Published in the Proceedings of the IEEE Wireless and Optical Communications, WiOPT05
3rd IEEE Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks
Riva Del Grande, Italy, April, 2005, pp.219-228

AN INTERACTIVE TRANSPARENT PROTOCOL FOR CONNECTION ORIENTED MOBILITYITY–PERFORMANCE ANALYSIS WITH VOICE TRAFFIC

Raid Y. Zaghal, Sandeep Davu and Javed I. Khan

Networking and Media Communications Research Laboratories
Computer Science Dept., Kent State University
233 MSB, Kent, OH 44242
{rzaghal, javed, sdavo}@cs.kent.edu

Abstract*—Loss-Free handoff in Mobile Networks is an extensive research area. Mobile IP (MIP) provided a solution to enable a mobile node to roam from one location to another while maintaining its network level connectivity. However, handoff latencies and longer triangular routing paths in MIP can severely degrade communication performance and in particular cripple connection oriented protocols like TCP. In this paper we propose an alternate approach for robust mobility. The scheme is based on the principle of 'Interactive Transparent Networking' where all networking layers remain lightweight but are engineered for interactivity. This would allow principle intelligent actions to be performed at the application layer. With protocol interactivity we demonstrate a novel scheme that switches IP address in the TCP/IP stack on both end-points and perform loss-free rapid handoff. The scheme offers not only loss-free handoff, but also offers several fundamental system advantages; (i) it does not impose any changes on original network protocols or their dynamics, and (ii) it fully adheres to the end-to-end principle and do not require intermediary nodes as in MIP. We have achieved a real implementation of the scheme on FreeBSD and tested the real system over Internet with voice traffic. We show that this scheme can dramatically reduce handoff latency and improve TCP performance by offering shorter routes with loss-free handoffs and smooth, low-jitter voice stream.*

Keywords: *Mobile IP, handoff latency, triangular routing, Interactive protocol.*

I. INTRODUCTION

In a typical wireless environment, if a mobile node (MN) roams from one access point to another within the same IP subnet; it only needs to perform link-layer (L2) handoffs in order to maintain its wireless connectivity. These L2 handoffs remain transparent to the IP layer (L3). However, if the MN migrates to a different IP network, its current IP address becomes topologically invalid and it must reconfigure a new IP address from the newly visited network –i.e. it must also perform L3 handoff. In this latter case, L3 handoff should also remain transparent to upper layers. For example, if the MN in figure-1 moves from AP1 to AP2 –it is roaming within subnet1—it only needs to perform L2 handoff. However, if it moves from AP2 to AP3 it must perform L3 handoff and configure a new IP from subnet2 since AP3 resides in a different IP subnet.

If a MN performs handoff it acquires a new IP from the new access point which renders all the existing TCP/IP connections useless. Since classic IP was originally designed without any mobility support, its routing mechanism relies on IP address semantics to deliver data to the correct destination.

Mobile IP (MIP) [14] provided a global solution to this problem by introducing indirection through a set of Mobility Agents. In MIP, each MN is identified by an IP address assigned to it by its home network –called *home address*—regardless of its current point of attachment. MIP introduced three new entities, namely

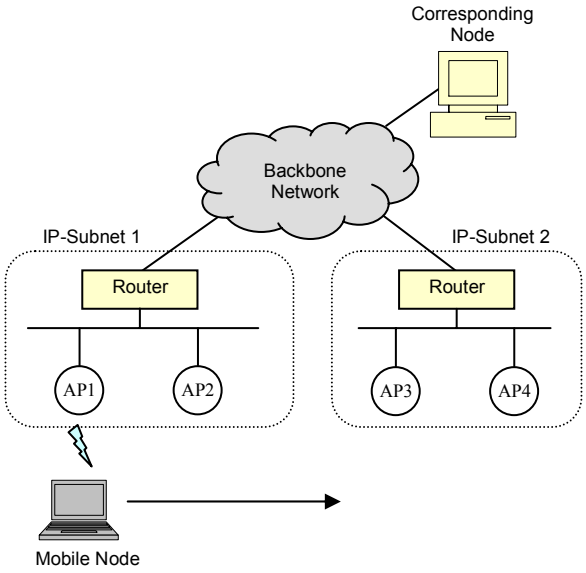

Figure-1. Only when a MN migrates from one IP subnet to another it must perform L3 handoff.

Published in the Proceedings of the IEEE Wireless and Optical Communications, WiOPT05
3rd IEEE Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks
Riva Del Grande, Italy, April, 2005, pp.219-228

the *home agent*, the *foreign agent* and the *MN*. Whenever a MN performs L3 handoff, it must register its current point of attachment with the home agent. For every registered MN, the home agent intercepts all incoming traffic from a given sender –usually referred to as the *corresponding node*—and redirects it through tunneling (packet encapsulation) to the MN's most recently registered location. As a sender, the MN can bypass the home agent and transmit packets directly to the corresponding node. This kind of traffic flow is referred to in the literature as *triangular routing*.

