An Efficient Mechanism of TCP-Vegas on Mobile IP Networks

Cheng-Yuan Ho, Yi-Cheng Chan, and Yaw-Chung Chen
Department of Computer Science and Information Engineering
National Chiao Tung University
1001 Ta Hsueh Road, Hsinchu 30050, Taiwan, R.O.C.
cyho@csie.nctu.edu.tw, ycchan@csie.nctu.edu.tw, ycchen@csie.nctu.edu.tw

***Abstract*— Mobile IP network provides hosts the connectivity to the Internet while changing locations. However, when using TCP Vegas over a mobile network, it may respond to a handoff by invoking a congestion control algorithm, thereby resulting in performance degradation, because TCP Vegas is sensitive to the change of RTT (Round-Trip Time) and it may recognize the increased RTT as a result of network congestion. Since TCP Vegas could not differentiate whether the increased RTT is due to route change or network congestion. This work investigates how to improve the performance of Vegas after a Mobile IP handoff and proposes a variation of TCP Vegas, so-called Demo-Vegas, which is able to detect the movement of two end-hosts of a connection, and re-measure the BaseRTT (Minimum Round-Trip Time) if necessary. The proposed mechanism maintains end-to-end semantics, and operates under the existing network infrastructure. Demo-Vegas presents a simple modification in the two end sides of a connection and uses one reserved bit in TCP header. Simulation results demonstrate that Demo-Vegas features higher performance than Vegas in Mobile IP networks.**

Index Terms*— *TCP, TCP-Vegas, and Mobile IP.

I. INTRODUCTION

THE Internet is an infrastructure interconnecting a large number of heterogeneous computer networks using TCP/IP protocol suite (Transmission Control Protocol/Internet Protocol). TCP Vegas [1] ensures end-to-end integrity of data transfer, while IP performs datagram routing and internetworking functions. With the fast prevalence of Internet, the popular usage of laptop and notebook computers and the deployment of wireless communication devices, users demand the mobility of hosts, i.e., they expect that the hosts can change their locations continuously without interrupting current communication sessions.

However, current IP routers make use of IP address for datagram routing decisions. After a host moves from one network to another, datagrams destined to the original address will not be routed to the new location. To receive datagrams at the new location, a host must obtain a new network address and advertise it. To provide an economical solution which implements mobility support over the existing Internet infrastructure, the Mobile-IP Working Group of the Internet Engineering Task Force (IETF) has compiled a series of Internet Drafts and Request for Comments (RFC) to define Mobile IP [2], [3].

MIP (Mobile Internet Protocol) provides hosts with the ability to change their point of attachment to the network without compromising their ability in communications. The mobility support provided by MIP is transparent to other protocol layers so as not to affect those applications which do not have mobility features. MIP introduces three new entities required to support the protocol: the Home Agent (HA), the Foreign Agent (FA) and the Mobile Node (MN). Further information on MIP functionality can be found in [2], [3]. With the introduction of IPv6, MIPv6 [8] excludes the FA, because FA's functionality has been distributed amongst the MNs and Dynamic Host Configuration Protocol (DHCP) enabled hosts. Therefore we will often refer to the word 'agent' (i.e. mobility agent) as a generalization to identify HAs and FAs in MIPv4 and HAs in MIPv6.

There are several problems of using TCP scheme in a Mobile IP network. Since TCP is tuned to perform well in traditional wired networks in which most packet losses are due to congestion. However, in a wireless mobile network, packet losses usually occur due to either random loss or handoff. After a handoff, the throughput of TCP Vegas may be decreased due to a longer BaseRTT of the new routing path, which is usually caused by either triangular routing or route optimization. In this paper, we focus on the performance of TCP Vegas after Mobile IP handoff. We propose a modification of TCP Vegas, called Demo-Vegas (Detect Mobility Vegas). When a node resides in its home network, it communicates normally. While it enters into a foreign network, it will get a new IP address, called COA (Care of Address). Demo-Vegas is able to detect the movement of both a sender and a receiver based on their COAs. It is simple with very little overhead because only one reserved bit in TCP header is used, and a small modification in the end side is made. This facilitates incremental deployment in today's Internet. Our intensive simulation shows that Demo-Vegas significantly improves the overall TCP Vegas throughput in mobile environment.

