EMPOWER: A Network Emulator for Wireline and Wireless Networks*

Pei Zheng[1] and *Lionel M. Ni*[1,2]

[1]Department of Computer Science & Engineering
Michigan State University
East Lansing, Michigan, USA

[2]Department of Computer Science
Hong Kong University of Science and Technology
Kowloon, Hong Kong, China

Email: {zhengpei, ni}@cse.msu.edu

***Abstract*— The increasing need of protocol development environments and network performance evaluation tools gives rise to the research of flexible, scalable, and accurate network emulators. The desired network emulator should be able to facilitate the emulation of either wireline or wireless networks. In the case when network topology is critical to the underlying network protocol, the emulator should provide specific mechanisms to emulate network topology. In this paper, we present a distributed network emulation system EMPOWER, which not only can fulfill those requirements, but also can generate user-defined network conditions and traffic dynamics at packet level. EMPOWER is highly scalable in that each emulator node could be configured to emulate multiple network nodes. Some significant research issues such as topology mapping scheme and scalability of the emulator are discussed and addressed. Preliminary emulation results show that EMPOWER is capable of assisting the study of both wireless and wireline network protocols and applications.**

Keywords- network emulation, topology mapping, wireless network

I. INTRODUCTION

Network emulation and simulation are widely used to develop, test, and debug new protocols, to explore and study a specific network-related research issue, or to evaluate the performance of an existing protocol or a scheme. Network simulators such as ns [1] generally provide a rich set of protocol modules and configuration tools that can be easily used to conduct customized simulation experiments. However, the functionalities provided by those supported modules in network simulators are merely logical operations rather than real implementations. Thus a protocol implementation in a network simulator must be modified before being deployed to a target network. Moreover, network simulation will consume a large amount of time when the simulated network is sufficiently large.

Network emulation is the execution of real network protocol implementation code in a controllable and reproducible laboratory network environment. Unlike network simulation, the protocols and applications as well as the interaction between protocols are "real". Network traffic physically traverses the emulation environment, in which underlying protocols are tested and evaluated against user-defined network conditions and traffic dynamics, such as packet latency, link bandwidth, packet drop rate, Bit Error Rate (BER), and link failure.

Most existing network emulators can only provide an environment for end-to-end protocol evaluation since they abstract a network cloud to a simple router with specific packet handling operations [2-4], therefore topology related protocols cannot be evaluated with those emulators. Some emulators employ a simple one-to-one mapping scheme to emulate a target network topology in a local area network. However, when the target network is quite large, the emulation environment will be too costly to establish. In addition to emulate wireline networks, some network emulators [5, 6] are specifically designed to support mobile wireless networks. They typically use some models to emulate a wireless link or a channel. However, they fail to provide means to emulate the mobility of mobile nodes, which is critical to the emulation of a dynamic network topology. Additionally, most wireless network emulators are non-scalable in terms of emulation capacity.

In this paper, we present a flexible and scalable distributed network emulation system called EMPOWER (EMulation the Performance Of WidE aRea networks). Using properly configured commodity computers, EMPOWER is able to emulate large wireline networks with moderate cost. In addition, EMPOWER can be used for some wireless network research by providing mobility emulation of mobile nodes in a wireline network. Arbitrary network topology is mapped to an emulation configuration where predefined network conditions and traffic dynamics can be applied. Each emulator node in EMPOWER can be configured to emulate multiple real network nodes, making the system highly scalable in emulating large networks.

The rest of the paper is organized as follows. Section 2 will present the design of EMPOWER, including the design objectives, EMPOWER architecture, and some important design issues. Preliminary results and the validation of the emulator will be presented in Section 3, followed by a review of related work in Section 4. Finally we conclude the paper with future work in the last section.

*This research was supported in part by NSF grant EIA-9911074.

II. DESIGN OF EMPOWER

A. Design Objectives

Network emulator refers to a workstation or a group of connected workstations that can mimic a target network by imposing predefined network effects to traversing traffic. Network emulators are highly needed in research community when the target network is not accessible, or the target network cannot be used as a test environment. Even if the target network is accessible, one still needs an emulator as a fully controlled reproducible test environment to facilitate protocol development and performance evaluation. Based on the need and the applications of network emulation, the design objectives of a network emulator can be summarized as follows.

First, an emulator must be an open and extensible network environment such that any protocol modules can be loaded into the emulation system. A network emulator is not specifically designed for certain protocols.

Second, an emulator must be scalable in terms of emulation capacity and accuracy. One cannot afford a costly emulation environment with a large number of workstations. Thus an emulator should provide means to reduce the number of workstations required for a specific emulation configuration. Moreover, the emulation accuracy should not be affected when emulating large networks. Otherwise the overhead of the emulator cannot be ignored and its impact must be carefully considered.

Third, an emulator must be able to be flexibly configured and operated for various network topologies and emulation parameters. The physical layout of the emulator environment should not be changed while emulating different target networks. Instead, software modules in the emulator should facilitate the configuration of an emulation experiment.

B. EMPOWER Architecture

As a distributed network emulation system, EMPOWER essentially consists of a number of workstations in a local area network. The workstations can be divided into two categories: emulator node and test node. Each emulator node has been outfitted with multiple 4-port 100Base-TX Network Interface Cards (NICs). Each test node typically has one NIC with one Ethernet port. All the ports connect to fast Ethernet switches. Figure 1 shows an example of the physical layout of EMPOWER. Whatever the target network topology is, the physical layout of EMPOWER will not change. The workstations in EMPOWER are low-cost commodity computers running Linux. Therefore the number of emulator nodes can be increased without high cost.

In order to generate specific network effects that can be applied to traversing network traffic, EMPOWER provides multiple *Virtual Devices* (VDs) as software modules loaded into each emulator node. To emulate network topology, EMPOWER employs an efficient topology mapping scheme called *Virtual-Router mapping*. TCP and UDP Traffic generators are executed on test nodes. Note that in network emulators such as EMPOWER, the kernel of the underlying emulator node will physically handle each traversing packet. With appropriate configurations, any available protocol modules can be loaded into the kernel's network stack without any modification. Furthermore, specific packet level operations could be hooked into a VD.