In MIP, foreign agents periodically broadcast Agent Advertisements to detect MN movement. When the MN decides to migrate to a new network, it configures a new care-of-address, and then it registers this address with the home agent. The home agent updates its address binding cache and sends an ACK to the MN. Communication between the two endpoints cannot resume until registration is completed at the home agent. This interruption affects transport level performance – e.g. may cause TCP retransmission timeouts—and may cause severe quality degradation especially for time-sensitive (audio/video) applications.

In this paper we propose IPMN (*Interactive Protocol for Mobile Networks*) for network level mobility support which diverts from the MIP approach. The scheme is based on the new paradigm of *Interactive Transparent Networking* [10] –which we have investigated recently—and enables the corresponding node to send packets directly to the MN and therefore eliminates the need for a home agent and triangular routing. The Interactive Transparent Networking paradigm calls for meta-engineering of existing protocol to add interactivity –protocol end-point components are reengineered to create an interactive version of a protocol with added handles and API that facilitates event passing and access to its internal states. The paradigm complies with the following three principles of backward compatibility:

1) The interactive version of a protocol remains functionally compatible with legacy non-interactive versions.
2) The API is an extended set, and thus classical applications remains fully usable with the interactive versions of the end-point components.
3) Interactivity –if used—does not change the network side dynamics of the original protocol and thus the dynamics of the network.

The paper is organized as follows: in section 2 we discuss MIP handoff latency and related work. In section 3 we briefly present the Interactive Transparent Networking paradigm. In section 4 we discuss IPMN and in section 5 we present experiment details and performance results. We give concluding remarks in section 6.

II. MOBILE IP HANDOFF LATENCY

Handoff in Mobile IP goes through a three-stage life cycle (i) L2 handoff (e.g. IEEE802.11 handoff), (ii) movement detection, and (iii) address registration. Normally, the three stages do not overlap and should occur in the same order. Therefore, the summation of their latencies constitutes the total handoff latency. If we can reduce the latency of any one of these stages –or better, if we can eliminate a stage—we can improve the total handoff latency.

The original MIP proposal maintained a clean separation between link-layer (L2) and network layer (L3), which affected performance adversely; whenever L2 performs handoff, L3 is not notified. L3 then has to resort to *agent advertisements* to discover movement and initiate its own handoff procedure. Movement detection algorithms that rely on Agent advertisements are called *advertisement* based algorithms. These are the *Lazy Cell Switching* (LCS) [14], and *Eager Cell Switching* (ECS) [15]. In [4], Fikouras, et al, introduced a *hinted* based movement detection algorithm called *Fast Hinted Cell Switching* (FHCS) which allowed L2 to send '*triggers*' to L3 whenever a handoff event is initiated. FHCS –which is consistent with our event-based scheme—was able to significantly reduce handoff latency by negating the need for movement detection and agent selection. Also, Researchers aimed at reducing registration signaling delay by introducing a hierarchical structure and therefore allowing regional registration and reduced round trip delay. Among these are [2] who introduced Micro-mobility by dividing the network in a hierarchical structure and [7] who allowed the MN to perform local registrations in the visited domain in addition to home registration. [8] And [16] can be also classified in the same category; the former suggested a network infrastructure which covers a large

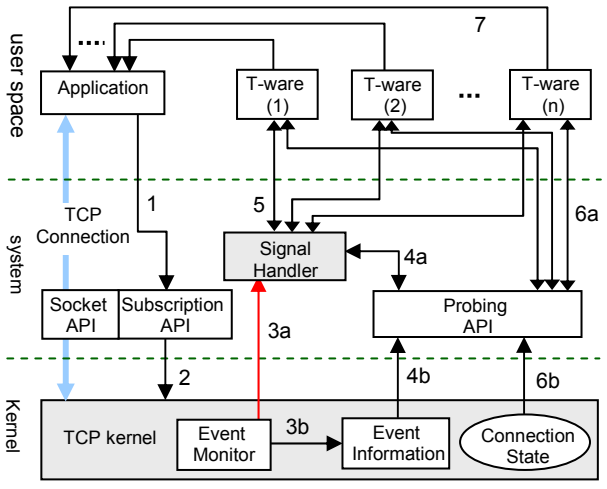

Figure-2. TCP interactive extension and API.

geographical area with L2 switches, and the latter used proprietary control messages for MN location management and routing within a regional area. Other proposals took a different direction by suggesting a deployment scheme of MIP based on existing infrastructure. The RAT (Reverse Address Translation) architecture [17] is based on the network address translation (NAT) protocol. It uses a RAT device to support IP mobility by providing packet re-direction service between the correspondent node and the MN.

The above proposed solutions though have promised advantages over classical MIP, unfortunately, they require new infrastructure and/or considerable network level modifications. Thus, they are yet to see deployment!

III. INTERACTIVE TRANSPARENT NETWORKING

During the past few years, we have been developing a new paradigm called *Interactive Transparent Networking*. It provides a framework for researchers to deploy protocol solutions/modifications without embedding custom codes within the network layer itself. Instead, we propose using inter-layer *interactivity* and state *transparency* to be able to perform these solutions at the upper layers –e.g. at the application layer. In this section we briefly define the new paradigm; more information can be found in our previous work in [10] and [11].

