TCP Multi-homing Support in Heterogeneous Networks

Dang Duc Nguyen, Yang Xia, Chai Kiat Yeo, Bu Sung Lee

Abstract—**Mobile technology provides users with ubiquitous connectivity across a wide range of access networks. The integration of multiple network technologies results in different perspectives in bandwidth usage and delay requirements during handover when a multi-homed mobile host traverses the different access networks. Here, we focus on a multi-homed mobile node with seamless mobility and bandwidth aggregation over multiple links. The system lies on the network layer and supports end-to-end TCP multi-homing using SIP signaling protocol. It helps to reduce TCP handover delay and achieve aggregated bandwidth. We propose an In-Kernel Re-Order Packet scheme to reduce out-of-order packets for TCP. Experimental results show that the proposed system can achieve aggregated bandwidth, increase reliability and reduce handover delay.**

Index Terms— **Multi-oming, TCP, heterogeneous wireless networks, multiple interfaces**

I. Introduction

As technology has empowered mobile devices with multiple accesses to the Internet, mobile users will certainly wish to take advantage of the multi-homing technology to maximize their benefits when using the Internet, especially for data streaming, such as video streaming or VoIP services. However, different technologies with different access and complicated network protocols prevent applications from effectively exploiting them. In most cases, the application needs to rely on either network-based solution or terminal-based solution due to the following reasons: (1) Unable to handle different interfaces at once; (2) Complicated protocol needs sophisticated solution - how to address the issue of maintaining QoS level while using multihomed solution; (3) A virtual interface is needed, and therefore, Care-of-Addresses (CoAs) and Home addresses (HoAs) mapping are complicated.

Furthermore, Transmission Control Protocol (TCP) provides congestion control and it may get complicated when it comes to different links being used at the same time and at different bitrates. The changes in network location dynamically affect the input/output data traffic. Therefore, it is crucial to notify the transport layer at both ends to adjust the rate of data transfer when the multihomed mobile node (MMN) moves to a new network. If MMN's sending rate is too fast for the new network link, substantial packet loss will result if the transport layer does not reinitialize its congestion control state for the new network path. If the receiver senses a link degradation and requests to reduce the window size of the TCP flow, performance, especially of real-time applications, will be impacted.

TCP packet is sequenced and sent in order. For multiple links, each packet is redirected to multiple network paths with unknown number of hops. This results in the differential delay experienced by each data packet. If the rate difference is significant, the overall average delay will be substantial and link quality will be degraded. Out-of-order incoming packets could spawn a series of redundant retransmission packets, further jamming the slowest link, thus compounding the out-of-order problem.

In this paper, we propose Multi-homed Mobility System (MMS) that supports seamless connectivity and achieves aggregated bandwidth for TCP in heterogeneous networks. The novelties are: (1) Seamless connectivity; (2) Aggregated bandwidth; (3) Support of TCP connection with a new packet distribution policy to reduce out-of-order packets; (4) Terminal based routing is used instead of router based routing. Hence, it does not need home agents or foreign agents to help re-route the traffic.

II. Architecture of Multi-homed Mobility System

A. Overview

We employ a cross-layer solution by using layer 2 information (link layer info) to help perform handoff in layer 3 (network layer). The proposed Multi-homed Mobility System (MMS) framework spans from the user space to the kernel space, comprising both kernel modules and application agents as per TMSP [1]. MMS is based on SIP [2] protocol for the registration process, uses IP address swapping technique as per TMSP to handle location management and performs a ``make-before-break'' handoff. Compared with TMSP [1], MMS provides seamless connectivity with aggregated total bandwidth and handles re-ordering of out-of-order or missing packets for QoS support in heterogeneous networks.

Fig. 1 shows the architecture of MMS comprising IP Swapping, IP/Ports Mapping, Packet Redirecting and Communication Modules as well as the Intelligent Software Agent.

The application layer is unaware of the changes under the data-link layer where different interfaces are used to transfer the packets. A Mapping Reference Table (MRT) is maintained where source and destination IP addresses are mapped against the CoAs. Outgoing data is transferred from

Manuscript received Dec 15, 2012

D. D. Nguyen, Y. Xia, C. K. Yeo, B. S. Lee are with the School of Computer Engineering, Nanyang Technological University (NTU), Nanyang Avenue, S 639798, Singapore. (email: {dndang, xiayang, asckyeo, ebslee, eiysoon}@ntu.edu.sg. Corresponding author is C. K. Yeo, phone: +65 67904929; fax: +65 67926559; e-mail: asckyeo@ ntu.edu.sg).

the application to the kernel space, IP header is swapped and matched to the corresponding CoAs of the destination, then the packet is routed to the interface accordingly. Incoming data packets IP header is compared and swapped with the stored CoAs and HoAs in the Mapping Reference Table (MRT). After that, the packet returns to the user space to the application with the same IPs as the previous packets. The application accepts the packets as a normal connection.