The rest of this paper is organized as follows. Section II introduces TCP Vegas and Mobile IP. Related work is described in Section III. We characterize the motivation, scheme, and pseudo code of Demo-Vegas in section IV. Section V presents the simulation results and Section VI concludes the paper.

0-7803-8968-9/05/$20.00 (C)2005 IEEE

II. Background: TCP Vegas and Mobile IP

A. TCP Vegas

Vegas [1] uses the difference in the expected and actual flow rates to estimate the available bandwidth in the network. When the network is not congested, the actual flow rate would be close to the expected flow rate. On the other hand, if the actual rate is smaller than the expected rate, it indicates that buffers in the network are filling up and the network is approaching congestion. This difference in flow rates can be calculated as *Diff* = *Expected* – *Actual*, where *Expected* and *Actual* are the expected and actual rates, respectively.

(i) Congestion Avoidance

In its congestion-avoidance phase, Vegas uses two threshold values, α and β (whose default values are 1 and 3, respectively), to control the adjustment of the congestion window size at the source host. Let d denote the minimum-observed packet round-trip time (also known as BaseRTT), D denotes the actual round-trip time (RTT), and W denotes the size of the congestion window size, then *Expected* = *W/d* and *Actual* = *W/D*. In addition, W is measured in segments as is normally done in any TCP version. The estimated backlog of packets in the network queues can then be computed as

$$\Delta = (Expected - Actual) \times BaseRTT = W \times \frac{(D-d)}{D}. \quad (1)$$

For every RTT, the congestion-avoidance algorithm adjusts W as follows:

$$W \leftarrow \begin{cases} W+1, & if \Delta < \alpha \\ W-1, & if \Delta > \beta \\ W, & otherwise(\alpha \le \Delta \le \beta). \end{cases}$$

Conceptually, Vegas tries to keep at least α packets but no more than β packets per flow queued in the network. Thus, when there is only one Vegas connection, W converges to a point that lies between *window* + α and *window* + β where *window* is the maximum window size without considering the queuing in the network.

(ii) Slow Start

Like Reno, Vegas uses a slow-start mechanism that allows a connection to quickly ramp up to the available bandwidth. However, unlike Reno, to ensure that the sending rate will not increase too fast to congest the network during the slow start, Vegas doubles its congestion window size only every other RTT, and calculates the difference between the flow rates (*Diff*) and Δ given in (1) in every other RTT. When $\Delta > \gamma$ (whose default is 1), Vegas leaves the slow-start phase, decreases its congestion window size by 1/8 and enters the congestion-avoidance phase.

Since Vegas estimates the BaseRTT to compute the expected flow rate and adjust congestion window size, it is important to get an accurate BaseRTT. When one or both of two end hosts of a connection stay connected with each other while changing the location, the routing path may change and lead to rerouting, which in turn may result in Δ bias and throughput degradation due to the change of end-to-end fixed delay, the sum of propagation delay and packet processing time. If the fixed delay becomes shorter, it will not cause any problem. Otherwise if there is a longer fixed delay, it could not differentiate whether the increased RTT is due to route change or network congestion. The sender may reduce its window size and it may cause inefficient throughput. In short, while Vegas presumes that packet delay or loss is due to congestion, in wireless networks, packet delays can also be due to changes in mobile IP address.