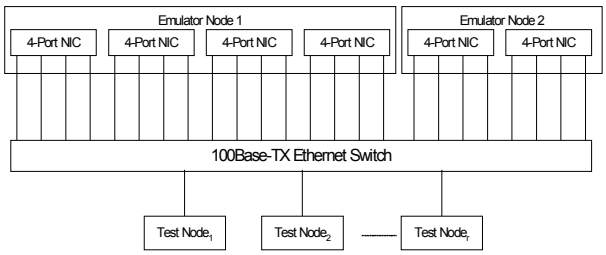

Figure 1. An example of the physical layout of EMPOWER

The basic component of EMPOWER is a VD that intercepts traversing packets and applies predefined network conditions and traffic dynamics. A VD must be attached to a physical network port in an emulator node. They share the same IP address. In the kernel of an emulator node, the routing table has been modified to divert an egress packet flow from the kernel's IP layer to the VD, where specific packet level operations are defined. After that the packet flow will be redirected to the underlying network port and transmitted to the media.

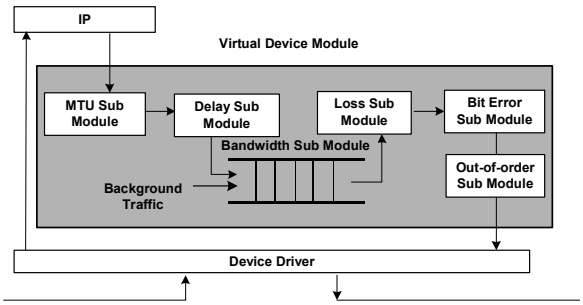

Figure 2. Virtual Device (VD) in EMPOWER

As shown in Figure 2, six sub-modules exist in a VD module: the MTU sub-module, the delay sub-module, the bandwidth sub-module, the loss sub-module, the bit error sub-module, and the out-of-order sub-module. Each sub-module can substantially incur one aspect of IP traffic dynamics and network conditions. In addition, background traffic [7] such as ON/OFF and self-similar traffic can be generated within the emulator node such that the impact of background traffic to the emulation experiment can be studied. The VD module does not affect packet-receiving procedure of an emulator node. Packets entering the underlying network port traverse the network layers of the operating system in its original way. A VD module also supports trace-driven emulation, in which a trace file of real network traffic is used as the input to the emulator instead of traffic generated by some applications. In the EMPOWER prototype, the VD module is implemented as a kernel loadable module in Linux. A GUI interface is also provided to easily configure the network conditions and traffic dynamics and to apply background traffic for each VD module.

Useful traffic statistics can also be viewed on the GUI interface.

In EMPOWER, a router in the target network will be mapped to a virtual router in an emulator node. The router and its mapped virtual router have the same number of interfaces and share the same network configuration such as IP address and routing table. Thus a real network topology with certain number of routers is mapped to a virtual topology with the same number of virtual routers in one or more emulator nodes. Each virtual router is independent of each other in terms of packet forwarding mechanism and routing table maintenance. Even if a packet is sent out from one virtual router to another in the same emulator node, it has to traverse a physical network port and an Ethernet switch, and then be received by another network port. This feature ensures that the packet handling process of EMPOWER closely mimic the same operation in the target network. Otherwise the packet will be directly forwarded between two network ports without going through user-defined packet handling operations and network effects. Figure 3 shows an example of emulation configuration with two emulator nodes generating four virtual routers; each virtual router has four virtual device modules attached to network ports.

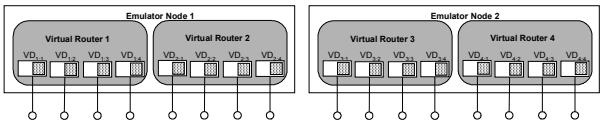

Figure 3. An example of EMPOWER emulation configuration

In out prototype system, there are two types of machines as the emulator nodes. One (*em1*) has an Intel Pentium 4 1.7GHz CPU with 128 MB PC800 RDRAM memory, 32bit 33MHz PCI bus and 400MHz system bus. The other (*em2*) has an AMD K6 300MHz processor with 128MB PC100 SDRAM memory, 33MHz PCI bus and 100MHz system bus. DLink DFE-570TX 4-port Network cards are used as the NICs. Each has four Digital "tulip" 21143 chips and one 21152 PCI-PCI bridge on board.

C. *Virtual Router Mapping*

The virtual-router mapping scheme is proposed to map a target network to an emulation configuration in EMPOWER. The scalability of the whole system in terms of emulation capacity is largely determined by the virtual-router mapping scheme. Given a target network with certain number of routers or nodes in a specific topology, our goal is to find an efficient algorithm that can partition the target network into a number of blocks such that each block can be accurately emulated by an emulator node and the number of emulator nodes is minimal.

An emulator node with multiple network ports is a host-based router. The emulation capability of an emulator node is simply determined by its maximal network throughput, which is in turn determined by a variety of factors such as CPU speed, system interrupt frequency, and PCI bus speed. The maximum network throughput also varies with the number of network ports involved in forwarding packets. In order to find the relationship between these two parameters, we setup a "virtual router chain" with both types of emulator nodes, in which each virtual router has two network ports. The network throughput of the virtual router chain with various number of active network ports is measured. As shown in Figure 4, the more network ports are assigned to virtual routers, the smaller the maximum end-to-end throughput will be. We show in Figure 5 the maximum effective throughput with different number of network ports in an emulator node. The maximum effective throughput is calculated by multiplying the maximum end-to-end throughput by the number of virtual routers in the virtual router chain. We can see that when there are more than 5 virtual routers with 10 network ports in an emulator node, the maximum network throughput drops sharply due to resource competition in the emulator node.

According to the virtual-router mapping scheme, in order to obtain a minimal number of emulator nodes for an emulation configuration, a heuristic best-fit bin-packing algorithm [8] is used. The constraint of traditional bin-packing problem is the bin capacity. In our case, routers or nodes in the target network are the products; emulator nodes with a limited number of network ports and network throughput are the bins. More importantly, the bin size varies due to the change of maximum effective throughput with the number of active ports. Initially, the target network is partitioned into a number of blocks with certain number of routers or nodes. To ensure that each block of the target network can be emulated by an emulator node in terms of network throughput, Effective Emulator Load (EEL) of each block is computed. EEL is defined as the sum of the link bandwidths of each block. It can be regarded as the desired emulated network throughput that an emulator node needs to achieve, which in turn determines the number of virtual routers and network ports that the emulator node can generate. If the number of network ports in a block is larger than the number of network ports corresponding to the EEL in Figure 5, the block has to be repartitioned and the virtual routers have to be generated with more than one emulator node. This procedure will continue until each block meets the network throughput requirement. Using this technique, we are able to considerably alleviate the impact of the resource competition on the network throughput. We have applied this algorithm to a target network with 12 routers. Only three emulator nodes (*em1*) are required to faithfully emulate this network. Note that with one-to-one mapping emulators, 12 workstations are required for the same purpose.

