An Emulated IPv6 based Self-configuring Multi-hop Mobile Network Testbed: Architecture and Performance Analysis

Dev Pramil Audsin*, A. Hamid Aghvami†, and Vasilis Friderikos‡

Centre for Telecommunications Research
King’s College London, London, UK WC2R 2LS
Email: {dev.audsin, hamid.aghvami, vasilis.friderikos}@kcl.ac.uk

Abstract—This paper describes the architecture of a self-configuring Multi-hop mobile Network and analyse the performance of the ad-hoc routing protocols. A testbed is deployed to analyse the performance of the ad-hoc routing protocols in a self-configuring multi-hop environment. The testbed comprises of a set of mobile nodes and an Internet gateway. The mobility of the nodes and the resulting network topology are emulated. During the experiments, the mobile node move virtually around a specified area and communicate with the gateway by sending UDP packets. The routing between the MNs and the Internet gateway is established by the ad-hoc routing protocols namely AODV and OLSR. Based on the measurements obtained during the experiments, some of the well known performance metrics are calculated.

I. INTRODUCTION

The architecture of Self-configuring Multi-hop mobile Network (SMN) defines that the Mobile Nodes (MN) co-operate with each other in self-configuring their network parameters. As the name implies, the SMN exhibit two characteristics. First the nodes in the SMN have the capability to self-configure themselves in a distributed fashion. Second not all the MNs are in direct link connectivity with the Internet Gateway (IGW) or the base station. In other words, some of the MNs might be multi-hop from the IGW and hence the MNs need to co-operate and extend the coverage and functionalities of the IGW. The second characteristics require that the MNs forward other nodes traffic and in turn act as wireless routers. This paper concentrates on the second characteristics of the SMN such that the MNs located more than one hop away from the IGW are able to communicate with the IGW. To provide multi-hop connectivity between the nodes, ad-hoc routing protocols are employed.

To study the functionalities of SMN and analyse the performance of the multi-hop routing protocols, a SMN testbed is employed. This paper studies the packet level network performance of a SMN, based on the measurements in a wireless mobile multi-hop network with one IGW and nine MNs. The defined SMN functionality should enable the MNs located multi-hop from the IGW to communicate with the IGW and send user data packet to the IGW. Two ad-hoc routing protocols namely Ad-hoc On-demand Distance Vector (AODV) and Optimised Link State Routing (OLSR) are selected and they run over wireless interface in ad-hoc mode.

The mobility of the nodes is emulated by creating a dynamic virtual topology according to the defined mobility pattern and the packets between the nodes that are not currently within the virtual transmission range are filtered and dropped by the nodes. However, if the nodes (as defined by the virtual network topology) are reachable through multi-hop routing, the packets are forwarded through the routes created by the routing protocols.

This paper studies the impact of the ad-hoc routing protocols on the network performance under various mobility scenarios. For performance comparison, two well known routing protocols are selected. A reactive routing protocol AODV and a proactive routing protocol OLSR are used for the performance comparison.

The remainder of the paper is organised as follows: Section II describes the multi-hop connectivity and a brief description of the AODV and the OLSR functionalities. Section III explains the architecture of the SMN testbed which includes the experimental configuration, mobility scenario, network topology, and the traffic generators. Section V describes the system design of the mobility emulator. Section VI briefly describes the chosen mobility model which is the Random Way Point (RWP) mobility model. Section VII defines the chosen performance metrics utilised for the performance analysis. Section VIII analyses the performance obtained by running the experiments on the SMN testbed. Finally section IX provides the conclusion of the paper.

II. MULTI-HOP CONNECTIVITY

Multi-hop Connectivity Ad-hoc routing protocols are chosen to provide connectivity between the nodes located within the SMN. For the convenience of performance comparison, a reactive and a proactive routing protocol are chosen. AODV [1] is chosen as the reactive routing protocol whereas OLSR [2] is chosen as the proactive routing protocol. In this section, both the routing protocols are briefly described. It is recommended to read the respective RFCs for detailed architecture of the above mentioned routing protocols.

