

# Experimental Comparison of Routing and Middleware Solutions for Mobile Ad Hoc Networks: Legacy vs Cross-Layer Approach \*

E. Borgia, M. Conti, F. Delmastro, and E. Gregori  
Pervasive Computing & Networking Lab. (PerLab) - IIT Institute - CNR  
via G. Moruzzi,1 - 56124 Pisa, Italy  
{firstname.lastname@iit.cnr.it}

## ABSTRACT

In this paper we present an experimental evaluation of a full ad hoc network architecture with particular attention to routing and middleware layers. In particular we set up a MANET prototype on which we performed a large set of experiments: in a first phase, we analyzed performances of a proactive and a reactive routing protocols in case of low mobility scenarios in small-medium scale ad hoc networks; then we studied the performances of a first prototype of an optimized p2p system for ad hoc networks (CrossROAD), based on a cross-layer interaction with a proactive routing protocol. Our analysis shows that the use of a proactive routing protocol does not negatively influence system performances, furthermore it allows the optimization of a structured p2p system on ad hoc networks, providing a complete and timely updated knowledge of the network topology. In this way, the overlay network is completely self-organizing and correctly manages network partitioning and topology changes.

**Category and Subject Descriptors:** C.2.2 [Computer Communication Networks]: Network Protocols - *Protocol Architecture, Routing Protocols*; C.4 [Performance of Systems]: *Measurements Techniques, Performance Attribute* **General Terms:** Design, Experimentation, Measurement, Performance **Keywords:** experimental evaluation, routing protocols, p2p systems, cross-layer architecture

## 1. INTRODUCTION

In the last ten years the mobile ad hoc networks research community has been very active. These research activities produced a huge quantity of protocols belonging to different layers of the stack [8]. Results have been validated mainly

\*This work was partially funded by the Information Society Technologies Programme of the European Commission, Future and Emerging Technologies under the IST-2001-38113 MobileMAN project, and by the FIRB-VICOM project.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGCOMM'05 Workshops, August 22–26, 2005, Philadelphia, PA, USA.  
Copyright 2005 ACM 1-59593-026-4/05/0008 ...\$5.00.

through simulative studies. Simulators allow the performance evaluation of protocols in different scenarios, defined by varying several parameters (e.g. number of nodes, mobility models, data traffic), however they often introduce simplifying assumptions that mask important characteristics of the real protocols behavior, as shown in [3] [13]. In order to obtain more realistic performance results, and to evaluate the actual inaccuracy of simulation's models, protocols evaluation via simulation has to be complemented by experiments with real prototypes, even if experimental testbeds are not so easy to implement and only small-medium size testbeds can be set up. The availability of prototypes can also increase the creation of user communities that, experimenting this technology, can provide feedbacks on usability and possible applications relevant for the contemporary society. Currently, only few measurements studies on real ad hoc test-beds can be found in literature, see e.g., [14] [15]. The Uppsala University APE test-bed [14] is one of the largest, having run tests with more than 30 nodes. The results from this test-bed are very important [11] and point out that more research in this direction is required to consolidate the ad hoc networking research field. This paper provides a contribution in this direction. Hereafter, we report our experiences and results obtained by measurements on a real ad hoc network implementing a full ad hoc network architecture. In particular, we set up a MANET prototype on which we performed several sets of experiments; specifically we focused the study on different solutions for routing protocols and middleware platforms. The novelty of this paper is twofold:

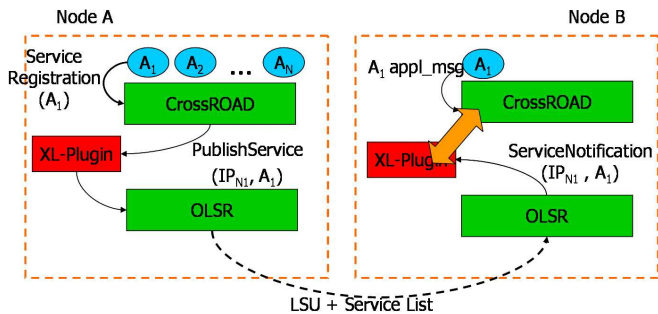
1. we investigate a full protocol stack with particular attention to routing and middleware layers;
2. we evaluate through experimental results the advantages of a cross-layer architecture, presented in [9], mainly focusing on routing and middleware interactions.