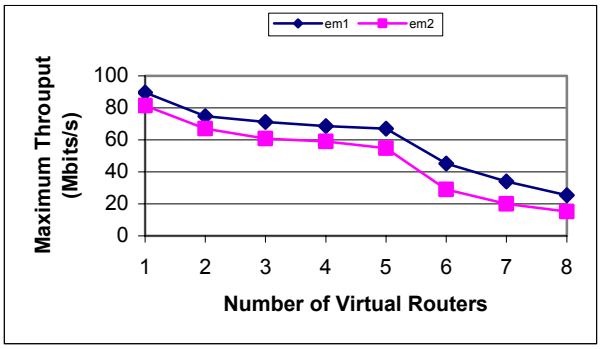

Figure 4. Maximum throughput with various number of virtual routers

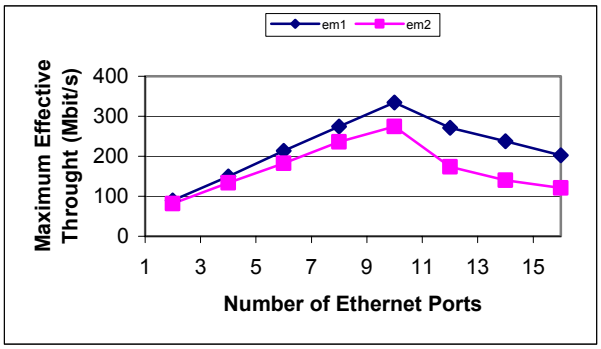

Figure 5. Maximum effective throughput with various number of network ports

D. *Emulation Accuracy and Traffic Models*

Another design issue of EMPOWER is the improvement of emulation accuracy of network effect such as packet delay, link bandwidth, and packet drop rate. Time accuracy is the key to the emulation accuracy of EMPOWER. By default the time accuracy of Linux kernel is 1 jiffy (10 milliseconds) on Intel i386 platforms, which is not acceptable for accurate packet latency emulation. We can increase the system interrupt frequency to obtain a finer-grained time resolution. Since a time accuracy of 1 millisecond is required for specific packet delay emulation, the system interrupt frequency must be at least 1000. The drawback of increasing system interrupt frequency is that it will result in overhead to the performance of the system. Our experiment shows that the modification to the interrupt frequency causes a maximum overhead of 3%, which is acceptable for most applications.

Since multiple routers can be emulated in a single emulator node, system resource competition among those virtual routers in the emulator node may also affect packet latency emulation in EMPOWER. Our result shows that EMPOWER is quite accurate in emulating packet latency with a single virtual router in an emulator node. When there are multiple virtual routers in an emulator node and each virtual router may impose specific packet delays to some packets, the resource competition, specifically the CPU competition, will affect packet latency emulation. To find the overhead of multiple virtual routers in an emulator node, we measure the average end-to-end delay of a UDP flow on *em1* that will traverse a virtual router chain with various numbers of virtual routers. There is no packet latency emulation on each virtual router. We assume the packet forwarding delay of the fast Ethernet switch is constant regardless of the number of port used. Note that even if two adjacent virtual routers are generated by the same emulator node, packets from one virtual router must be first physically transmitted to the Ethernet switch before arriving at the other virtual router. The result shows that when the number of virtual routers increases from 1 to 4, the average packet delay incurred by each virtual router also increases from 15 microseconds to 22 microseconds. If the time accuracy of the emulator node is close to 100 microseconds, the overhead must be considered and compensated in the virtual device module. If the time accuracy of the emulator node is sufficiently larger than the overhead, we may simply ignore this overhead.

To mimic the Internet closely, statistical models are needed to generate background Internet traffic [7]. However, due to the complex nature of Internet traffic, it is rather difficult to create an efficient and effective workload model. Paxson [9] showed that the Poisson model fails for emulating wide area networks. EMPOWER provides several traffic models to generate background traffic. In the packet train model traffic originating from a source is modeled as trains of packets with the inter-train interval comparatively larger than the inter-car interval within a train. The analytical modeling of such sources is done by ON/OFF sources. During the ON periods the source generates packets at a certain rate and remains quiet during the OFF periods. The aggregate Internet traffic is considered to be a superposition of such ON/OFF sources. Taking this model into consideration a facility is provided in the emulator to set up multiple ON/OFF sources. The outputs of these sources are multiplexed together to generate the background traffic. In addition, we incorporate self-similar model into our emulator. Though there are several techniques to generate traffic that is long range dependent, we decide to use the method proposed in [10] due to its simplicity. It is based on the principle that superposition of many ON/OFF sources with strictly alternating ON and OFF periods and whose ON periods or OFF periods exhibit the Noah effect will produce aggregate traffic that exhibits long range dependence. In EMPOWER, we use Pareto distribution to determine the lengths of the ON and OFF periods. As in the packet train model during ON periods fixed size packets are generated at a constant rate. The user is allowed to configure the size of the packets, the rate at which they are generated, and the shape parameter of the Pareto distribution for the ON and OFF periods. In addition to the analytical traffic models discussed above, EMPOWER also provides an empirical method to approximate Internet traffic.

E. *Wireless Network Emulation*

To facilitate the emulation of mobile wireless network, EMPOWER has been designed to support the emulation of wireless networks in a wireline network. Similar to the wireline network emulation, a mobile node in a target network is mapped to a *Virtual Mobile Node* (VMN) in an emulator node, as shown in Figure 6. A VMN physically consists of two network ports, one for ingress traffic and the other for egress traffic. Since a VD only affects egress traffic, one single VD is attached to a network port for egress traffic to impose predefined network effects. Particularly, each VMN has a list of predefined or randomly generated mobility events, which describe the movement of the corresponding mobile node in the target network. Each entry in the mobility event list consists of a time value indicating the time the event fires and the list of neighbors of the mobile node at that point. When a packet is directed to a VMN from IP layer of the emulator node, the corresponding VD will consult the neighbor table to determine if the packet should be forwarded. A similar approach of mobility emulation is described in [5], in which a receiver-side filter program is executed on a central control workstation to determine if an ingress packet should be accepted by a mobile node that connects to the control workstation. Our approach differs from it in that, instead of using a central standalone program that may become the bottleneck of the emulation system, EMPOWER employs a distributed architecture in

which each VMN maintains a packet filter independently. The coordination among multiple emulator nodes is done by clock synchronization using network time protocol. We are in the process of developing another synchronization scheme to improve the scheduling accuracy.