A. AODV

AODV initiates route discovery to a destination only on demand. When a source node has a data to send to a destination, it queries internal data structures and routing table...
for a route to the destination. If no route is found, it buffers the packet and initiates the route discovery process. The route discovery process is triggered by multicasting a Route Request (RREQ) message. The intermediate nodes that receive a RREQ message checks its internal data structures and routing table entries. If a node has a route towards the destination specified in the RREQ message, it unicasts a Route Reply (RREP) back to the source node. Otherwise, the intermediate node forwards the received RREQ message. This process is continued until the RREQ reaches the destination node or the RREQ message reaches the specified hop count. Eventually, the source node receives the RREP message indicating the route to the destination node. It then sends the data packets to the destination node. To reduce the flooding and congestion of the signalling packets, the AODV implements rate limiting mechanism. In rate limiting mechanism, a node cannot generate more than a specified number of RREQ messages per second. After multicasting a RREQ, the source node waits for a RREP and sets a timer. If no RREP is received within the expiration of the timer, the source node might try again to discover a route by multicasting another RREQ message, up to a specified maximum retry times. If no route to the destination node is found even after trying maximum number of times, the route discovery process is stopped. As an optimisation, AODV employs expanding ring search, in which the initial RREQ is sent with a hop limit of 1. If no RREP is received within a specific time, the RREQ message is sent with a large hop limit. If no reply is received, the next RREQ is sent larger hop limit until a certain maximum value.

B. OLSR

OLSR is a table driven routing protocol and the routing information is available readily when needed. To achieve this, each node exchange the topology information with other nodes periodically and every node maintains the topology of the whole network. Hence, each node periodically constructs and exchanges a list of its 1-hop and 2-hop neighbours. OLSR uses nodes that act as Multi Point Relays (MPR) to optimise flooding. Based on the exchanged link state information, each node selects a set of nodes as MPR. In OLSR, only MPR nodes can forward the control traffic intended for diffusion into the multi-hop network. Therefore, the MPRs provide an effective mechanism for the propagation of network topology and optimise the number of active relays needed to cover all the 2-hop neighbours. In other words, a node forwards an OLSR topology information packet only if it is selected as a MPR thereby optimising the propagation of topology information. The routes are calculated based on the topology information that each node holds. The OLSR periodically transmits link state information over the selected MPR nodes to construct and maintain its routing table. Therefore, each node exchanges the topology information with other nodes periodically and every node maintain the topology of the whole network. Upon convergence, an active route is created at each node to reach any destination node in the network.

III. TESTBED ARCHITECTURE

This section describes the testbed architecture including the network configuration, network topology, design of the mobility emulator and the traffic generators.

A. Network Configuration

The SMN testbed architecture is presented in Figure 1 and comprises of nine MNs and one static IGW. All the MNs are equipped with one wired ethernet and one wireless interface in ad-hoc mode. The wired Ethernet acts as a control channel while the wireless interface acts as a data channel. The static IGW contains two wired Ethernet and one wireless interface in ad-hoc mode. One of two wired Ethernets is connected to the global IPv6 Internet and provides Internet connectivity to the MNs located within the SMN. The second wired Ethernet acts as control channel to distribute the virtual network topology and the wireless interface is used as the data channel to communicate with other MNs. The network topology informations are passed through the control channel while the routing and user data packets are transported through the data channel. The testbed employs both reactive protocol AODV and proactive routing protocol OLSR. However, only one routing protocol is active at a time to calculate the route between the multi-hop MNs and manipulate the Linux kernel routing table. In other words, while running the experiments either AODV or OLSR is active but not both of them at the same time. For performance analysis, first a set of experiments with the AODV enabled is run and the performance metrics are calculated. Then the set of experiments are repeated with the OLSR to provide connectivity and route calculation.

B. Network Topology

The experimental network topology has two network areas: a SMN and a global IPv6 Internet. The SMN is connected to the global IPv6 network through the IGW. An IGW acts as an interface or a bridge between the SMN and the global IPv6 Internet. Any user traffic from the MNs located in the SMN to the IPv6 Internet are transported through the gateway. The ad-hoc routing protocols AODV and OLSR are responsible for maintaining the connectivity and data path between the nodes in the SMN. The SMN testbed utilises the AODV-UU for IPv6 [3] and IPv6 complaint OLSR [4] implementations. If needed, more networks can easily be added. The whole SMN tested is based on the fixed network and the mobility is made viable by emulation.

C. Mobility Scenario

The mobility of the nodes and the resulting link connectivity is emulated. The MNs move virtually according to the mobility pattern stored in a mobility configuration file. Packets received by the MNs that are not within the virtual transmission range of the source node or the forwarding node with respect to the virtual topology are filtered out and dropped. However, if the nodes (as defined by the virtual network topology) are reachable through multi-hop routing, the packets are forwarded through the routes created by the routing protocols. The virtual dynamic topology of the SMN is controlled by a central node called controller via the wired Ethernet connection. It informs the MN about the virtual location or position of a MN in the defined network topology. The actual filtering, mangling and dropping of the routing and data packets are achieved in the MAC layer by the use of netfilter [5] modules and ip6tables [5]. The setup is implemented by the NRL Mobile Network Emulator (MNE) [6] running on the MNs and the controller node. In our testbed, the IGW exhibits the
functionality of the controller node. The MNs move in such a way that each of them can establish a direct link to another MN forming a fully meshed network topology. The following section describes the architecture of the mobility emulator.