The rest of the paper is organized as follows. Section 2 presents the testbed architecture and experimental environment. The methodology of experiments and the performance analysis is reported in Section 3. Finally, conclusions and future works are drawn in Section 4.

## 2. TESTBED ARCHITECTURE AND EXPERIMENTS ENVIRONMENT

Measurements studies on real ad hoc testbed that can be found in literature are generally focused on a single layer of





**Figure 1: Cross-layer interactions between middleware and routing through the XL-plugin.**

all participants, making the overlay network self-organizing. In order to analyze CrossROAD performances on top of a real multi-hop ad hoc network, a particular implementation of OLSR has been used [17]. This implementation allows the development of an internal plugin for the definition of additional information to be sent on the network through the routing protocol, and the related processing. This represents a first approach to the development of a complete cross-layer architecture, aimed at implementing interactions between middleware and routing layers. The plugin developed during this study for version 0.4.8 of [17], has been called *XL-plugin*. This plugin consists of a dynamic library loaded by the routing daemon at startup. To implement the cross-layer interactions, a messages' exchange protocol between middleware and routing has been defined. Fig.1 represents a simple example of how cross-layer interactions can be exploited in the creation of the overlay between two nodes (A and B). The interaction between CrossROAD and the plugin starts when the application  $A_1$ , running on node A, registers the related service identifier creating a new instance of CrossROAD. In this way the local node joins the overlay sending to the plugin a message of *PublishService* containing its IP address and the service identifier associated to the specific application. When the plugin receives this message, it encapsulates that information in the first routing packet that will be sent on the network, and it stores this content in the *Local Services Table*, which maintains the list of services provided by the local node. On the other hand, when this message is received by another node (e.g. B), the plugin processes the additional information, and stores it in the *Global Services Table*, maintaining all services currently provided by every node of the network. At this point, when the application decides to send a message on the overlay, first CrossROAD checks the consistency of its internal data structures with the plugin, and then it selects the optimal destination for that message between nodes currently running the overlay and contact it through a simple p2p connection. In this way, all remote connections, needed by Pastry to build and maintain the overlay data structures, are eliminated, and every node of the overlay knows all the other participants, avoiding the multihop middleware routing introduced by the subject-based policy of Pastry [16]. In addition, until CrossROAD is active, the plugin sends periodical messages on the network as a maintenance of provided services, while, as soon as CrossROAD stops running (even in case of application's crash), the plugin sends a *DisconnectMessage* with which informs the others about the disconnection of the local node from the overlay. In this way,

when a node decides to join the overlay, CrossROAD directly asks to the plugin the IP addresses of all nodes providing the same service, autonomously building the complete overlay. To compare and contrast the Pastry model with its cross-layer enhancement, in our prototype, we integrated on top of FreePastry [2] and CrossROAD a simple application of Distributed Messaging (DM). Nodes running DM set up and maintain an overlay network related to this service. Once a node has created/joined the overlay, the application provides the possibility to create/delete one or more mailboxes distributed on the other nodes and to send messages to them. A mailbox's physical location is randomly selected applying the hash function to the associated identifier, used as the key value of the related messages.

### 3. EXPERIMENTS AND PERFORMANCE EVALUATION

In order to obtain a complete performance evaluation of the entire system on a real multi hop ad hoc network, several sets of experiments had to be executed, focusing on different aspects of the MANET architecture. First of all we examined routing performances in case of mobility scenarios and topology changes. Then we analyzed CrossROAD in terms of functionalities, overhead introduced on the network, and delays needed to build the overlay network and to become aware of overlay and topology changes.