A critical design issue of mobile wireless network emulation in EMPOWER is to make it flexible in emulating different layers of mobile wireless networks. More specifically, EMPOWER should not only provide means to emulate wireless network behaviors beyond network layer (including network layer), but also facilitate the emulation of physical layer and data link layer. It is possible to create a simulated physical layer of wireless network with a decent model. We plan to explore a number of such models and implement a simulated physical layer of wireless network on top of the system's link layer. In particular, most wireless network researches use simulation to evaluate MAC layer protocols such as [11] and [12]. We intend to emulate MAC layer protocols by using IP layer operations in EMPOWER. To achieve this, a special emulated MAC layer (eMAC) has to be added to each VMN. eMAC resides between the VD and network layer. Whenever a packet is about to be sent out from a VMN, eMAC of the VMN will check the channel flag of its current neighbors. If one of its neighbors has a channel flag raised, that neighbor is active in transmitting now. Thus the VMN has to backoff for next try. Once occupying the channel successfully, it will raise the channel flag to indicate the occupation. The channel flag will be reset when the transmission is over. In addition, RTC/CTS can be sent from one VMN to another for a complete channel operation. eMAC is also the place where a backoff algorithm can be incorporated into EMPOWER.

Another major design issue is the hybrid emulation in EMPOWER. A hybrid emulation experiment means the laboratory emulation network includes both wireline and wireless infrastructure and nodes. In addition to workstations in a local area network, mobile devices can also be involved as emulator nodes to make the emulation more accurate and faithful. Additionally, mobile devices can be used as test nodes to generate traffic and measure the performance of an end-to-end protocol.

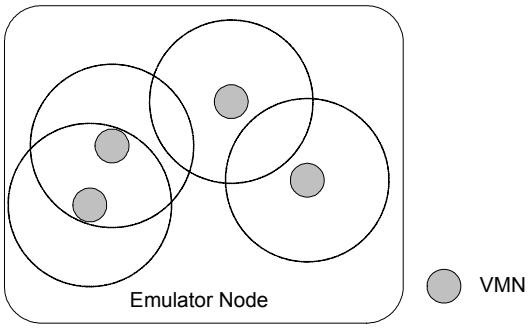

Figure 6. Logic view of an emulator node with four VMNs and the range of each VMN

III. PRELIMINARY RESULTS

A. Network Effect Emulation

In emulating predefined network conditions and traffic dynamics, EMPOWER imposes a set of network effects on each packet passing by such as packet latency, link bandwidth, packet drop rate, bit error rate, MTU change, and out-of-order delivery. The validation of network effect emulation is important for further experiments. The parameter of packet latency emulation can be a fixed value, a user defined pattern, or an empirical delay distribution. The parameter of link bandwidth emulation can be a fixed value to indicate the upper bound of the emulated bandwidth. The parameter of packet drop rate emulation can be a fixed value or a user defined sequence. More details of the validation of network effect emulation can be found in [13].

For the validation of packet latency emulation, we only generate one single virtual router in an emulator node because our intent is to validate the basic component of EMPOWER. The emulator node is the router between the test nodes. The packet latency of the active virtual router is set to a wide range from 1 millisecond to 1 second. (In Figure 7, we only show the packet latency range from 1 millisecond to 60 milliseconds, where the difference between the expected value and measured value is larger than the others.) On one of the test nodes, we send 100 ICMP ECHO REQUEST packets to the other test node, one for every second. To improve the time accuracy of the system, we increase the system's interrupt frequency from 100 to 1024. The resulting time accuracy of the system is 976 microseconds, more than 10 times smaller than the default one. As shown in Figure 7, the measured packet latencies are quite close to the expected values.

To validate link bandwidth emulation, we generate a 4-virtual-router chain in an emulator node. The link bandwidth parameter of one virtual router is set to a range from 100Kbps to 60Mbps to avoid the overhead of resource competition. We send a total volume of 10Mbits UDP packets from one of the test nodes to the other, and calculate the throughput with the measured transmission time. As show in Figure 8, the measured bandwidth is very accurate when the expected bandwidth values are less than 50Mbps. Beyond that, there is noticeable difference since the system is too busy to handle all the network interrupts on all active network ports, as discussed in the previous section. To achieve a higher bandwidth in the virtual router chain, we may partition the virtual router chain into two blocks, and move one block to another emulator node such that the EEL of each block is smaller than the corresponding maximum network throughput of each emulator node shown in Figure 4.

For the validation of packet drop rate emulation, we use a single virtual router to generate specific packet drop rates. We send 500 ICMP ECHO REQUEST packets from one test node to the other, measuring the effective packet drop rate. As shown in Figure 9, the difference between the expected packet drop rate and the measured packet drop rate is very small in less than 8%.