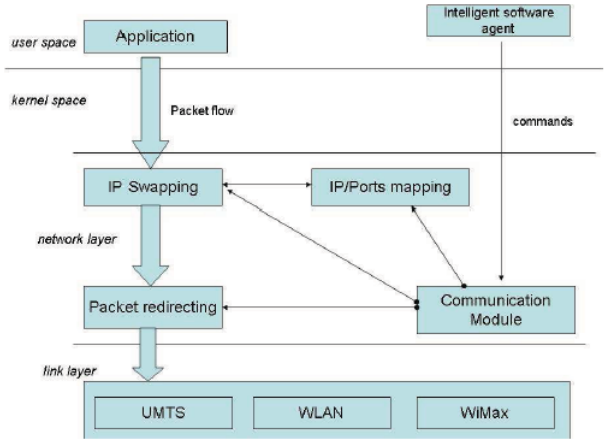

Fig. 1. System Architecture of MMS.

B. System Flow

The system flow of MMS is described as follows:

Registration Process

MMS uses SIP to signal information from each node to the SIP server. SIP protocol is employed to address the multi-homing issue. Each multihomed mobile node is identified by a unique SIP address. A SIP server is used to manage the association between the mobile node's SIP address and its Care-of Addresses (CoA). When a mobile node (MN) joins the network, it will publish its SIP address and the list of CoAs using REGISTER message with the SIP server. The corresponding node (CN) can then obtain the list of CoAs of the multihomed mobile node (MMN) from the SIP server using QUERY message. When CN wants to transfer data to MMN, it will first attempt to set up a session with MMN using INVITE message. When MMN receives the INVITE message, it will accept and reply with 200 OK message with a list of preferred CoAs. This list of CoAs is a subset of MMN's interface addresses which have better signal strength or lower traffic. This implies that the list of CoAs used in different sessions may be different and this makes it possible to balance the traffic load among multiple links. The session is established after CN replies with an acknowledgment.

Data Communication Process

When the session is established, both MMN and CN will record each other's address as reference peer address of this session and the network port used for data transfer. This peer IP address is the address used by the transport layer connection and the network address translation process. The network ports are used to identify data flows. Then the data transmission will start and CN can distribute the traffic among the list of CoAs of MMN based on the QoS requirements.

Handover Process

When MMN senses that one of its wireless interfaces is seeing lower signal strength, and it is picking up stronger signals from a new network, or it suddenly drops out of the network, it will send a binding update (BU) message to SIP server and any communicating CNs to update its CoA list. The BU message will also include information on the traffic condition for each CoA. CN can then re-distribute the flow based on the new CoA list. In MMS, we can use both flow distribution and packet distribution. Each method has its pros and cons and in this paper we use packet distribution to investigate and study the performance of our system.

C. System Design

The system detail is described in detail as follows:

IP Swapping Module

IP swapping module is shown in Fig. 2 located at the network layer. It gains access to both incoming and outgoing packets, allowing it to alter the IP headers. When multihoming packets are received, each packet is assigned to a particular interface from the Packet Redirecting module. The IP swapping module matches the source IP with the corresponding CoAs in the MRT and replace it with HoA. Likewise for the destination IP address. Note that one HoA can correspond to multiple CoAs in the mapping table if that machine (the current device or the communicating party) has multiple interfaces in active mode. It means that all CoAs and HoAs have been registered before at the SIP server. The registration step is discussed in the next section.

At the initialization stage, the HoAs of both parties are recorded together with the local port and the remote port.