#### 3.1 Routing performances

In this section we present the performance evaluation of the routing protocols on a string topology. In a previous work [5], we considered only static networks and we focused on the overhead introduced by them; here we introduced mobile nodes in order to evaluate the impact of the mobility on routing protocols. To this end, we considered a string topology network of four nodes and we performed three sets of experiments increasing the number of mobile nodes (all the scenarios are shown in fig.2). In this scenarios the connectivity between the sender and the receiver changes from 1 hop to 3 hops and viceversa during the experiments. To have a comparison between OLSR and AODV, we studied the *Packet Delivery Ratio* (total number of packets received at the intended destinations divided by the total number of generated packets) and the delay needed for the network's reconfiguration due to the movements of nodes. In addition to the routing protocol, we introduced some traffic at the application layer using the ping utility. This guarantees that AODV runs in a complete manner; otherwise, without any application-level traffic, its routing information is reduced only to *Hello* packets exchanges. In particular, in our scenarios, all the nodes start running the routing protocol and, after an initial period necessary for the network topology stabilization, node A pings continuously node D until the end of the experiment. We repeated the same set of experiments several times; obtained results were similar, so we present an average of them. One may argue that similar set of experiments were already available in literature. On the other hand we think that there are several main reasons to perform these experiments in our environment:

1. Our cross layer architecture assumes an underlying proactive routing protocol. Comparing AODV and OLSR performance enable us to better understand if and how the proactive assumption impacts on the overall system performance. Previous results [6] and those

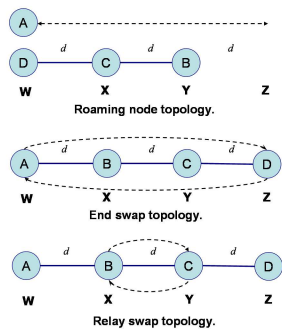


Figure 2: String topology

presented in this paper indicate that in small-medium scale networks and low mobility scenarios OLSR does not penalize the system performance;

- Measurements related to the topology management provide a reference to understand the behavior of the p2p protocols and, in the specific case of CrossROAD, also give a direct measurement of the expected delays in the overlay construction and reconfiguration. Therefore, a better understanding of the routing protocol performance will be useful when analyzing the behavior of the p2p platforms.

The configuration and the methodology used for the experiments follow those published in [12], and can be taken as a reference for our performance evaluation. In the first set of experiments, called "Roaming node", there are 3 static nodes (B, C, D) and the "roaming" node A. The experiment lasts 2 minutes: from the initial position W, node A starts moving and every 20sec it reaches the next position in the line (X, Y, Z); once it has reached the last position Z, it immediately moves in the opposite direction following the reverse path and reaches the starting position near node D after another minute. The second set of experiments is referred as "End node swap" due to the movement of the two communicating nodes (A and D), while the rest of the network remains in the same configuration. More specifically, the two end nodes maintain their initial position for the first 20sec of the ping operation, then they start moving reaching the next position in the line every 20sec. The experiment lasts other 20sec after the end nodes have swapped their positions. The last set of experiments, named "Relay swap", is similar to the previous one: there are 2 mobile nodes in the network that change positions during the test. In this case after 20sec from the beginning of the ping operation, central nodes start moving and swap their positions after 20sec, then they remain in this new configuration until the end of the experiment (it lasts 60sec). In all the performed experiments each mobile node moves along the line with a speed of about 1m/s, since we are interested in investigating low mobility scenarios. Looking at the Packet Delivery Ratio index, as shown in Table 1, we notice that increasing the complexity of the proposed scenarios, the performance of the two routing protocols decreases up to about 60% of packets delivery in case of Relay swap scenario. Specifically, in the Roaming node scenario we can note that both protocols have similar behaviors: there is a packet loss of about 25%. Examining the log files, we observe that, for both protocols, packet losses mainly occur when node A goes beyond position Y and reaches the string's end; specifically this represents the time in which the connection A-D changes from

PDR	Roaming node	End node swap	Relay swap
AODV	0.87	0.67	0.60
OLSR	0.83	0.77	0.66

Table 1: Overall Packet Delivery Ratio.