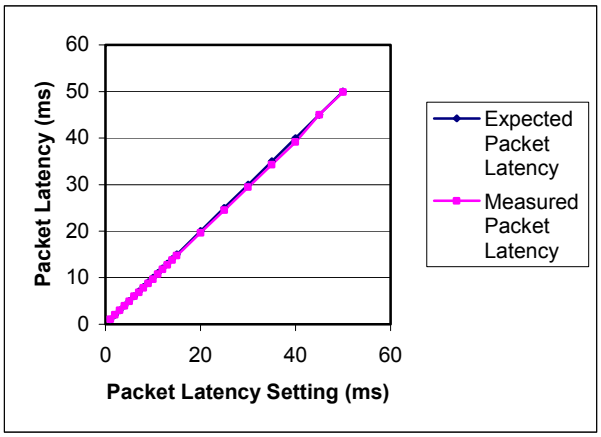

Figure 7. Packet latency emulation validation

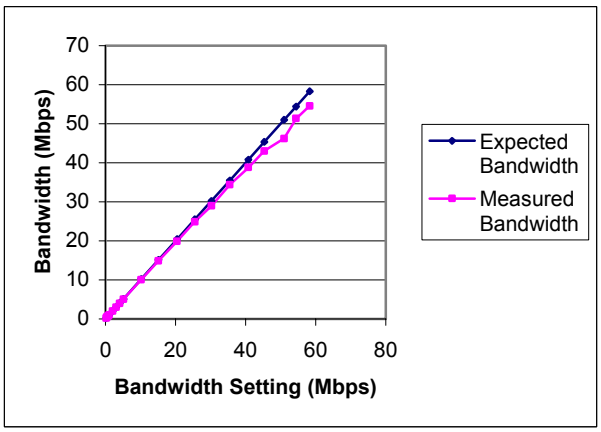

Figure 8. Link bandwidth emulation validation

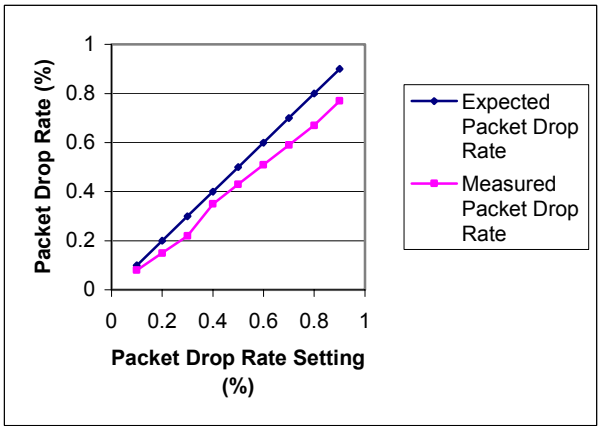

Figure 9. Packet drop rate emulation validation

B. *Multiple Nodes Emulation*

To illustrate how EMPOWER can be used to emulate a network topology, we present a sample network emulation experiment. The sample target network is shown in Figure 10, in which three routers exist. $Router_1$ and $Router_3$ have three physical ports. $Router_2$ has two physical ports. Each physical port has been marked with the last two octets of its IP address. A packet flow is generated between $PP_{12.1}$ on $Router_1$ and $PP_{16.1}$ on $Router_3$. To improve the complexity of the sample emulation scenario, asynchronous routing is used in this case. More specifically, network traffic entering the sample network at physical port $PP_{12.1}$ will be forwarded to $PP_{14.2}$ through $PP_{14.1}$ on $Router_1$; network traffic entering the sample network at $PP_{16.1}$ will be forwarded to $PP_{15.1}$ through $PP_{15.2}$ on $Router_3$.

The sample target network is mapped to an emulation configuration shown in Figure 11. An emulator node (*em1*) with eight network ports can be sufficiently used for the emulation of the target network since the EEL of the target network can be handled by the emulator node with respect to the corresponding maximum effective network throughput in Figure 5. Three virtual routers, VR_1, VR_2, and VR_3 have been generated. Both VR_1 and VR_3 have three network ports; VR_2 has two network ports. Each virtual router has been associated with a routing table. Virtual links between the virtual routers are established by configuring the routing tables of these virtual routers. All the network ports connect to a fast Ethernet switch.

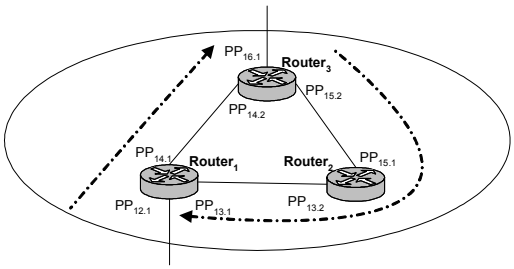

Figure 10. A sample target network

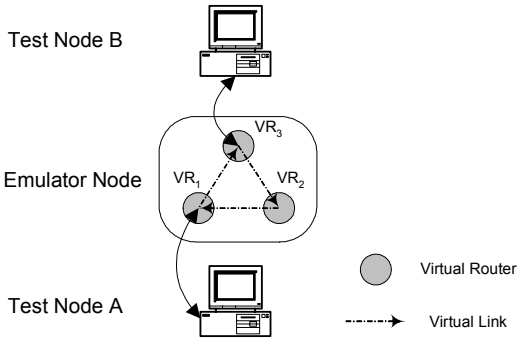

Figure 11. Emulation configuration of the sample target network

To generate network traffic and measure the end-to-end performance, we setup two test nodes, Test Node A and Test Node B. The gateway of Test Node A is the network port for $PP_{12.1}$. The gateway of Test Node B is the network port for $PP_{16.1}$. We use ttcp [14] as the traffic generator on both test nodes. In order to show the flexibility of EMPOWER, we change the network topology shortly after the data transmission begins by removing the unidirectional route of VR_3-VR_2-VR_1 from VR_3's routing table. Thus Test Node A cannot receive any acknowledgement from Test Node B. After several seconds, a new unidirectional route, VR_3-VR_1 for the traffic from Test Node B to Test Node A is established by adding the route to VR_3's routing table. A new unidirectional virtual link

(from VR_3 to VR_1) is generated as well. As a result, data transmission between two test nodes resumes. The time sequence graphs of the TCP connection between two test nodes on each virtual router are shown in Figure 12, Figure 13, and Figure 14, respectively. Note that in this scenario, VR_2 forwards packets from Test Node B to Test Node A for about 3.8 seconds. After that no route is available for packets from Test Node B to Test Node A, until a new route is installed into VR_3's routing table.

In addition to the experiment of topology mapping, we conduct another experiment on the emulation of network conditions and traffic dynamics. The virtual links in the sample emulation are configured to generate specific propagation latency and emulated bandwidth, as shown in Table I. We send 60 ICMP packets from Test Node A to Test Node B using "ping" program, measuring the round trip time of each packet. Similar to the previous experiment, the unidirectional route of VR_3-VR_2-VR_1 is removed at some time during the experiment, and a new unidirectional route of VR_3-VR_1 is added after some period of time. Figure 15 shows the round trip time of each ICMP packet during the experiment. There are three stages in this graph. Packets in Stage 1 have an average round trip time of 7 milliseconds. No response is measured in Stage 2. Packets in Stage 3 have an average round trip time of 5 milliseconds. Thus the route change in the sample network is faithfully emulated in the emulation experiment.

Our sample experiments show that the prototype of EMPOWER is capable of mapping a network topology to an emulation configuration. Performance analysis, measurement, and traffic monitoring can be done on each virtual router to provide more insight of the underlying protocols and applications.