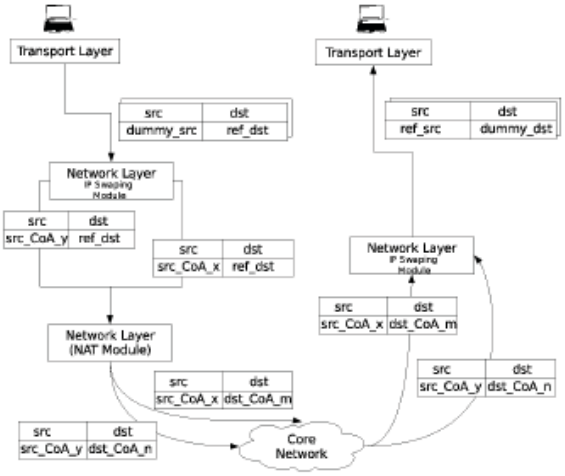

Fig. 2. Illustration of Network Address Translation (IP Address Swapping).

For the outgoing packets, the destination address is compared and matched to the corresponding CoAs. There might be more than one, in case the other party is also using multiple interfaces. The source address for each packet is replaced by the kernel, with the corresponding IP address of the interface that it uses to send out the packet.

For the incoming packets, the destination IP address is swapped with the virtual interface's IP address which is a

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

fixed IP address assigned at the start of communication. The source IP address is passed to the IP/Ports mapping module, to check whether the packet belongs to the current connection. If it does, the source IP address is swapped with the corresponding HoA. After that, the packet is passed to the upper layers. At the application level, the running application is not aware of the changes in the IP addresses since there are no changes in the source and destination IP addresses from the application's perspective. Therefore, the packet is accepted and processed as per normal. For TCP packet, there is no out-of-sequence packet problem, as this IP Swapping module is located at the network layer.

The transport layer processes the packet after it is modified at the network layer where the checksum and QoS controls are processed by the kernel functions. There is a problem with TCP packet when the window size of each packet is reduced during handover as the receiver will perceive that the connection is interrupted due to network congestion. It will therefore send a message to reduce the window size of the subsequent packets. This is called slow start and this problem can be solved by a buffer at the kernel level. More details will be given in the following sections.

IP/Ports Mapping Module

Here, the IP address is translated or mapped to multiple CoAs stored in a table. During initialization, the destination IP address, source IP address, local port, remote port are combined into a row entry to represent a connection. Each connection is assigned with a unique index which is generated from the local and the remote ports.

An arbitrary constant IP address from an unused private prefix is assigned to the virtual interface as the permanent address of the local machine. For UDP connection, since there is no relation between the two parties, the connection could be broken at any time and is called connectionless. Each connection could be represented with a remote port and destination IP address. For TCP connection, however, each connection is maintained with a session controlled by the transport layer. TCP process creates a sub-connection with a random port to the other party automatically without informing the module. Therefore, we need to hash both the local and the remote ports to generate a unique index for each active connection as follows:

Index = port XOR peer_port

This unique identification number is used to match incoming packets if they belong to an existing connection else a new connection is established.

As a peer node may have multiple CoAs, the mapping table is used to direct packets to the respective connections and to look up the relevant CoAs to represent the IP address. As there is no awareness of multiple CoAs, the IP address is swapped and passed to the application and hence the entire process is transparent to the upper layers. It enables ongoing sessions to survive when multiple interfaces in use are brought down.

Data Communication Process

This module is to redirect outgoing packet to the corresponding interface or incoming packet to the upper level. The module is called only to handle outgoing packets. After the outgoing packet is examined and altered, the packet is passed to the Packet Redirecting module. The module is responsible for redirecting packet according to the source and destination addresses set up by the IP Swapping module. Each interface has a different local address and is assigned to a different route table of a given destination address. Each pair *[src_CoA, dst_CoA]* matches with one row in the routing table. Each physical interface has a separate routing table which stores the gateway, next node and the name of the interface. Routing table is updated during switching time when new access point is found and new connection is established. They are stored in different tables since the default Linux OS does not allow usage of multiple interfaces.

The module takes *sk_buff* containing a tuple of data packet and outgoing interface as input. We use modified *ip_route_output_key()* function to help look up the correct entry in the routing table. Thereafter, the *sk_buff* which contains the altered packet is attached to a different route entry and returned to the lower layer of the intended physical interface.

The path availability is decided during the handover when a new network is discovered and routing information is gathered from the router. This router can be the nearest router or a router specified by a centralized router, in this case, the SIP server serves this role. The path could be complicated if each domain does not have direct contact with each other. Router information is propagated in the routers within the path. Here, we assume that the routing information is retrieved from the centralized SIP server which is known to be accessible to all other nodes.