D. Traffic Generators

The SMN testbed is capable of generating both UDP and TCP packets. Two traffic generators are used in the testbed: Multi-Generator (MGEN) [7] and bench6 [8]. MGEN is used for the generation of UDP packets. MGEN generates real time UDP traffic patterns so that the network can be loaded in a variety of ways. Hence, MGEN provides the ability to perform network performance tests and measurements using UDP over IPv6 traffic. However, MGEN does not have the feature to generate TCP packets. Hence bench6 is used to generate TCP traffic. Bench6 can be used to generate bulk data transfers to emulate a FTP file transfer between the IGW and the MNs. The generated traffic is logged both in the sender and receiver for analysis. Performance analysis such as packet delivery fraction, end to end delay, packet loss rates and throughput can be calculated with the help of the log data. All the nodes including the IGW are equipped with the MGEN and bench6 software.

IV. MOBILE NETWORK EMULATOR SYSTEM DESIGN

Each node has two interfaces - wired Ethernet and wireless 802.11b in the ad-hoc mode. The wired Ethernet is used as the control channel to emulate the topology and the wireless channel is used to send the user data. Figure 2 shows the high level architecture of the emulator. Each node in the testbed consists of a Mobile Network Emulator (MNE), traffic generator and visualisation tools to visualize the topology and mobility of the nodes.

The MNE uses the ip6tables network filtering capability of the Linux operating system. The emulator includes a software process that writes location information (e.g. emulated GPS longitude and latitude data) to shared memory. Other processes in the testbed, such as traffic generators, can read from this shared memory and use the dynamic information provided. The MNE can also be used to generate motion patterns that can be communicated via a control channel using IP multicast to all nodes participating in the emulation. In this way, emulation nodes can listen on this channel to pick up pertinent location information for other nodes. The emulation nodes can then compare this information with their own emulated location. By having this location and other information (e.g. transmit power) available locally, nodes can determine the nodes with which they should and should not be able to establish the effective communication links. They can do this using a variety of wireless propagation models, the simplest of which is a basic range model. Using this model, when the calculated distance to a given node is beyond a specified range, all incoming packets from that node on the wireless interface are dropped before being delivered to the appropriate application using a MAC address filter. To the network applications, this makes the local link between the two nodes appear unusable, just as it would be in the real world.

V. FUNCTIONAL COMPONENTS OF THE EMULATOR

In this section the components of the emulator are described in detail. Figure 3 shows the full functional components of the emulator which are defined below.

A. Dual Channel

Control traffic to run a dynamic emulation is designed to operate independently from the user traffic sent over the dynamic network topology created by the emulation. The location information can be sent via a wireless or wired interface regardless of which interface is being used for emulating dynamic routing and user traffic. For example, we can make a test-bed with a single hard-wired interface, a single wireless interface, or both a wired and a wireless interface. In the single-wired interface environment, the routing protocols can be exercised fully, but actual results may suffer under congestion, since control traffic to run the experiment is sharing the same channel as potentially congested user traffic. This possibility is still supported in function by the design and it may be cost or complexity-advantageous to use a single interface (e.g., Ethernet) for basic testing purposes. Continuous transport of control traffic is supported even under dynamic motion and virtual network fragmentation because the control channel uses a well-known multicast address and port number that is not filtered by ip6tables and is received continuously by all nodes. Instead of using a wired Ethernet, the use of a wireless interface (such as a WLAN) may better reflect some of the MAC layer behaviour and interference issues unique to wireless operation. In both of these cases, when a single interface emulation environment is used and the network gets too congested, the location information sent out by the MNE can interfere with the normal test data, resulting in a lower throughput and increased latency. Similarly, the normal user traffic can interfere with the location information, resulting in nodes receiving stale position updates for other nodes. If nodes have out-of-date position information, blocking and unblocking may not be done at the correct times. Some priority packet processing of the control traffic can be done to provide probabilistic improvements to this condition, but interference effects cannot be completely avoided in this case. Therefore, two network interfaces are used for general operation of the emulator, especially when congested user data testing is
desired. Since TCP traffic will naturally operate near or at network congestion levels, this is a fairly typical assumption to make for most realistic testing scenarios. In addition to two interfaces, the most recommended configuration is to use both a wireless (e.g., 802.11 in the ad hoc mode) and a hard-wired Ethernet interface. The normal test data, including the routing protocol and user data, can be sent out via the wireless interface, just as it would be in a real-world scenario. The hard-wired interface is used as the supplemental back channel for the emulation control information and status. Once again, a second wireless interface on a separate channel could also be used for the back channel instead of a wired interface, but this is likely of no benefit for a tabletop emulation and may create additional wireless channel interference. Therefore, we use a wired Ethernet interface for control channel purposes.

