Emulating Large-Scale Wireless Networks using ModelNet

Priya Mahadevan, Ken Yocum, and Amin Vahdat
Department of Computer Science
Duke University
{*priya,grant,vahdat*}*@cs.duke.edu*

1. INTRODUCTION

Distributed services for mobile and ad hoc networks are difficult to evaluate due to their scale and the current infrastructure available. Unfortunately, it is difficult to deploy development software at a large number of mobile nodes or to evaluate it under controlled conditions. Simulation is a useful approach, but it abstracts key details of real implementations. It also typically requires rewriting of the application for the simulator.

ModelNet is an emulation environment designed to remove these obstacles. ModelNet emulates a target mobile network on a scalable gigabit LAN cluster, enabling researchers to deploy unmodified software prototypes in a configurable mobile environment and subject them to faults, varying network conditions, different routing protocols and MAC layer implementations. Edge nodes running user specified OS and application software are configured to route their packets through a set of ModelNet core nodes, which cooperate to subject the traffic to the bandwidth, latency, and loss profile of the target network topology.

2. MODELNET ARCHITECTURE

ModelNet was initially developed for testing distributed services for wired networks [1]. The ModelNet architecture comprises of *Edge Nodes* and *Core Nodes*. Figure 1 summarizes the ModelNet architecture. Target applications run on the edge nodes - each application instance on an edge node is a *Virtual Edge Node* (VN) with a unique IP address. The core nodes are equipped with large memories and modified FreeBSD kernels that enable them to emulate the behavior of a configured target network under the offered traffic load. The core nodes, upon receiving packets from the source VN, route the traffic through an emulated network of pipes. Each pipe has a packet queue, bandwidth, latency and loss-rate associated with it. The emulation runs in real time, so packets traverse the emulated network with the same rates, delays and losses as the real network. When a packet exits the chain of pipes, the core transmits the packet to the edge node hosting the destination VN.

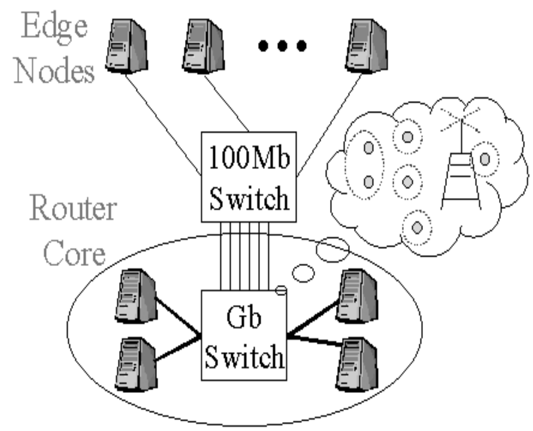

Figure 1: ModelNet Architecture

3. MODELNET USAGE

ModelNet runs in four phases. The first phase *Creation*, generates a user specified network topology (either an Internet or a wireless network topology). A topology is a graph whose edges represent network links and whose nodes represent clients or intermediate nodes in the network. Next, we convert the network topology to a pipe topology, that models the target network. Every link in the graph topology corresponds to a single pipe in the pipe topology. The *Assignment* phase distributes the pipes across the ModelNet core nodes, partitioning the pipe graph to distribute emulation load across the core. The *Binding* phase assigns VNs to the edge nodes and configures them to execute applications. ModelNet multiplexes multiple VNs onto each physical edge node, then binds each physical node to a core. The final *Execution* phase executes target application code on the edge nodes. We have developed simple scripts to automate the entire process.

4. WIRELESS EMULATION

From the ModelNet perspective, there are two key differences between wired and wireless network emulation. First, in the wireless network, the network topology is constantly changing as a result of node mobility. Second, wireless communication is broadcast-based. Thus, any packet transmission typically consumes bandwidth at all nodes within com-

MOBICOM'02 September 23–28, 2002,Atlanta,Georgia,USA

munication range of the sender. We therefore associate a pipe with each node in the target topology. A packet transmitted from node A to node B consumes bandwidth not only at B's pipe but also at all pipes whose associated nodes are within direct communication range of A. To emulate the effects of broadcast, we modify ModelNet to insert dummy packets into the queues of all nodes within range of a transmitting node, capturing the bandwidth sharing inherent to the broadcast medium. The presence of a perfect frequency hopping MAC layer (and a wide enough spectrum) would eliminate the need for emulating bandwidth sharing in this manner.