As an open and extensible framework for network emulation, EMPOWER can be used in many areas of network research. Fox example, in order to examine the impact of non-DS domains in a multi-domain DiffServ network, EMPOWER has been configured for a domain chain emulation experiment [15], in which five routers are generated and virtually connected in two emulator nodes. Routing protocols such as RIP and BGP can also be evaluated in a large-scale emulated network with EMPOWER.

TABLE I. EMULATION SETTINGS FOR THE SAMPLE TARGET NETWORK

Virtual Link	Propagation Latency (millisecond)	Bandwidth (Mbps)
VR_1->VR_3	3	50
VR_3->VR_2	2	100 (default)
VR_2->VR_1	2	100 (default)
VR_3->VR_1	2	25

C. *Wireless Network Emulation*

To demonstrate EMPOWER's support for mobile wireless network emulation, we now introduce a sample emulation of a multi-hop mobile ad hoc network. The sample wireless network consists of four mobile nodes, A, B, C and D, as shown in Figure 16. A and C are handheld devices that do not move in this scenario, while B and D are laptop computers that move in a predefined manner. The mobility information of this scenario is shown in Table II.

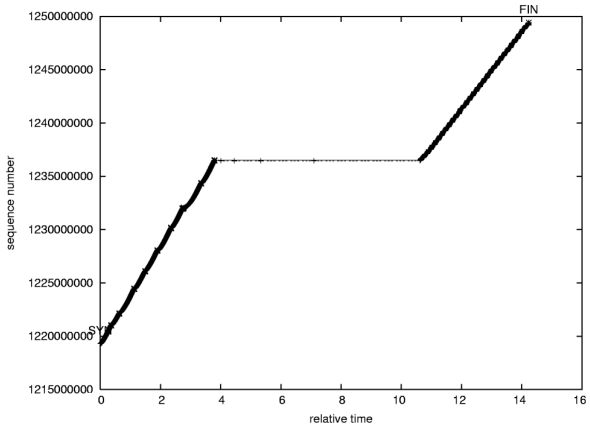

Figure 12. Time sequence graph on VR_1

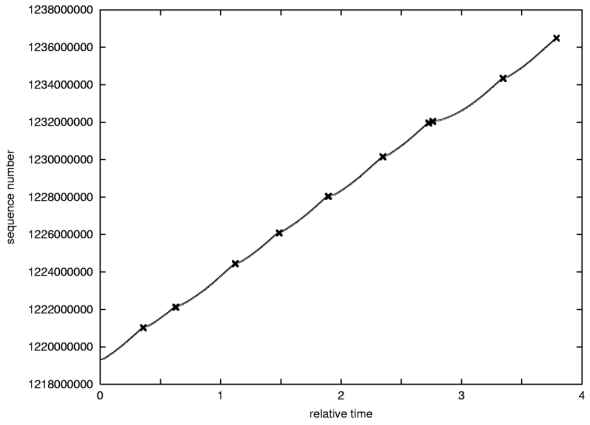

Figure 13. Time sequence graph on VR_2

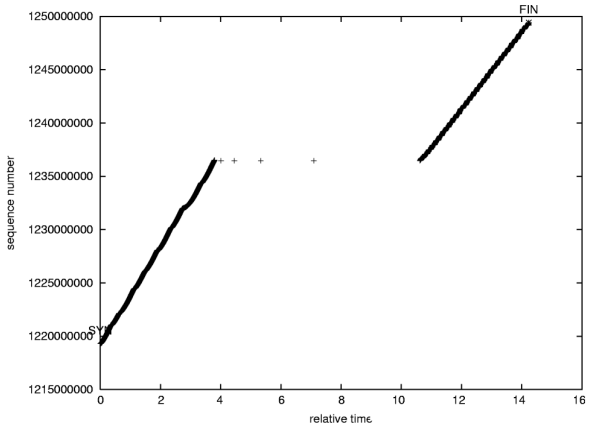

Figure 14. Time sequence graph on VR_3

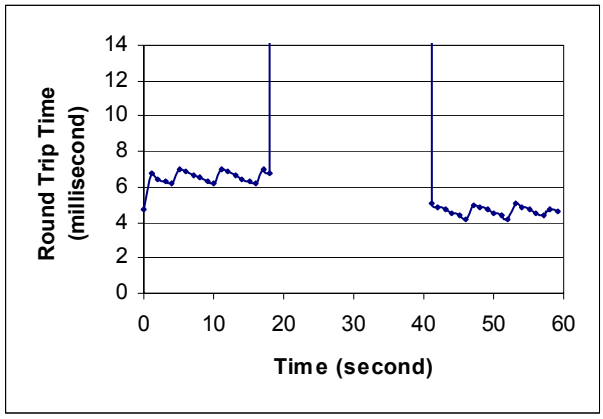

Figure 15. Round trip time of ICMP packets in the sample target network

The origin of the 2-dimension coordinates is at the position of A. The distance between A and C is 40 feet; B and D are in the middle perpendicular to A and C. Each mobile node has a transmission range of 25 feet. The moving speed of B and D is 1 foot per second. Note that the transmission range and moving speed can be any value. We choose these values to simplify the illustration of mobility emulation. Whatever those scenario parameters are, EMPOWER will use them to compute the link event list of each node and schedule those events for VMNs to reflect a dynamic topology. In this scenario, A and C cannot communicate with each other directly because of the distance between them. Initially B performs as a router between A and C; then B moves upwards and is out of the transmission range of A and C. As a result, the connection between A and C is lost. Later D moves to B's original position, and the connection between A and C resumes. Using EMPOWER, we do not need to have two persons holding handheld devices and moving in a defined manner. More importantly, we are able to impose any network effect on those emulated links, which is highly desirable for the research of wireless protocols.

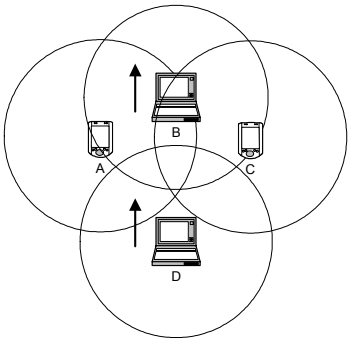

Figure 16. A sample wireless network

The emulation settings of this sample network are shown in Table III. We choose these values to demonstrate EMPOWER's functionality in emulating network conditions and traffic dynamics. In fact user can define any packet latency larger than 1 millisecond, and any emulated bandwidth that the emulator can support. Another way to obtain these parameters is to first setup a static wireless network testbed according to the emulation scenario, and then to conduct measurement of related performance metrics in the testbed.