Within the interactive-transparent framework, we provide means to allow programs in upper layers to subscribe with protocols in lower layers to be notified when certain events (state transitions) occur. Subscribers can then pull up the required service state information, perform the actual action by programmable components running in the upper layer –called *Transientware* modules—and then create handles to push down the generated actions (state modifications) into the target network layer. A subscriber application should be interested in certain events that might occur in the target protocol, and by subscribing to events, the subscriber wishes to be notified when any one of the events has occurred. In figure-2 we show the general architecture of a TCP-based interactive protocol. Upon opening the socket, an adaptive application may bind a Transientware module to a designated TCP event by subscribing with the kernel. This is represented by arrows 1 and 2 in figure-2. The binding is optional; if the application chooses not to subscribe, the system defaults to the silent mode identical to TCP classic. When the event occurs in TCP, the kerenl sends a signal to the upper layers (3a) and at the same time it saves the event information (3b). A special handler catches the signal and probes the kernel for the event type (4a, 4b), The handler then invokes the appropriate Transientware module to serve the event (5),. A Transientware module can also use the probing API to access the kerenl state (6a, 6b) or to pass some information to the subscriber application itself (7).

IV. INTERACTIVE PROTOCOL FOR MOBILE NETWORKS (IPMN)

A. The Scheme

We employed the Interactive Framework [11] to design an alternative solution for the address binding, registration, and tunneling in MIP. The basic idea of our scheme is to enable the MN to obtain a new IP from the future FA before handoff is performed, replace the 'source IP' field in the TCP/IP stack of the MN with the new IP, and relay the new IP to the fixed host (FH). Once it receives the new IP, the FH immediately switches to the new IP by replacing the '*destination IP*' field in the TCP/IP stack with the new IP. A best case scenario for this scheme would happen if the MN can locate the new Access Router and obtain a new IP address, e.g., through a DHCP server before loosing connection with the current Access Router. Once it obtains its new IP address, the MN proceeds with L3 handoff as follows:

1. *Freeze the TCP connection by advertising a zero window to the FH.*
2. *Perform actual L3 handoff by replacing the IP fields in the TCP/IP stack at both the MN and the FH with the new IP address.*

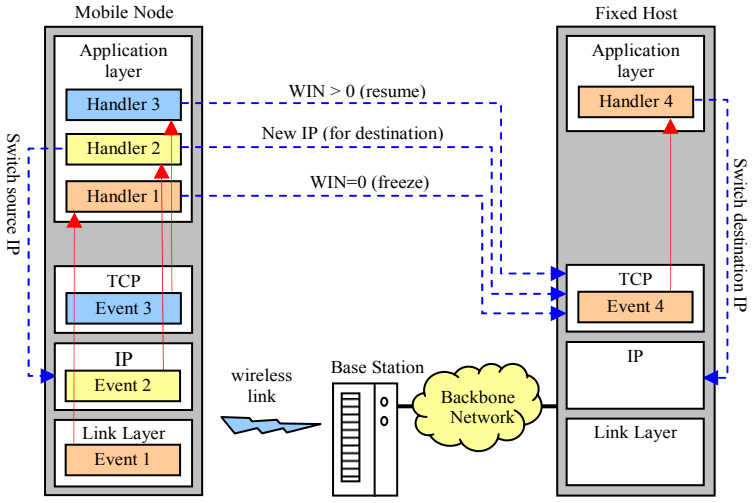

Figure-3. Architecture of the full interactive mode of IPMN.

TABLE-1. IPMN EVENTS AND THEIR HANDLING

Node	Event	Layer	Event tracked	Action taken by event handler
Mobile Node	1	LL	L2 handoff has been initiated.	Advertises a zero window to the FH. The freeze mechanism of TCP will force the FH to stop transmission.
	2	IP	A new IP has been assigned to the MN from the future BS.	Call the `switch_ip()` system call. This will replace the source IP filed in the IP header of the MN with the new IP and will send a segment to the FH with TCP option = SWITCH_IP to replace the destination IP field on the FH.
	3	TCP	The 'SWITCH_IP' segment has been ACKed.	Advertises a non-zero window to the FH. This will unfreeze the connection and enable the FH to resume transmission.
Fixed Host	4	TCP	A special TCP segment received with TCP option=SWITCH_IP.	Strip the new IP number from the options part of the segment, then call the `switch_IP()` system call which stores the new IP in the destination IP field of the IP header overwriting the old IP number.

TABLE-2. API SET OF IPMN

System Call	Usage
Subscribe(evt)	Subscribes with a network layer protocol for event (evt).
GetEventInfo()	Used by a global *signal handlers* which catches all signals from the kernel to probe the kernel and retrieve event information and then activate appropriate event handler (Transientware module).
Relay_IP(IP_addr)	Let TCP transmit a special segment carrying the new IP to the other end.
Switch_Source_IP(IP_addr)	Changes the source IP address in local TCP/IP stack. Used by MN only.
Switch_Dest_IP(IP_addr)	Changes the destination IP address in local TCP/IP stack. Used by FH only.
Freeze_TCP()	Advertise a zero window to the other end (the FH) to force it to halt transmission.
Resume_TCP()	Advertise a non-zero window to the other end (the FH) to allow it to resume transmission.