B. Mobile IP

MIP extends the existing IP protocol to support host mobility while preserving a security level as good as today's Internet standards. The basic idea is to use an authenticated registration procedure between a MN and a HA in its home network, and via a FA while MN is visiting a foreign network. The mobility bindings is created between MN's home IP address and a temporary COA, which is granted by the FA or selected autonomously by the MN (with DHCP), in the visited network. The binding information is stored in the HA and FA, and updated whenever a MN moves to the jurisdiction of a different FA. While datagrams sent to a correspondent node (CN) are routed through normal means using the CN's IP address (if FA permits), datagrams sent to the MN are routed to the MN's home network as the CN only knows the MN's home IP address. Once the mobile registration is completed, the HA will intercept these datagrams on behalf of the MN, and use the mobility binding to redirect the datagrams to the FA by means of tunneling. Based on the current binding for the MN, the FA forwards the datagrams to the MN.

The triangular routing of datagrams to the MN through its home network may be inefficient, particularly if the CN is much closer to the MN's visited network than the home network. A route optimization method, incorporated in a superset of Mobile IP called the Internet mobile host protocol (IMHP), has been proposed [4]. Route optimization is a backward compatible extension to the Mobile IP. The idea is to allow an HA to take the available MN's current binding to the requesting CN, or a specific router on behalf of the CN. In either case, the node must maintain a cache of MN's bindings, called the cache agent (CA), which can tunnel datagrams directly to the MN's current COA, bypassing the MN's home network and eliminating the triangular routing.

III. Related Work

Because the related work of Vegas in this area is not much, we present some TCP modifications targeted to Mobile IP networks instead. The first approach is M-TCP, which forces the sender to enter a TCP persist mode when an intermediate node detects a disconnection [5]. The Mobile End Transport Protocol (METP) is a split-connection protocol that replaces the TCP/IP protocol stack over the wireless interface which uses a simpler protocol with smaller headers [6]. In order to set up a communication with a fixed host, the routing and flow control functionality of the MT (mobile terminal) is undertaken by the BS (base station). An interesting feature of

METP is that it exploits a link layer ACK and retransmission mechanism to deal with the errors in the wireless link. The BS maintains both a sending and a receiving buffer along with state information for each connection. It delivers ACKs of TCP packets back to the source host; then a separate process undertakes the transfer of buffered packets to the MT.

Lightweight Mobility Detection and Response (LMDR) [7] holds a counter that represents the number of times a side has changed attachment points. There are 24 bits in LMDR TCP option to represent TCP Option Type (8 bits), TCP Option Length (8 bits), Reserved bits (2 bits), CNTR (3 bits), and ECNT (3 bits) respectively. In addition, the value of CNTR is decremented once for every subnet change, and the value of ECNT is the echoed value of CNTR. When the source concludes that there has been a remote subnet change, it will do two steps. First, it resets the congestion control state, RTTM state, and RTO timer as if this is a new connection. Second, for each "stale ACK", which is the acknowledgement corresponding to data sent on the old path, is received, it doesn't adjust the congestion window and send any new data into the network until there is a timeout or destination tells source to send new data. When CN doesn't support the route optimization, this mechanism may be not efficient.

IV. Demo-Vegas

A. Motivation

TCP Vegas is a rate based mechanism, it adjusts the congestion window based on the current congestion window size, BaseRTT, and newly measured RTT. Vegas can successfully avoid the congestion in the network, so there are implementations of Vegas in some operating systems such as Linux and NetBSD. However, it would compute Δ bias when the fixed delay of routing path changes. Since Vegas could not detect a prolonged BaseRTT, so it decreases the congestion window size until the value of Δ is between α and β. As a result, on a Mobile IP network, Vegas may not utilize the bandwidth efficiently. We propose a variant of TCP Vegas, Demo-Vegas, to solve this issue. Our method does not influence the original scheme on the wired network.