TABLE II. MOBILITY INFORMATION OF THE SAMPLE SCENARIO

Node	Time (second)			
	0	***15***	***18***	***33***
A	(0,0)			
B	(20, 0)	(20, 15)		
C	(40,0)			
D	(20, -18)	(20, -18)	(20, -15)	(20, 0)

TABLE III. EMULATION SETTINGS FOR THE SAMPLE WIRELESS NETWORK

Link	Latency (ms)	Bandwidth (Mbps)
A->B	3	10
B->A	2	10
B->C	2	10
C->B	<1	10
A->D	3	10
D->A	2	5
C->D	<1	10
D->C	4	5

Four VMNs, VMN_A, VMN_B, VMN_C and VMN_D, are generated and properly configured according to the information in the emulation configuration file. One emulator node is sufficient to generate those VMNs, as shown in Figure 17. The mobility information in this scenario is pre-defined before the emulation experiment. The routing table of each VMN has to be modified during the experiment due to the changes in the neighbor table. For example, at the time when D is about to enter the transmission range of A and C, the routing tables of VMNs for A and C must be modified to reflect this topological change. We plan to implement some proposed multi-hop ad hoc routing protocols in EMPOWER such that the installation and removal of a route for a VMN can be done automatically.

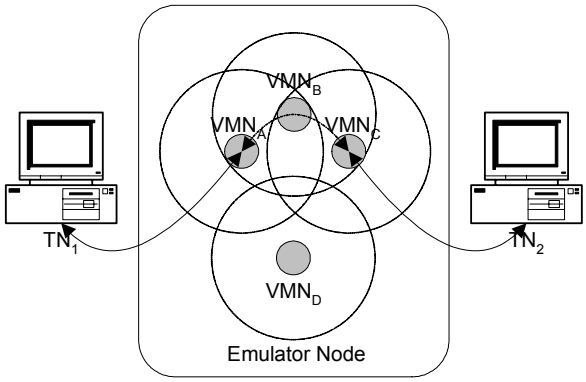

Figure 17. Emulation configuration of the sample wireless network

First, to verify the mobility event scheduling of each VMN, a number of ICMP ECHO REQUEST packets are sent from

TN_1 to TN_2. The round trip time of each packet is shown in Figure 18. In the first 15 seconds while B performs as the router between A and C, the round trip times are on an average of 7 milliseconds. After that, there is no connection between A and C for 3 seconds. When D moves within the range of A and C and becomes the router, the round trip times are on an average of 9 milliseconds. Note that the time accuracy in EMPOWER is almost 1 millisecond.

To explore the impact of node mobility on end-to-end performance in this scenario, we conduct an FTP file transfer operation between TN_1 and TN_2. The time-sequence graph in Figure 19 suggests that the ftp data transfer continues for about 11.5 seconds before the connection between A and C is lost. Then after three seconds the ftp data transfer resumes. Thus all the network events in this scenario are accurately emulated by EMPOWER with only one emulator node. This emulation experiment demonstrates that EMPOWER provides an alternate for the evaluation of various TCP flavors in wireless networks.

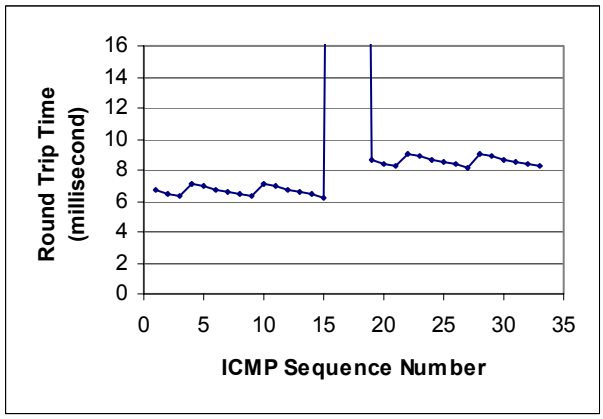

Figure 18. Round trip time of ICMP packets in the sample wireless network

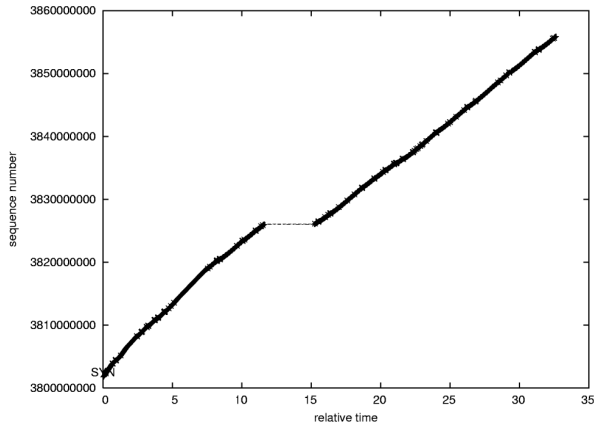

Figure 19. Time sequence graph of VMN_A

IV. Related Work

Most existing network emulators such as ONE [2], ENDE [3], and NIST NET [4] simply abstract a target network to a "gateway" with a set of network parameters. They do not provide a mechanism to emulate network topology. Some topology-aware emulators such as WAN emulator [16] and Emulab [17] employ a one-to-one mapping to map a target network to a number of workstations. However, when the target network is quite large, a costly emulation system with the same number of workstations as the target network must be available. Unlike those network emulators, EMPOWER is designed to emulate the topology of a target network with a significant less number of workstations than the target network due to the unique virtual-router mapping scheme.

Mobile wireless network emulators such as Seawind [6] allow one to define multiple sets of parameters that are constantly changing during an emulation experiment. Additionally, specific channel models are provided to closely mimic a unidirectional wireless link. In JEmu [5], the emulator program works as a router of all the mobile nodes in the system and determines if the forwarding message will cause a collision on the receiver side. These central-control wireless network emulators are easy to configure, and their parameter sets are usually sufficient for the performance evaluation of end-to-end network protocols. However, this approach fails to provide a test environment for some topology-related protocols such as multi-hop ad hoc routing protocols, which is now highly active in mobile network research community. Trace-based emulation [18, 19] is particularly useful because wireless network effects that it reproduces originate from real network traces. However, similar to the central-control approach, the trace-based approach can only provide a reproducible environment for the performance evaluation of end-to-end protocols and applications. Additionally, since each mobile node in the target network should be traced for an emulation experiment, this approach cannot emulate a large-scale network because of the high cost in collecting wireless network traces.