B. Packet Filtering

To emulate a nominal wireless link range, a method for blocking packets from given nodes is needed. This can be achieved by the input filtering function of ip6tables [5]. The ip6tables is a standard tool that comes with the Linux operating system to filter the packets on a given network interface. Before a data packet is passed on for further network kernel processing, it is examined by applying a set of rules to determine when the packet should be accepted or dropped from the "input" chain. A similar set of rules applies to the "output" chain. It is important to note that inserting and deleting an entry in the input chain of the ip6tables can emulate the wireless connectivity or topological dynamics. When two nodes are out of range from each other, blocking the output of wireless interface does not emulate the wireless neighbour link outage properly, because in broadcast packet radio systems other nodes may be in range while a particular node is out of range. Only by selectively blocking input from transmitting source nodes we can realistically emulate an out-of-range situation in a broadcast network. The blocking and unblocking is done by inserting or deleting the MAC address of the interface into the input filter of ip6tables. Filtering by IPv6 address was not chosen to perform the blocking and unblocking function because the nature of ad-hoc networks is...
to forward one node’s IPv6 traffic via other nodes. Blocking a particular IPv6 address is therefore incorrect behaviour and would disrupt the emulation of the IPv6 forwarding capability. In addition, nodes cannot be blocked via IPv6 address if they do not have an IPv6 address, as when a node is performing broadcast requests or discovery services prior to obtaining a routable source address.

C. Node Motion Emulation

The motion emulation portion of the emulator has three distinct components integrated into one program. This emulated mobility can be used in ad hoc routing protocol tests and other tests that require location movement. The first part of the emulator is the movement module that provides the location information - represented in terms of longitude and latitude coordinates and movement generation. The second component is the dynamics module that determines the range and potential wireless blockage and provides a dynamic filtering function for neighboring links. The third component is the control module that provides a communications backchannel for the dynamic location information and run-time control commands between the nodes that are involved in testing.

The movement module of the motion emulator is responsible for generating node movement by putting the location information, represented in standardized global positioning system (GPS) format, into shared memory. The GPS format is used because it is compatible with other software tools that use the same shared memory technique for real-world mobile network experiments. In this case, the information is only used to track and report node locations. The movement module is woken up on a periodic polling interval (e.g., once every second) by the control module to post a new location into the shared memory space. The new location is calculated by adding its current location with its current speed on its current heading. The link dynamics module of the emulator is responsible for determining the range between nodes and if any obstacles are between nodes then carry out the blocking action using iptables.

VI. MOBILITY MODEL

Random Way Point (RWP) model is the mostly used mobility model on the mobile multi-hop research area. Hence RWP is chosen as the mobility model for the experiments. A node randomly chooses a destination $D_1$ within the bounded area and moves on a straight line with a speed $V_1$ towards $D_1$. On reaching the destination $D_1$, it waits for a pause time $P_1$. On completion of the pause time, it chooses a new destination $D_2$ within the bounded area. It then moves towards the chosen destination with a specified speed $V_2$. The movement from one destination to another destination is called as the movement transition [9]. This process is continued for the whole experimental duration.

Figure 4 describes the RWP mobility model. In this, a MN starts with a location $S$ at the beginning of the experiment. It then chooses a destination $D_1$ and moves towards it with a speed of $V_1$. On reaching $D_1$, it pauses for time $P_1$. It then moves towards $D_2$ with a speed of $V_2$. On reaching $D_2$, it pauses for time $P_2$. The procedure is repeated till the completion of the experimental period. With reference to the experiments conducted, for every iteration of the experiment, a file containing a pre-recorded mobility model is given as an input to the emulator to determine the movement of the MNs. The input file contains the necessary metrics like the destination, the speed with which the MN should move towards the destination and the pause time.