B. The Scheme of Demo-Vegas

When a node moves from its home network to a foreign network, it will change its COA to receive packets. Since the fixed delay of routing path may be changed, we modify the end hosts including both the sender and receiver to detect their location change in Demo-Vegas. In the sender side, while the sender knows that its COA is changed (if it moves to another foreign network) or is cancelled (if it moves back to its home network), it will re-measure the BaseRTT. In the receiver side, a receiver will tell the sender to re-measure BaseRTT when it changes its COA. Thus, we use a reserved bit in TCP header, called 'SIG' bit as shown in Fig. 1, to represent the change of receiver's COA. The sender records the value of 'SIG' and the receiver will keep the 'SIG' bit value of each ACK unchanged until its COA is changed again. When the value of 'SIG' bit changes from 0 to 1 (or from 1 to 0), it means

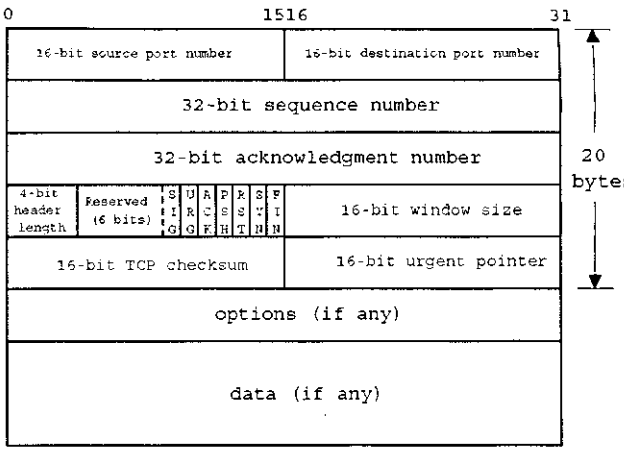

Fig. 1. The 'SIG' bit in TCP header records the change of receiver's COA.

that the receiver's COA has been changed. In the beginning, the value of 'SIG' bit is set to 0. Once a receiver detects the change of its COA, it will alter the 'SIG' bit value to inform the source to re-measure the BaseRTT. We do not only set the 'SIG' bit in the first packet because the packets may get lost during handoff. Whenever a sender receives an ACK, it compares the value of the 'SIG' bit with its current value. If they are different, the sender will re-measure the BaseRTT.

Since idea is simple and space is limitative, we omit the pseudo codes in this section. The proposed scheme can improve the performance of TCP Vegas based on the simulation results in the following section.

V. Performance Evaluation

A. The Simulation Environment

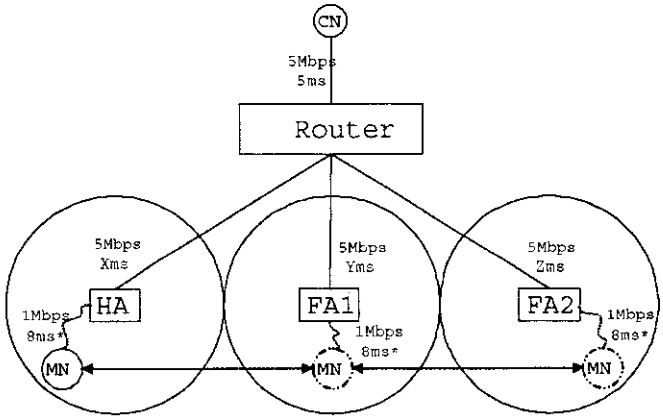

Fig. 2. A simple topology for simulation experiments.

The simulation experiments are conducted using the *ns2* [9], version 2.26. A simulation network topology is shown in Fig. 2, where CN and MN represent end hosts. CN and MN are the two end sides which execute either Vegas or Demo-Vegas. The application service in our simulation is FTP. The receiver sends an ACK for every data packet received. For

the convenience of presentation, we assume that all window sizes are measured in number of fixed-size packets, which are 1000 bytes. Router, HA, FA1 and FA2 represent three finite-buffer gateways. The buffer size in each gateway is set to 50 packets. For the constant-load experiment, drop-tail gateways with FIFO service are assumed. The bandwidth is 5 Mbps for all wired links. The propagation delay is 5 ms from CN to router, X ms from router to HA, Y ms from router to FA1, and Z ms from router to FA2, respectively. From an agent (HA, FA1 or FA2) to MN, the bandwidth is 1 Mbps and wireless transmission delay is a multiple of 8 ms which includes both packet transmission delay and the propagation delay. The former may account the layer 2 retransmission due to unsuccessful frame delivery, while the later can be ignored because the propagation delay is much smaller comparing with the packet transmission delay. In addition, this is a two dimensional plane in topology. The distance of radio coverage for the agent is 75 meters. The position of HA is (200, 300), FA1 is (350, 300) and FA2 is (500, 300).