The emulation facility in VINT/ns [20] is developed to integrate ns simulation and network emulation such that the rich resources available in ns such as protocol modules, algorithms, and visualization tools, can be used for the purpose of network emulation. However, this approach inherits the limitation of network simulation: real protocol implementations and applications cannot be evaluated without conversions, and simulation assumptions might strongly affect the simulation results.

V. Conclusions and Future Work

A scalable and flexible network emulation system is highly needed for the research of network protocols, systems, and applications. In particular, network topology should be emulated such that topology-related network protocols could be tested and evaluated against emulated network conditions and traffic dynamics. In this paper, we have presented our distributed network emulation system EMPOWER, which provides such an emulation environment with a number of commodity computers as the infrastructure and software modules to map a target network to an emulation configuration and impose predefined network conditions and traffic dynamics to traversing traffic. Each emulator node in EMPOWER can possibly emulate multiple virtual routers, making the whole emulation system highly scalable in terms of emulation capacity. The resource competition problem of EMPOWER

can be substantially solved by the virtual-router mapping scheme, in which the maximum number of active network ports in an emulator node can be determined and the target network is then partitioned accordingly to avoid resource competition in an emulator node. In addition, EMPOWER provides a mechanism to emulate the mobility of a wireless network in a wireline network, thus mobile ad hoc networks can be emulated in EMPOWER. Our preliminary results show that EMPOWER is quite accurate in emulating network effects and network topology, as well as node mobility in wireless networks.

The problem of bandwidth emulation scalability remains an open research issue in EMPOWER. Ethernet channel bonding or Gigabit Ethernet might be used to provide a higher bandwidth emulation infrastructure. However, this will in turn reduce the number of virtual routers in an emulator node and bring another factor of resource competition to the system. We need to investigate the impact of those solutions to the bandwidth scalability problem. In addition, we plan to incorporate hybrid emulation and a simulated physical layer of wireless network into EMPOWER such that the wireless emulation facility of EMPOWER can be enhanced to support the emulation of physical layer and data link layer. We are also in the process of using EMPOWER to explore a variety of research issues about routing protocols and QoS provisioning in both wireless and wireline networks.

ACKNOWLEDGMENT

The authors would like to thank Kaushik K. Dam and Manqing Huang for their early work in the standalone emulator.

REFERENCES

[1] "The network simulator: Ns-2," http://www.isi.edu/nsnam/ns/

[2] M. Allman, A. Caldwell, and S. Ostermann, "One: The ohio network emulator," Ohio University, Tech Report TR-19972, August 1997.

[3] I. Yeom and A. L. N. Reddy, "ENDE: An end-to-end network delay emulator," Department of Electrical Engineering, Master's thesis, Texas A&M University, 1998

[4] "NIST net," http://snad.ncsl.nist.gov/itg/nistnet/

[5] J. Flynn, H. Tewari, and D. O'Mahony, "Jemu: A real time emulation system for mobile ad hoc networks," In *Proceedings of the Firsit Joint IEI/IEE Symposium on Telecommunications Systems Research*, Dublin, Ireland, November 2001.

[6] M. Kojo, A. Gurtov, J. Manner, P. Sarolahti, T. Alanko, and K. Raatikainen, "Seawind: A wireless network emulator," In *Proceedings of 11th GI/ITG Conference on Measuring, Modeling and Evaluation of Computer and Communication Systems(MMB 2001)*, RWTH Aachen, Germany, 2001.

[7] W. S. Cleveland and D. X. Sun, "Internet traffic data," *Journal of the American Statistical Association*, vol. 95, pp. 979-985, 2000.

[8] J. E. G. Coffman, M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: A survey," in *Approximation algorithms for np-hard problems*, D. Hochbaum, Ed. Boston: PWS Publishing, 1997.

[9] V. Paxson and S. Floyd, "Wide area traffic: The failure of Poisson modeling," *IEEE/ACM Transaction on Networking*, vol. 3, pp. 226-244, 1995.

[10] W. Willinger, M. Taqqu, R. Sherman, and D. Wilson, "Self-similarity through high variability: Statistical analysis of Ethernet LAN traffic at the source level," *IEEE/ACM Transaction on Networking*, vol. 5, 1997.

[11] S. S. Kang and M. W. Mutka, "Provisioning service differentiation in ad hoc networks by the modification of backoff algorithm," In *Proceedings of Int'l Conference on Computer Communication and Network (ICCCN'01)*, Scottsdale, Arizona, October 2001.

[12] B. Bensaou, Y. Wang, and C. C. Ko, "Fair media access in 802.11 based wireless ad-hoc networks," In *Proceedings of Mobile and Ad Hoc Networking and Computing*, 2000.

[13] P. Zheng and L. M. Ni, "EMPOWER: A scalable framework for network emulation," In *Proceedings of the 2002 International Conference on Parallel Processing (ICPP'02)*, Vancouver, Canada, 2002.

[14] "Ttcp," http://www.cisco.com/warp/public/471/ttcp.html

[15] P. Zheng and L. M. Ni, "The impact of non-DS domains in a multi-domain DiffServ network," In *Proceedings of The 11th International Conference on Computer Communications and Networks (ICCCN'02)*, Miami, Florida, 2002.

[16] J. S. Ahn, P. B. Danzig, Z. Liu, and L. Yan, "Evaluation of TCP vegas: Emulation and experiment," In *Proceedings of SIGCOMM*, 1995.

[17] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," In *Proceedings of 5th Symposium on Operating Systems Design and Implementation (OSDI'02)*, 2002.

[18] B. D. Noble, M. Satyanarayanan, G. Nguyen, and R. Katz, "Trace-based mobile network emulation," In *Proceedings of Proceedings of SIGCOMM'97*, September 1997.

[19] M. Satyanarayanan and B. Noble, "The role of trace modulation in building mobile computing systems," In *Proceedings of the 6th Workshop on Hot Topics in Operating Systems*, Cape Cod, Maryland, May 1997.

[20] K. Fall, "Network emulation in the VINT/ns simulator," In *Proceedings of 4th IEEE Symposium on Computers and Communications*, July 1999.