VII. EXPERIMENTAL CONFIGURATION

During the experiment, each MN moves independently on an square area of unit length $l$ based on the mobility pattern set in the mobility configuration file. Each MN sends UDP packets with a Constant Bit Rate (CBR) to the IGW using multi-hop routing and forwarding. All nodes have the same transmission range $R$. The IGW is located in the centre of the square area. In RWP, speed and pause time affect the connectivity and the routing performance. For reasonable comparison, the parameters are set the same while performing experiments with both AODV and OLSR. The speed of the MNs is set to the specified value for the entire experimentation period and the pause time is set to zero.

The experiment runs in two phases namely the initial phase and the measurement phase. The initial phase is introduced to bring stability to the MN’s mobility patterns. During this phase, no user data packets are sent and no measurements are taken. The experiment runs on an initial phase for a period defined as the stability period. On completion of the stability period, experiment is transferred to the measurement phase. During this phase, the MNs generate user data packets and the measured values are used for performance analysis.

A. Performance Metrics

Performance metrics are defined to evaluate the performance of the connectivity management within the SMN. The following metrics are chosen to evaluate the connectivity mobility

B. Packet Delivery Fraction

The Packet Delivery Fraction (PDF) is the fraction of the packets that are successfully transmitted from the MN to the destination. It is calculated by taking the ratio of the number of packets received at the destination to the number of packets sent by the source. With reference to this paper and the testbed architecture, PDF is calculated by taking the ratio between the number of packets delivered to the IGW and the number of packets sent by the MNs.

C. Packet Delay

Packet delay is defined as the time a transmitted packet takes to reach the IGW from a MN. Average packet delay refers to the average end to end transmission delay calculated
by considering only the successfully received packets. With reference to this paper and testbed, the average packet delay is calculated as the average of the end to end delays of all the packets transmitted by all the MNs reaching the IGW.

VIII. EXPERIMENTAL RESULTS

Figure 5 shows the PDF as a function of the mobility rate. Figure 5 compares the PDF of AODV and OLSR under various mobility rates. From the figure it can be observed that for $V = 5\, m/s$, the PDF of the OLSR is slightly higher than the AODV. However for mobility rates higher than that, the AODV provides better PDF than OLSR. This can be attributed to the buffering capability of the AODV protocol in which the user data packets are buffered while a route to the destination is found. Hence the data packets are not lost while the MN moves or the link connectivity and network topology changes leading to a route discovery to the destination.

Figure 6 shows the average end to end delay with reference to the mobility rate. Figure 6 compares the average end to end packet delay of AODV and OLSR using different mobility rates. From the figure, it is evident that the average end to end delay of the OLSR is lower than the AODV for all mobility conditions. This can be attributed to the proactive nature of the OLSR, in which the routing information is periodically exchanged and packets are forwarded immediately or dropped in case a route to the destination is not found. In contrast to the OLSR, AODV buffers the packet during the route discovery phase. Once a route is found, the AODV forwards the data packets along the discovered route. This increases the end to end delay of the buffered packets thereby increasing the average end to end delay. This explains the lower end to end delay in OLSR compared to AODV. It is essential to note that the performance results obtained in the SMN testbed follows the same pattern as the simulation results analysed in [10].

IX. CONCLUSION

This paper describes the architecture of a SMN to analyse the performance of the multi-hop routing protocols. A testbed was developed to perform experiments and to analyse the characteristics of the SMN. The mobility of the nodes in the testbed and the resulting link connectivity is emulated. The experiments conducted, reveal that the performance of routing protocols depend on the mobility rate, pause time and mobility model. The inherent characteristics of the mobility metrics have a direct influence on the network connectivity and topology, which in turn have an impact on the routing protocol performance. This paper has analysed the performance of two well known ad-hoc routing protocols namely AODV and OLSR in a testbed environment. AODV implementation has the capability to buffer undeliverable data packets which the route discovery is ongoing. However, the buffering of undeliverable data packets is not implemented in OLSR. High mobility rate introduces frequent link and network topology changes. This result is more packet loss with OLSR than AODV. Hence, with respect to PDF, AODV performs better than OLSR for high mobility rates. However, for low mobility rates, the OLSR provides better PDF than the AODV. OLSR offers low end to end delay compared to AODV. Since OLSR doesn’t buffer packets, the packets are forwarded immediately if the route is present or dropped otherwise. Hence the average end to end delay is low. In conclusion, testbed experiments really help to understand the functionalities of the ad-hoc routing protocols in real life scenarios and help in standardisation and benchmarking procedures. The traces and log files obtained with the tested environments can be used for real time analysis.

REFERENCES