2-hop to 3-hop connection, due to the loss of the direct link A-C. In the End swap scenario, the proactive protocol performs better than the reactive protocol: delivered packets increase of 10%. OLSR introduces the high percentage of its packet loss in the last 40sec of the test-run when the connection becomes again a 3-hop connection; on the other hand at the beginning of the experiment all packets were correctly received since the network was already stabilized when data transfer started. In contrast AODV distributes uniformly its packet loss during the entire test-run. As previously said, in the third set of the experiments the packet delivery ratio of OLSR and AODV decreases up to 66% and 60%, respectively. In particular, from the log files we notice that packet losses occur during the relay swap phase (i.e., from 20 to 40sec), in which only half of the number of packets generated by node A reaches the destination successfully. To evaluate the delay introduced by the two routing protocols due to nodes' movements, we measured the time needed to update the routing table for OLSR and to discover new paths to the destination for AODV. In the first scenario, when node A moves toward position Z, OLSR requires 5sec to discover a 2-hop path to D after the direct link A-D is lost; while it needs 10sec when the path in the connection increases from 2 to 3 hops. AODV introduces a delay of 2sec for the first topology change, and 7sec for the second one. Both protocols do not introduce any additional delay in the reverse path (from Z to W position). In the End swap scenario, OLSR introduces a delay of 15sec when the topology changes from a fully connected (each node see all the others) to a topology of three hops. In the same topology change, AODV experiences a delay of 10sec but it also introduces a similar delay to move from the starting configuration to a fully connected topology. In the last scenario, during the relay movement, OLSR introduces a delay of 15sec for the routing table reconfiguration, while AODV requires 11sec to discover a new route to the destination.

### 3.2 Overlay network performances

In order to evaluate CrossROAD performances, several sets of experiments were performed using 8 nodes (see fig.3) but only 6 running the overlay. In fact nodes B and G just worked as routers. During the experiments all nodes were synchronized and started running OLSR enhanced with the *XL-plugin*. In the first set of experiments, the overhead introduced by CrossROAD and FreePastry has been measured. The experiment consists of defining an order with which nodes start running the overlay, and no message generation is required from the application. In this way we can measure only the overhead introduced for the overlay construction and management. The main difference between these experiments is due to the self-organizing nature of CrossROAD and its complete independence on building and managing the overlay. In fact, as shown in fig.4, nodes B and G, that just work as routers, only perform routing traffic that is almost negligible (about 50Bps). But also the other nodes, which start running the overlay with a delay of 10sec one from the other (interdeparture time), introduce an over-

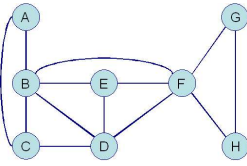


Figure 3: Experimental network topology

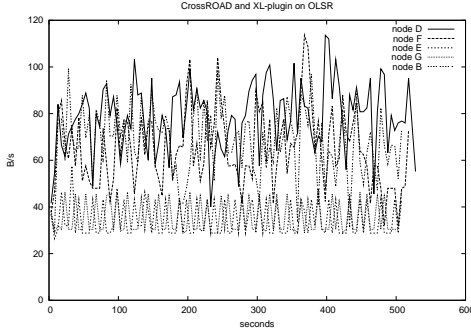


Figure 4: CrossROAD: Throughput related to main nodes.

head less than  $100Bps$ . Comparing these results with Pastry performances on the same set of experiments (see fig.5), we can notice that nodes running the overlay introduce a much higher overhead than in case of CrossROAD, with several traffic peaks corresponding to TCP and UDP connections used to initialize and maintain the overlay data structures. Another important feature of CrossROAD is represented by the timeliness with which every node at the startup becomes aware of the other participants. Since the *PublishService* message is sent on the network as soon as the next routing packet is ready, the delay introduced for each node to collect information about all nodes taking part to the overlay, is lower than in case of establishing one or more remote connections. In order to obtain these results, we set up two sets of experiments using an interdeparture time of  $10sec$ . In both cases the first node starts running CrossROAD and waits for being notified from the XL-plugin that some other nodes join the overlay. In particular in the first set the second node is 1-hop distant from the first, while in the second set the second node is 3-hop distant from the first node. As shown in table 2, the first node of the overlay experiences different delays in those experiments, due to the physical distance from the second node. When they are 1-hop neighbors, the first node experiences a delay of  $180msec$ , while in the second case it experiences a delay of  $325msec$ . The measured delay corresponds to the interval time between the sending of the *PublishService* from the second node to the notification of the list of nodes of the overlay to the first node. All the other nodes experience a delay of about  $30-40msec$  because when they start the overlay, the plugin has already stored at least one other node providing the service in its local data structures. Hence, in this case, the delay represents only the processing time needed to exchange messages between CrossROAD and the plugin on the local node.