3. Wakeup TCP by advertising a nonzero window to the FH.

Handoff pre-processing, i.e., locating a future Access Router and obtaining a new IP address, can also be done at the application level prior to L2 handoff. Fortunately, since we allow protocol interactivity, we can configure L2 to send an early signal the application layer about an impeding handoff. This gives the application layer a grace period to do all the bookkeeping while it is still connected through the current Access Router. Naturally, a simple application level IP-lookup module should perform the task. We can benefit from interactivity again by allowing this IP-lookup module to probe L2 for the identity of the next Access Router (e.g., its IP address). Then, this module can contact the router and obtain its next IP address via a DHCP attached scheme. A number of previous works like [4], [17], and [21], have shown excellent schemes that can support this methodology. We can re-model these schemes –or some aspects of them—with the interactive paradigm to implement the handoff pre-processing illustrated above. Furthermore, we believe that since the MN can obtain a new IP before handoff, this pre-processing will not impact handoff latency.

The purpose of this current implementation is to experiment the basic idea of physically changing the IP number at both end-points whenever the MN configures a new IP address. Therefore, this version of IPMN, only implements the three-step L3 handoff procedure shown above. In the future, we plan to augment IPMN to include the handoff pre-processing as well.

B. The Architecture

We track three events at the MN and one event at the fixed host (FH). Figure-3 describes the conceptual architecture, and Table-1 describes the corresponding events and their handling sequences at each endpoint. At the MN, when the link layer detects signal fading and initiates L2 handoff (event 1), it signals the subscribing application. When the event is received at the application layer, a Transientware module (handler 1) is activated immediately; this module simply makes a simple system call which lets TCP advertise a zero window to the FH. This would normally cause the FH to stop transmission. When the MN gets a new IP from the future network (event 2), it activates (handler 2) which transmits the future IP to the FH at TCP level through a system call. The new IP is sent in a special TCP segment with '`option=SWITCH_IP`'. At the FH, When TCP recognizes this option (event 4) it activates (handler 4) which then triggers a `switch_ip()` system call to replace the 'destination IP' field in the TCP/IP stack with the newly received IP number. In the mean time, at the MN (handler 2) also makes a similar system call which changes the 'source IP' filed in its own TCP/IP stack. When the previous '`SWITCH_IP`' segment is ACKed at the MN (event 3), the MN advertises a non-zero window to the FH which allows it to resume transmission.

Advertising a zero window to the FH to temporarily freeze the TCP connection was proposed in [6] to improve TCP performance over wireless networks. We adapted this part of their solution in our interactive scheme as a way to avoid packet loss during handoff.

TABLE-3. CN LOCATIONS WHICH RAN THE SERVER PROGRAM

Node name	Location	IP number	Average RTT	Hops from MN
Local	Kent, Ohio	131.123.36.11	1 ms	3
Virginia	Chantilly, VA	66.94.95.236	90 ms	19
Texas	Huston, Texas	70.241.64.99	183 ms	26

Although this will slightly disrupt the service while handoff is being performed, but since we avoid packet loss, the FH will not resort to congestion control procedures avoiding unnecessary retransmissions and sender rate throttling. As we show later, this will definitely improve TCP performance and save network resources.

V. EXPERIMENT AND PERFORMANCE ANALYSIS

We have implemented the scheme on FreeBSD-4.5 by extending the kernel source code. We call the extended implementation *BSD-interactive*[1]. We have created a number of system calls that implement the system's API. The API list is shown in Table-2. The first two system calls are standard in the interactivity service[2]. They allow demanding user applications to subscribe with lower network protocol to be notified when certain events occur. We configured a *Signal Handler* with the OS to catch all signals and filter them. Whenever an IPMN relevant signal is detected, the signal handler uses the `GetEventInfo()` function to retrieve event type from the network protocol which generated the signal. The signal handler then activates the appropriate *Transientware* module at the application level (e.g. handler 1 in figure-3) to handle the event. The rest of the API functions in the table are used by these event handlers as described earlier in the IPMN architecture.

A. Experiment Setup

Figure-4 explains the experiment testbed. We used three machines with AMD 1.6 GHz processor (BS1, BS2, and BS2) as our Base Stations and a laptop with Intel P-II processor as our MN. The (GW) machine was our gateway to the Internet and was also used to configure each one of the Base Stations as a separate subnet with four IP numbers per subnet. We installed FreeBSD-4.5 on all BS machines, the MN, and the CN. For Interactivity experiments we installed the BSD-interactive on the MN and the CN only. For the MIP experiments we installed the MIP implementation of the

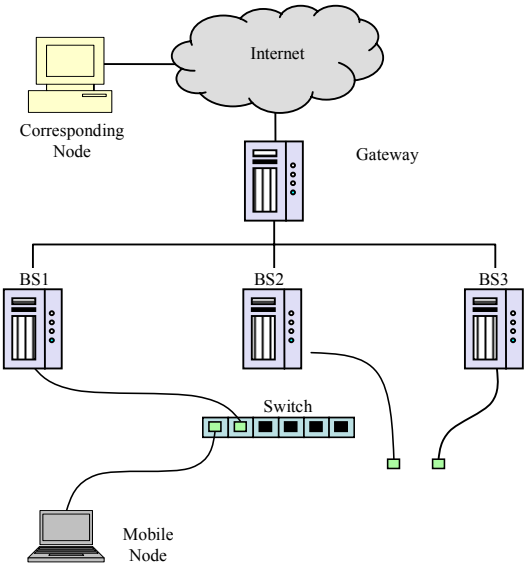

Figure-4. Testbed and experiment setup

Portland State University [1]–also known as PSUMIP—on the MN and the three BS machines. One of the three Base Stations machine (BS1) was configured as the Home Agent (HA), and the other two (BS2 and BS3) were configured as Foreign Agents.