B. Simulation Result

TCP Vegas re-initializes the BaseRTT and re-transmits the lost packets when a short session finishes its handoff process. Therefore, in following simulations, we show the comparisons between TCP Veags and our mechanism using a long session, and focus on their behavior after the handoff(s).

1) Configuration with a fixed sender and a mobile receiver: In the first simulation, CN is the sender, MN is the receiver, X=3, Y=3 and Z=1. The BaseRTT is about 32 ms if the packet is transmitted successfully the first time. The MN moves with a speed of 10 m/s from (150, 275) to (350, 275) at the 10^{th} second, then moves to (550, 275) at the 40^{th} second. It starts to come back (350, 275) at the 70^{th} second, then return to (150, 275) at the 100^{th} second. When MN is in the foreign network, the datagrams are routed from CN to HA, tunneled from HA to FA, and FA de-tunnels these datagrams to MN. The ACKs are routed directly from MN to CN through the FA. The RTT is at least 38 ms or 34 ms respectively depending on whether the MN is in the foreign network 1 or in the foreign network 2. In Vegas, the sender is unable to tell the routing path change from congestion, so the latter is assumed. Therefore it reduces the congestion window size and thus the throughput decreases. While in Demo-Vegas, the sender is able to detect the receiver's location change through 'SIG' bit and consequently re-measures the BaseRTT. The throughput of Vegas and Demo-Vegas are shown in Fig. 3, where we can observe that Demo-Vegas outperforms Vegas when the MN is in the foreign networks.

We are interested in the influence of using reverse tunnel, in the second simulation, the MN moves from (150, 275) to (400, 275) at the 10^{th} second and comes back at the 60^{th} second with a speed 10m/s. Since a FA could not help a visiting MN route the ACKs which may be blocked by a firewall or other security systems in the visited FA. Therefore, the MN must use reverse tunnel to send ACKs back to its HA, then the HA routes the ACKs to the CN. Here, the BaseRTT is about 32 ms, CN is the source host, MN is the destination host, X=3 and Y=1 (in order to distinguish one-way tunnel from two-way tunnel). Fig. 4 shows that FA helps route the ACKs from MN when the MN is in the foreign network. In other words, only the mechanism of tunnel is used here. On the contrary, in Fig. 5 FA does not route the ACKs through itself from MN, so MN must use the reverse tunnel to send the ACKs. When the MN

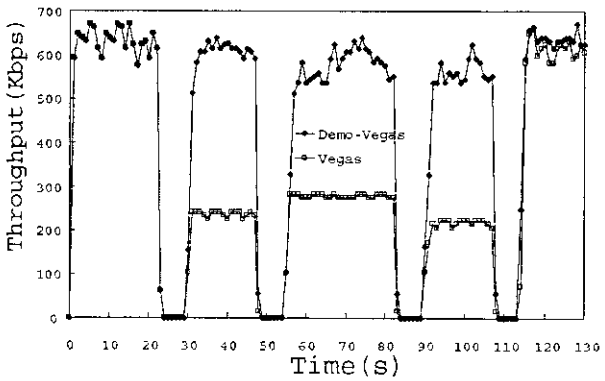

Fig. 3. Throughput of Vegas and Demo-Vegas. When MN is in the foreign network, datagrams are tunneled from HA to FA. X=3, Y=3 and Z=1, Sender: CN, Receiver: MN.