As the last set of experiments, we analyze a possible network partitioning and the consequent reaction of CrossROAD in the overlay management. To do this, a new network topology, shown in fig.6, has been set up. Specifically, the ad hoc

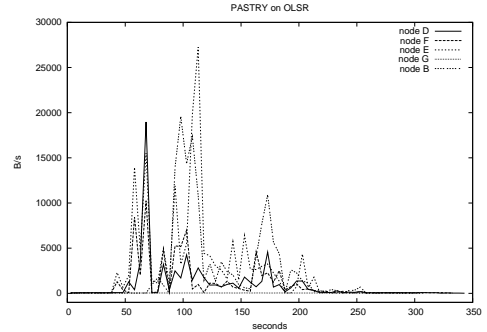


Figure 5: Pastry: Throughput related to main nodes.

Delays for overlay establishment	First node	Other nodes
$2^{nd}$ node at 1-hop distance	$180msec$	$40msec$
$2^{nd}$ node at 3-hops distance	$325msec$	$30msec$

Table 2: CrossROAD delays for the overlay construction.

network consists of 5 nodes, and only nodes in adjacent positions are in the transmission range of each other. All nodes start running OLSR enhanced with the XL-plugin, and after about  $30sec$  to have the network topology stabilized, they start executing CrossROAD with a delay of  $10sec$  from each other. When all nodes are correctly connected to the overlay, node C starts periodically sending an application message with a specified key value. The period with which the message is sent on the overlay is set to  $1sec$ . Initially, the key value results to be logically closest to the node identifier of B, hence node C sends those messages directly to B. Then after about  $30sec$ , node C starts moving towards position  $X$  with a speed of about  $1m/s$ , generating a network partitioning: nodes A and B create an independent ad hoc network as well as nodes C, D, and E. Since the direct link B-C is lost, the cross-layer interaction between CrossROAD and the routing protocol allows node C to become aware of the network partitioning and the consequent removal of nodes A and B from the overlay. Hence, the successive messages sent by node C on the overlay with the same key value, are directly sent to the new best destination: node D. After 2 minutes, node C starts coming back to the initial position re-establishing a single ad hoc network. At this point, the following messages are sent again to node B. As shown in fig.7, CrossROAD, thanks to the cross-layer interaction with the routing protocol, correctly manages data distribution in case of overlay and network partitioning. Specifically, the figure shows that *i*) during the first phase (a single ad hoc network), node C data are stored on node B; *ii*) when the partition occurs, after a transient, node C data are delivered to node D; and *iii*) when the network is again connected, C data are again stored on node B. It is important to specify that CrossROAD, before sending each application message, requires to XL-plugin the list of nodes currently taking part to the overlay. At the same time, the XL-plugin has to maintain the consistency of its internal data structures with the processing of service discovery messages periodically received. For this reason, a validity timeout has to be associated to each entry of the *GlobalServiceTable* and a *ServiceMaintenance* message has

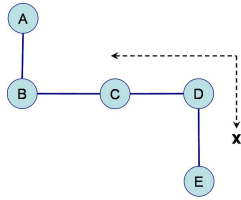


Figure 6: Network Partitioning Topology

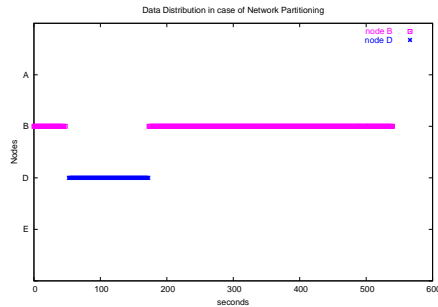


Figure 7: CrossROAD reaction to network partitioning

to be sent periodically on the network. In order to guarantee the maximum timeliness of CrossROAD to overlay topology changes, this period is set equal to that used by OLSR to flood Hello messages on the network (2sec). This procedure does not negatively influence the system performances because each service identifier is simply represented as 32-bit value. Hence, even changing the maintenance period, the total overhead does not considerably change. In conclusion, cross-layer interactions between routing and middleware make every node of the overlay completely autonomous from the others, optimizing data distribution and recovery for upper applications without introducing additional overhead on the network. In this way, all the advantages introduced by the use of a structured p2p system are furtherly optimized and all disadvantages introduced by Pastry on ad hoc networks are completely removed.