For MIP signaling to work correctly, the time must be synchronized on all machines that run the MIP daemons. For this purpose we used the (`ntpd`) utility in FreeBSD to synchronize with three STATUM 2 external time servers. We used the simplest possible MIP configuration to reduce unnecessary overhead.

We wanted to test our interactive scheme against MIP and compare performance by measuring application level voice delay, handoff latency, and jitter. We have run all experiments by placing the CN in three locations, one locally (in our lab) and two remotely; in Texas and

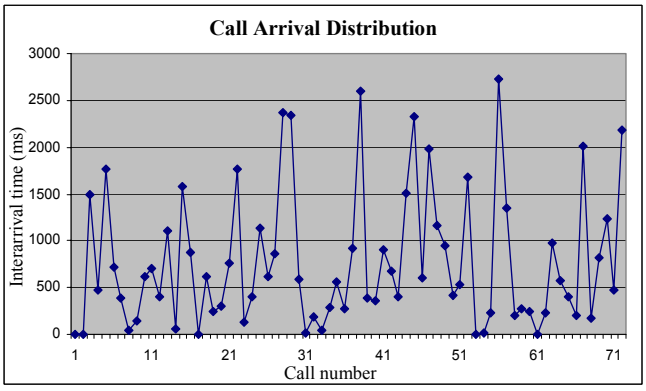

Figure-5. Call interarrival sampling over 5 hours for λ = 1. This generates a mean interarrival time of 1 second.

Virginia. Table-3 shows the three nodes and their route characteristics.

[1] A public distribution of the complete modified kernel is posted on our web page: http://www.medianet.kent.edu/ipmn/

[2] More details about the interactivity framework and associated service model and API can be found in [11].

Published in the Proceedings of the IEEE Wireless and Optical Communications, WiOPT05
3rd IEEE Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks
Riva Del Grande, Italy, April, 2005, pp.219-228

The CN generated voice traffic based on the NetSpec Source Models [9]. We also let the MN move along the cyclic path BS1→BS2→BS3→BS2→BS1… etc. In each run, we let server program at the CN transmit voice traffic until 5 Mbytes has been received at the MN. We configured the MN to perform handoff every 2 minutes. We used a switch to simulate L2 wireless handoff; for example, in figure-4 the MN is connected to BS1 through the switch. To perform L2 handoff from BS1 to BS2, we manually unplugged BS1 from the switch and instantly plugged BS2 to an empty port in the switch. We kept the MN connected to the switch all the time.

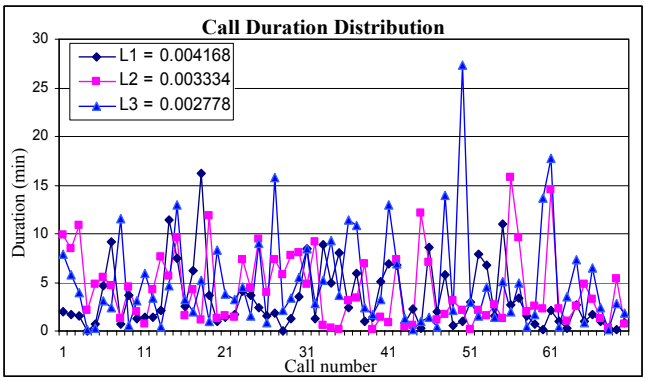

Figure-6. Call duration sampling over 5 hours. The three choices of λ shown here generate mean call durations of 3, 4, and 5 minutes repectively.

B. Traffic Characteristics

In order to model real-world traffic, we used a tool called NetSpec [9] which was developed at The University of Kansas—to generate traffic at the CN. Netspec offers several source models which can generate traffic for Telnet, FTP, Video, voice, and WWW [12]. In this experiment we only tested with voice traffic. We will consider the other types in future work.

In NetSpec., voice has been characterized by a constant bit rate (CBR) source. Sampling rate is 8 kHz and each sample is 8 bits. This gives the standard bit rate of 64 Kb/sec for acceptable voice quality. Call arrivals are modeled by a Poisson process with fix hourly rates within one-hour periods. This means that the interarrival time between two calls is exponentially distributed. The probability density function of exponential distribution is given by:

$$f_X(x) = \lambda e^{-\lambda x} \qquad ,\lambda = 1/mean$$

Session duration (holding time) for voice calls was also modeled by a Poisson process and followed the exponential distribution. Figure-5 shows an example of call arrivals with λ=1 over 5 hours sampling, and figure-6 shows an example of call duration over 5 hour sampling with three values of λ: λ_1=0.004167, λ_2=0.003333, λ_3=0.002777. If we take the inverse of these λs, we get mean call durations 3, 4, and 5 minutes respectively.