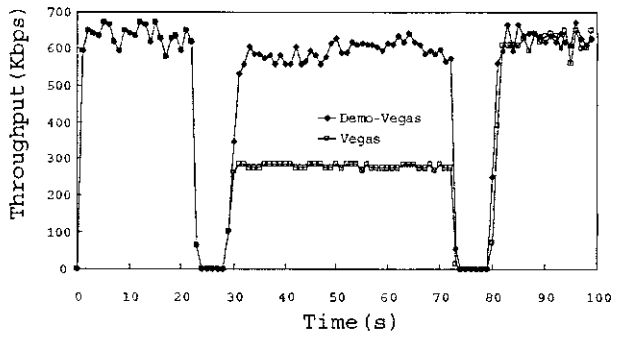

Fig. 4. Throughput of Vegas and Demo-Vegas. CN is the source host, MN is the destination host, X=3 and Y=1.

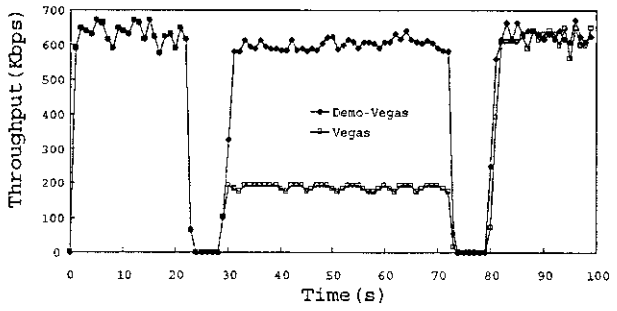

Fig. 5. Throughput of Vegas and Demo-Vegas. The conditions are same as in Fig. 4 except using reversed tunnel.

is in the foreign network, the RTT is at least 34 ms and 40 ms, as shown in Fig. 4 and Fig. 5, respectively. Accordingly, the throughput of Vegas in Fig. 4 is better than that in Fig. 5. However, Demo-Vegas always performs better throughput than Vegas no matter which mechanism is used, as observed in both Fig. 4 and Fig. 5, because it can detect the change of routing path.

In the third simulation, the CN supports the route optimization which enables datagrams to be routed directly from CN to MN (or from MN back to CN) when MN moves to a foreign network. Similarly, the CN is sender, the MN is receiver and the movement of the MN is same as the second simulation. There are two situations here, in the first one the BaseRTT of the new routing path is longer than the old one, while in the second one, it is just the opposite. Figure 6 shows the former case with X=1 and Y=3. When MN is in the foreign network, the throughput of Vegas is not very low because it could not detect the change of fixed delay in the routing path. However, in Fig. 7, when the MN moves back to its home network, the throughput is decreased to a small value. Here, we set X=3 and Y=1 to represent that the BaseRTT of the new routing path is shorter than the old one when the MN moves into a foreign network. When MN returns back to its home network, Vegas

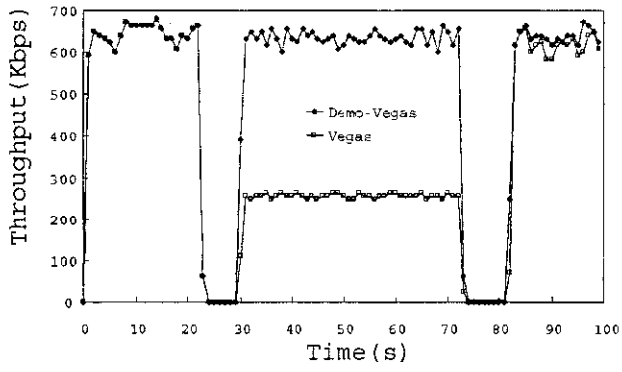

Fig. 6. Throughput of Vegas and Demo-Vegas with CN supporting route optimization and longer RTT in a foreign network.