We summarize here our emulation of ad hoc routing scenarios in the ModelNet core. Our goal is to enable direct comparison of different routing protocols and to quantify the energy vs. performance tradeoffs as well as coordination overhead for different MAC layers and application scenarios. We allow users to specify target movement patterns for the nodes. The ModelNet core maintains a list of neighbors for every node in the topology (per-node proximity information). For our initial implementation, we assume perfect routing, so the core has routes and node positions stored in its cache at all times. We are currently in the process of incorporating different ad-hoc routing protocols within the ModelNet core. On receiving a packet from a VN, the core performs a look up in its route cache. If a route is present, emulation takes place according to the information stored in the route cache. If routes are not present in the cache, ModelNet simulates the generation of route broadcasts and route replies. Error messages are generated when a node moves out of communication range, in which case the core initiates a route recovery mechanism. Note that various routing protocols can be plugged into ModelNet as required. Edge nodes remain unmodified and testing various routing protocols simply requires rewriting the protocol for the core.

Our current implementation assumes the presence of a "perfect" MAC layer that fairly shares available bandwidth among all nodes contending for the medium. Emulating contention at the MAC layer is also under way. We are looking in to the effects of interference from simultaneous transmission in a given region of space and emulating the effects of packets getting garbled, captured, etc. due to collision.

5. INITIAL VALIDATION

We ran the following simple experiment to explore the capacity of ad hoc networks and to validate our implementation of mobility support in ModelNet. While our system supports any movement pattern, we adopted the *waypoint* movement pattern, with 50 nodes randomly distributed across a 670m square. Nodes select a random destination in the space and move there with a velocity of 20 m/s. We varied the node communication range from 50m-250m. We randomly chose 5 pairs of hosts (10 total) from the 50 nodes in our system to perform constant bit rate communication. Each of 5 senders transmitted 108-byte UDP packets (20 bytes IP header, 8 bytes UDP header and 80 bytes of payload) to the receivers at a rate of 200 pkts/second. Each wireless node had a capacity of 1Mb/s with 1 ms transmission latency. Thus, each sender sent data at 168.75 kb/s. We ran each experiment for 300 seconds.

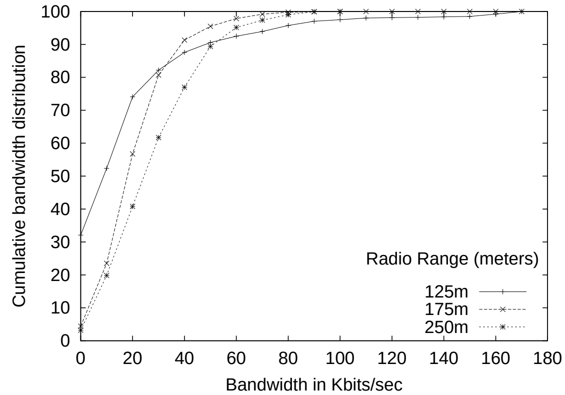

Figure 2: Distribution of bandwidths for 5 pairs of communicating among 50 mobile nodes in an ad hoc wireless environment.

Figure 2 shows the cumulative distribution of bandwidth across all 5 flows measured at 1 second intervals for three different node communication ranges. We average the curves over three runs with different randomly generated movement patterns. At 125m range, the environment is limited by lack of routes between a large number of nodes and hence 32% of the time there is no bandwidth available among a pair of communicating nodes. On the other hand, each node has lesser number of neighbors compared to ranges of 175m and 250m. Thus, nodes that are able to communicate can sometimes obtain the full 168.75 Kb/sec throughput, which never happens for ranges 175m and 250m. For the 175m and 250m ranges, routes are present in all cases, but the medium is often shared because of larger node density, limiting throughput to 50kb/s 90% of the time.

6. CONCLUSIONS AND FUTURE WORK

Our goal is to allow comparisons of various mobile, wireless and ad hoc applications and network scenarios. We have succeeded in emulating ad hoc scenarios and broadcast nature of the medium. Plugging in various routing protocols in the core to conduct a performance evaluation of the various protocols and MAC layer emulation is currently under way.

7. REFERENCES

[1] Amin Vahdat, Ken Yocum, Kevin Walsh, Priya Mahadevan, Dejan Kostic, Jeff Chase, and David Becker. Scalability and Acuuracy in a Large-Scale Network Emulator. *To appear in the proceedings of USENIX OSDI*, Dec 2002.