TABLE-4. HANDOFF LATENCIES OF THE FIRST FIVE HANDOFFS IN MILLISECONDS

Handoff	Local		Virginia		Texas	
	IPMN	MIP	IPMN	MIP	IPMN	MIP
1	106	12654	114	58669	202	51359
2	107	7124	106	24975	193	33187
3	111	1524	106	22672	195	29099
4	115	48945	111	77414	195	63523
5	109	1008	121	30772	200	41676
Average	**110**	**14251**	**112**	**42900**	**197**	**43769**

At the call level, the source is presented to the network as a constant-bit stream. To generate a 64 Kb/sec voice stream, talk bursts were generated by a 144-byte blocks separated by 18 ms silence periods.

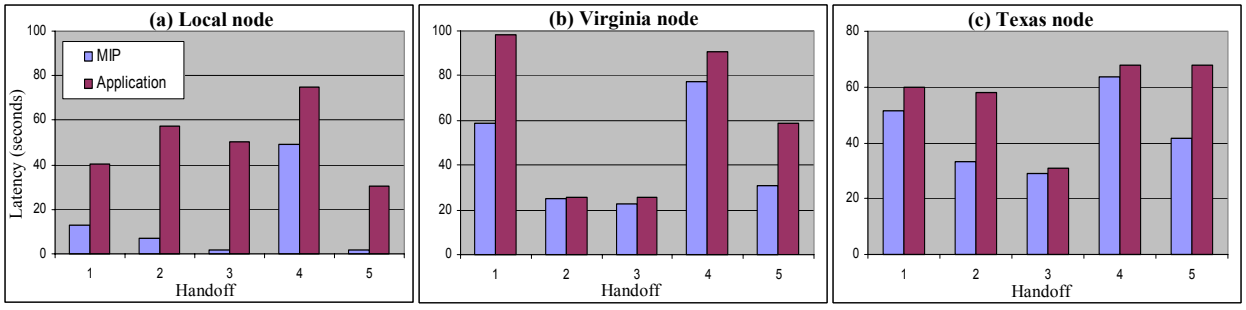

Figure-7. Handoff latencies on both MIP level and application level. While MIP introduced quite high latencies during handoffs, application level suffered even more latencies due to TCP recovery overhead.

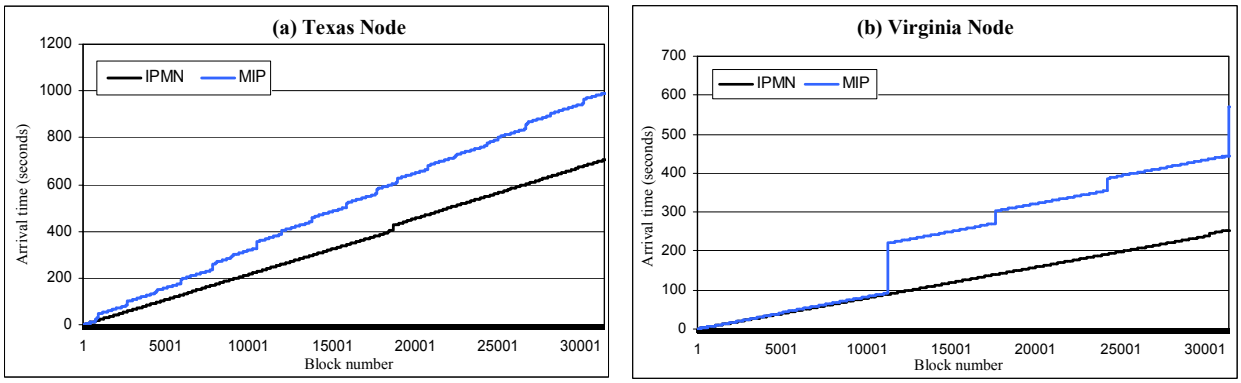

Figure-8. The arrival time of each 144-bytes block (talk burst) at the MN from (a) Texas node and (b) Virginia node. In both cases, IPMN outperformed MIP with a smooth plot and smaller step jumps during handoff.

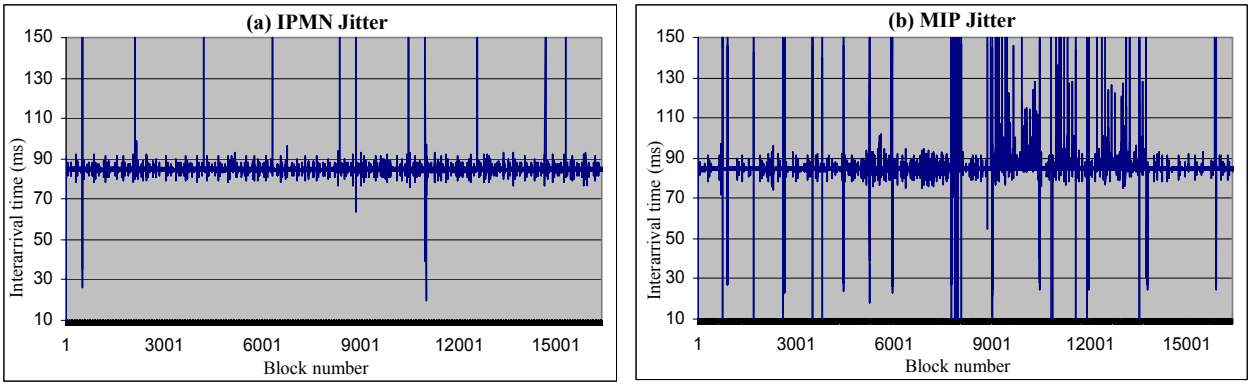

Figure-9. Block interarrival times at the MN. IPMN demonstrated acceptable low-jitter voice quality.