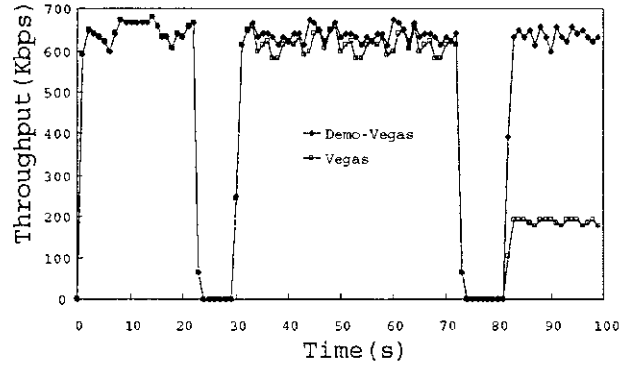

Fig. 7. Throughput of Vegas and Demo-Vegas with shorter RTT in a foreign network and CN supporting route optimization.

will assume the congestion occurs in the path and decrease the congestion window size. In Demo-Vegas, once a MN changes its COA, it informs the sender to re-measure the BaseRTT. Therefore, the throughput of Demo-Vegas can be kept at a steady value no matter the MN leaves from or comes back to its home network. In addition, the way of route optimization in IPv4 is much like that in mobile IPv6, so Demo-Vegas is still suitable in IPv6 network.

Due to the space limitation, we only show the results of configuration with a fixed sender and a mobile receiver. Furthermore, the diagrams of configuration with a fixed receiver and a mobile sender, or two mobile nodes are just like that with one fixed node and one mobile node. From Figures 3~7, we could observe that the performance of Demo-Vegas is much better than Vegas when one side is fixed and the other side is mobile because Demo-Vegas could re-measure the BaseRTT in time. Thus, when both end sides are mobile, the throughput of Demo-Vegas will be still better than Vegas.

VI. Conclusions

We propose and evaluate a new variant of TCP Vegas, called Demo-Vegas, to improve the performance over Mobile IP network. In this work, we achieve a significantly higher throughput comparing with original TCP Vegas on a Mobile IP network. Demo-Vegas could detect the change of routing path then re-measure the BaseRTT to avoid computing the value of Δ bias. From the numerical result of Vegas and Demo-Vegas with triangular routing, reverse tunnel and route optimization from the simulations, it shows that Demo-Vegas is more suitable than Vegas on a Mobile IP network. In addition, Demo-Vegas will still work well when the IPv4 environment changes to IPv6. Furthermore, the 'SIG' bit setting propagates past the NAT mechanism that enables mobile IP, to the remote host, forcing that host to recalculate its Vegas parameters. The mechanism of Demo-Vegas is simple and can be easily implemented on existing operating systems. We are going to modify Demo-Vegas to fit the node in the micro mobility networks in the future works.

References

[1] L. Brakmo and L. Peterson, 'TCP Vegas: End-to-End Congestion Avoidance on a Global Internet', *IEEE J. Sel. Areas Commun., VOL. 13, NO. 8, pp. 1465-1480, October 1995*

[2] C. Perkins, 'IP Mobility Support', *IETF RFC 3220, January 2002*

[3] C. Perkins, 'IP Mobility Support for IPv4', *IETF RFC 3344, August 2002*

[4] A. Myles, D. B. Johnson, and C. E. Perkins, 'A Mobile Host Protocol Supporting Route Optimization and Authentication', *IEEE J. Sel. Areas Commun., VOL. 13, pp. 839-849, June 1995.*

[5] K. Brown and S. Singh, 'M-TCP: TCP for Mobile Cellular Networks', *ACM SIGCOMM Computer Communication Review, VOL. 27, NO. 5, pp. 19-43, October 1997*

[6] K. Wang and S. Tripathi, 'Mobile-End Transport Protocol: An Alternative to TCP/IP over Wireless Links', *Proc. IEEE INFOCOM, VOL. 3, pp. 1046-1053, April 1998.*

[7] Y. Swami, K. Le, and W. Eddy, 'Lightweight Mobility Detection and Response (LMDR) Algorithm for TCP', *IETF Internet-Draft draft-swami-tcp-lmdr-04, August 2004*

[8] D. Johnson, C. Perkins, and J. Arkko, 'Mobility Support in IPv6', *IETF Internet-Draft mobileip-ipv6-24, June 2003*

[9] http://www.isi.edu/nsnam/ns/