## 4. CONCLUSIONS

During this study a large number of experiments has been set up in order to obtain real measurements on a full ad hoc network architecture. In particular routing protocols and middleware platforms have been evaluated in terms of overhead and reconfiguration delays in case of mobility scenarios and network topology changes. In addition, we set up a real prototype of an optimized p2p system (CrossROAD) that exploits the cross-layer interactions with a proactive routing protocol (OLSR), and the first experimental results of this platform have been presented. Specifically, this particular implementation of the cross-layer architecture is mainly focused on routing and middleware layers, aimed at optimizing the overlay network management on ad hoc networks. This real testbed demonstrated that: *i*) the use of a proactive routing protocol (OLSR) does not penalize the system performances either in terms of PDR and reconfiguration delays in static and low mobility scenarios; *ii*) CrossROAD drastically reduces the overhead introduced by a classical p2p system on ad hoc networks, and correctly manages cases of network partitioning and topology changes. The next step

of our study is to create a larger testbed in order to evaluate the overall system performances on a large scale ad hoc network.

## 5. REFERENCES

- [1] AODV. <http://user.it.uu.se/~henrik/aodv/>.
- [2] FreePastry Website. <http://freepastry.rice.edu>.
- [3] G. Anastasi, E. Borgia, M. Conti, and E. Gregori. Wi-Fi in Ad Hoc Mode: A Measurement Study. In *Proc. of PerCom 2004*, Orlando, Florida, March 2004.
- [4] E. Belding-Royer. *Routing approaches in Mobile Ad Hoc Networks*. IEEE Press and John Wiley and Sons, New York, 2004.
- [5] E. Borgia. Experimental Evaluation of Ad Hoc Routing Protocols. In *PWN Workshop in PerCom 2005 Proceedings*, Kauai Island, Hawaii, Mar. 2005.
- [6] E. Borgia, M. Conti, F. Delmastro, and L. Pelusi. Lessons from an Ad-Hoc Network Test-Bed: Middleware and Routing Issues. In *Ad Hoc & Sensor Wireless Networks, An International Journal*, Vol.1, Numbers 1-2, 2005.
- [7] J. Broch, D. A. Maltz, and D. B. Johnson. Quantitative lessons from a full-scale multi-hop wireless ad hoc network testbed. In *Proc. of WCNC 2000*, 2000.
- [8] I. Chlamtac, M. Conti, and J. Liu. Mobile Ad hoc Networking: Imperatives and Challenges. *Ad Hoc Network Journal, Vol.1 N.1*, Jan-Feb-Mar 2003.
- [9] M. Conti, G. Maselli, G. Turi, and S. Giordano. Cross layering in mobile ad hoc network design. *IEEE Computer*, Feb. 2004.
- [10] F. Delmastro. From Pastry to CrossROAD: Cross-layer Ring Overlay for AD hoc networks. In *MP2P Workshop in PerCom 2005 Proceedings*, Kauai Island, Hawaii, Mar. 2005.
- [11] P. Gunningberg, H. Lundgren, E. Nordstrom, and C. Tschudin. Lessons from experimental manet research. *Ad Hoc Networks Journal, Special Issue on "Ad Hoc Networking for Pervasive Systems"*, Vol. 3, Number 2, March 2005.
- [12] H. Lundgren. *Implementation and Experimental evaluation of Wireless Ad hoc Routing protocols*. PhD thesis, <http://publications.uu.se/theses/abstract.xsql?dbid=4806>.
- [13] H. Lundgren, E. Nordstrom, and C. Tschudin. Coping with Communication Gray Zones in IEEE 802.11 based Ad Hoc Networks. In *Proc. of WoWMoM 2002*, Atlanta, GA, September 2002.
- [14] D. of Computer Systems at Uppsala (Sweden). *APE: Ad hoc Protocol Evaluation testbed*. <http://apetestbed.sourceforge.net/>.
- [15] R. S. Gray et al. Outdoor Experimental Comparison of Four Ad Hoc Routing Algorithms. In *Proc. of MSWiM 2004*, Venice, Italy, October 2004.
- [16] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large scale peer-to-peer systems. *LNCS*, 2218:329–350, 2001.
- [17] A. Tonnesen. *OLSR*. <http://www.olsr.org>.