C. Handoff Latency

One of the key features of our interactive scheme is short handoff latency. After running the experiment several times on the three nodes we have observed a big difference –up to two orders of magnitude—in handoff latency between IPMN and classic MIP. Table-4 shows the handoff latency of the first five handoffs on the three nodes for both MIP and IPMN runs. IPMN managed to perform handoff in 110 to 200 milliseconds on average while MIP needed between 14 to 44 seconds. This substantial reduction in handoff latency highlights the advantage of event-based protocols like IPMN over timer-based protocols like MIP. The former allows protocols in different layers to interact and pass events and new state information –like the new IP number in our case—to upper layers instantly. This enables peer protocols to respond immediately, therefore cutting down unnecessary overhead time. Timer-based protocols on the other hand usually use a timer-based periodic probing mechanism to discover state changes. For example, in this particular implementation of MIP that we have tested, the foreign agent sends beacon signals (agent advertisements) to discover MN movement every 60 seconds! A best case scenario will happen if L2 handoff was performed right before the arrival of a beacon signal. Therefore, this process will take half of that time on average –i.e. 30 seconds. Adding to that communication and address registration overhead we can easily reach the 40 seconds average especially on the two remote nodes. Actual latency in MIP was even longer; by the time MIP recovers and becomes ready to resume service, TCP has already timed out and will probably need even more time to discover MIP recovery and then resume communication on its own level –and at the application level as well. We have observed this behavior of MIP by also registering the time when application level communication resumed after each handoff was completed. Figure-7 demonstrates this property by comparing the handoff latencies of the first 5 handoffs at the application level and at the MIP level. For example, on the Texas node –figure-7 (c), when handoff 2 was performed, the application level suffered 58 seconds of service disruption, even though the MIP was responsible for 33 seconds delay only. While in some cases, TCP overhead was small –like handoffs 2 and 3 on the Virginia node which had less than 2 second of TCP overhead, in other cases –like handoffs 2, 3, and 5 on the Local node, TCP added up to 50 seconds!

D. Delay on the voice stream

Now, we show application level performance by observing arrival delay. At the MN, we kept a log file to register the arrival time of each 144-bytes block (talk burst) in the voice stream. Figure-8 plots the arrival times of the first 31,000 blocks at the MN from the two remote nodes: Virginia and Texas. We did not include the Local node plot for space limitations.

As we can easily observe, the IPMN dramatically outperformed MIP on two levels: Firstly, in general, most blocks were delivered faster with IPMN due to shorter triangulation-free path that they had to travel to reach the MN as well as to smaller overhead. Secondly, IPMN plots were much smoother than MIP since the latter suffered from longer disruption of TCP service due to longer handoff delays. This can be observed in the Texas plots. After each handoff event, we see the impact of TCP's slow start behavior on the plot. These step jumps and the impact of TCP dynamics created jitter on the voice stream –as we show in the next section. One last observation is the impact of connection speed. The CN in Texas was transmitting on a 350 kbps DSL connection; MIP needed about 1000 seconds to transmit all 31,000 blocks while IPMN managed to transmit them in 700 seconds only –a 5 minutes difference. The node in Virginia was transmitting on 1.5 Mbps connection. It needed 570 seconds on MIP, and 250 seconds on IPMN –a 5.3 minutes difference.

E. Jitter on the voice stream

Figure-9 plots the interarrival times of the first 16000 blocks arriving at the MN from the Texas node, (a) on IPMN, and (b) on MIP. On IPMN almost all blocks were delivered at (75 to 90) ms apart, except (mainly) those that faced a handoff –only 22 blocks were delayed for more than 100 ms. In figure-9 we show a maximum of 150 ms on the y-axis to be able to see the mainstream case. Average interarrival time for all blocks on IPMN was 85.57 ms. On MIP the situation is different; about 177 blocks in the stream faced more than 100 ms interarrival –10 of these blocks faced more than 8000 ms delay—and average interarrival time for all blocks was 129 ms.

VI. CONCLUDING REMARKS

In this paper, we have presented IPMN –an interactive protocol for network mobility. It is based on interactive transparent networking paradigm which we have developed recently. IPMN uses true end-to-end signaling to update the current state of the mobile node's location at both end-points. Using interactivity, the MN was able to freeze the TCP connection and to perform loss-free, rapid handoff by simply changing the 'source IP' field in TCP/IP stack of the mobile node and the 'destination IP' field in the TCP/IP stack of the correspondent node.

We have shown by real experimentation with voice traffic that IPMN offered two key advantages over conventional timer-based MIP; (a) it allowed direct end-to-end communication between the correspondent node and the mobile node by eliminating triangular routing, and (b) it dramatically reduced L3 handoff latency by canceling movement detection and address registration. Also, we have shown that the scheme can meet the demands of a low-jitter voice stream.

