth SIG, *Specification of the Bluetooth System (version 4.0)*, Jul. 2010.

[4] M. Jacobsson et al., *PN Secure Networking Frameworks, Solutions and Performance*, IST-027396/MAGNET/WP2.3/DUT/D2.3.2, MAGNET Deliverable D2.3.2, Sep. 2008.

[5] S. Yang, W. Song, Z. Zhong, *Resource Allocation for Aggregate Multimedia and Healthcare Services over Heterogeneous Multi-Hop Wireless Networks*, Wireless Personal Communications, vol. 69, Issue 1, pp. 229-251, March 2013.

[6] R. Ali, *Improving Forwarding Mechanisms For Mobile Personal Area Networks*, PhD, University College London, London, 2011.

[7] E. Johansson et al., *AODV Routing in Ad Hoc Networks with Variable Data Rates*, Technical Report, ISSN 1650-1942, Dec. 2004.

[8] T. Clausen et al., *The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng)*, Internet Draft, draft-clausen-lln-loadng-08 (work in progress), Jan. 2013.