Essentially, the results demonstrate the benefit of the principle of interactivity in networking. It enables event based action and response. It distinguishes from the traditional timer-based MIP which depends on periodic actions. The periodic agent advertisements (beacon signals) used in MIP is one of the prime reasons for its sluggishness. MIP has to maintain a delicate balance between advertisement frequency/size and their impact on network throughput[3]. Event-based scheme such as the one demonstrated by IPMN does not require this compromise. Indeed the benefit of instant interactivity was so dramatic that it could easily wipeout the seeming advantage of MIP's low layer implementation.

VII. REFERENCES

[1] Binkley J. and Singh S., The Portland State University Secure Mobile Networking Project (PSUMIP), http://www.cs.pdx.edu/research/SMN/, 1999.

[2] Campbell A., et al., "IP Micro-Mobility Protocols," ACM SIGMOBILE Mobile Computer and Communication Review (MC2R), Vol. 4, No. 4, pp. 45–54, October 2001.

[3] Fikouras N. and Görg C., "Performance Comparison of Hinted and Advertisement Based Movement Detection Methods for Mobile IP Hand-offs," In Proc. of the European Wireless 2000, Dresden, Germany, September 2000.

[4] Fikouras N., Könsgen A., and Görg C., "Accelerating Mobile IP Hand-offs through Link-layer Information," in Proc. of the International Multi-conference on

[3] The original MIP proposal [14] recommended shortest agent advertisement rate of 1 per second. But most real implementations do not achieve this short a rate due to overhead concern. The implementation that we have tested in this paper (PSUMIP) uses a much slower rate of 1 per minute. We tried to lower this rate, but it did not work. Since PSUMIP was the only available implementation compatible with FreeBSD-4.5 kernel at the time, we could not test with faster agent advertisement rate. Few other MIP implementations allow the user to set a preferred rate of one or more seconds such as SunMIP, KAME, and Mosquito, though such a low rate in not recommended.

Measurement, Modeling, and Evaluation of Computer-Communication Systems (MMB), Aachen, Germany, September 2001.

[5] Fikouras N., El Malki K., and Cvetkovic S., "Performance Evaluation of TCP over Mobile IP," In Proc. of the International Symposium on Personal Indoor and Mobile Radio Communications 1999 (PIMRC), Osaka, Japan, September 1999.

[6] Goff T., Moronski J., Phatak D., Gupta V., "Freeze-TCP: A True End-to-End TCP Enhancement Mechanism for Mobile Environments," INFOCOM'00, Tel-Aviv, Israel, pp. 1537-1545, 2000.

[7] Gustafsson E., et al. "Mobile IPv4 Regional Registration" draft-ietf-mobileip-reg-tunnel-05, IETF, September 2001.

[8] Hristea C. and Tobagi F., "A network infrastructure for IP mobility support in metropolitan areas," Computer Networks, 38, pp.181-206, 2002.

[9] Jonkman R., Evans J, Frost, V., "Netspec: A Tool for Network Experimentation and Measurement", University of Kansas, 1998. http://www.ittc.ku.edu/netspec/

[10] Khan J., Zaghal R., and Gu Q., "Symbiotic Streaming of Elastic Traffic on Interactive Transport," IEEE ISCC'03, Antalya, Turkey, July 2003.

[11] Khan J. and Zaghal R., "Protocol Modeling with Transparent Networking", CCCT'04, Austin, TX, August 2004.

[12] Lee Beng-Ong, "Wide Area ATM Network Experiments using Emulated Traffic Sources," Master's Thesis, University of Kansas, Lawrence, Kansas, 1995.

[13] Mishra A., Shin M., Arbaugh W., "An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process," Dept. of Computer Science, University of Maryland, technical report number CS-TR-4395.

[14] Perkins C., "IP Mobility Support," RFC2002, IETF, October 1996.

[15] Perkins C., "Mobile IP, Design Principles and Practices," Addison Wesley, 1998.

[16] Ramjee R., et al. "HAWAII: A Domain-Based Approach for Supporting Mobility in Wide-area Wireless Networks," Proc. IEEE Int'l Conf. Network Protocols, 1999.

[17] Singh R., Tay Y., Teo W., and Yeow S., "RAT: A Quick (And Dirty?) Push for Mobility Support," 2nd IEEE Workshop on Mobile Computer Systems and Applications, pp. 32, Feb. 1999.

[18] Stevens W., "TCP/IP Illustrated Volume I," Addison-Wesley, 1994.

[19] Valk´o A., "Cellular IP: A New Approach to Internet Host Mobility," ACM SIGCOMM Computer and Communication Review, Vol. 29, No.1, pp. 50–65, January 1999.

[20] Wu J., et al. "Intelligent Handoff for Mobile Wireless Internet," Mobile Networks and Applications, Vol. 6, pp.67-79, 2001

[21] Wu W., Banerjee N., Basu K., Das S., "Network Assisted IP Mobility Support inWireless LANs," [22] Yokota H, et al., "Link Layer Assisted Handoff Method over Wireless LAN Networks," Proc. of MOBICOM '02, Sept. 2002.