Communication Module

This module is to communicate between the kernel modules and software agent at the user level. This module uses IOCTL commands so that information and command could be passed from the user level. The information includes active interfaces and policy that user desires to use at the Packet Redirecting Module.

The module use IOCTL commands supported by Linux Ubuntu [3] core kernel to build a virtual point in the network flow. Data commands are passed down from the user space in real time and could immediately affect the process. This module also sets up a signaling protocol to inform the user space of the interface changes in the MAC layer. Signal strength, lack of beacons or dropping of bit rates are polled by the intelligent software agent. When the link information is available, the agent shall decide whether to tear down and seek out a new connection. Depending on the hardware design, the MAC layer information requires different drivers to access. The main IOCTL commands are listed in Table I.

Intelligent Software Agent

Intelligent software agent is an application running in user level as a daemon. Its responsibility is to control the number of interfaces in use, creating communication channels with the modules at the kernel level. It searches for

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

new access points to find the best access point for wireless network or scans the wireless network to check if the current WLAN signal quality is lower than a pre-configured threshold. The signal quality of each link is passed from the kernel through the Communication Module} to this agent software. We apply a simple algorithm to rank each wireless network according to a certain level of acceptance. If the current wireless network is lower than the accepted level, the agent sends request to tear down the link and starts searching for new wireless network. The best access point is selected and a new connection is re-established. After this, the kernel modules would be informed of the new connection and a binding update is sent to the SIP server to inform other nodes so that they could update their IP mapping tables.

TABLE I
IOCTL COMMANDS

MSIP_IOC_SEND_INVITE _IOR
MSIP_IOC_PEER_IDENTIFIER_PROCESSING
MSIP_IOC_PRE_MANIPULATE_HOST_IDENTIFIER
MSIP_IOC_PRE_MANIPULATE_PEER_IDENTIFIER
MSIP_UPDATE_IFACELIST
MSIP_IOC_GET
MSIP_IOC_ADD
MSIP_IOC_UPDATE
MSIP_UPDATE_LOCAL_IP

D. Packet Distribution Policy with In-Kernel Re-Order (IKR) Service

For reliable TCP transmission, although there is flow and congestion control at the transport layer, the overhead incurred due to the high number of retransmission packets can degrade the performance. As there are different paths with unknown end-to-end delays and handoffs may occur unpredictably along any route from source to destination, the cumulative latency experienced by a packet will lead to an increase in retransmission requests since the transport layer is unaware of the presence of multiple interfaces. Retransmissions consume valuable bandwidth and degrade link quality, resulting in the receiving ends having to issue window resize advertisement to the immediate routers and the sender to reduce the sending packet's TCP window size. In practice, there is no feasible mechanism to eliminate reordering. Instead we suggest the following to minimize the issue.

As each link has different bitrate, links are utilized when they are up. Hence, we must distribute traffic across the different links based on the ratio in [4] as follows:

$$f_i = \frac{b_i}{b_1 + \ldots + b_N} \quad (1)$$

where b_i is the bandwidth, f_i is the fraction of the number of packets being sent to link ith and N is the number of links. For outgoing packets, as the load on each link is divided proportionally to the actual bandwidth capacity, each link would have a priority. Let us assume there are two links and the bandwidth ratio is 4:1, from Eqn. \ref{eq:ratio} it means that we need to send 4 packets to the faster link and 1 packet to the other. However, if we send packet 1 to the lower link and packets 2, 3, 4, 5 to the faster link, the reception of packet 4 would trigger the fast-retransmission mechanism and the sender must re-transmit packet 1. We implement the In-Kernel Re-Order (IKR) service to solve the out-of-sequence packet. A buffer is used to hold the packets before they enter the network layer. The objective is to re-distribute packets intelligently before sending them out. It selects the packets in an order such that it would not cause fast-retransmission while satisfying the given distribution ratio. In this case, a group of 5 packets are buffered and packet 5 is sent through the slower link while all 4 other packets are sent through the faster link. The delay during the waiting time is ignored since the moment the queue is full, all packets are released at once. The arrivals of packets 2, 3, 4 and 5 are almost at the same time after packet 1's arrival. This approach might cause transmission of multiple packet bursts and subsequent bursts of ACKs. In order to reduce burstiness, we create a delay between every outgoing packet when the buffer is released. On the other hand, if the sender waits for too long to receive the ACK, it could justify that is a network error and reduce the window size. Therefore, in our proposal, the implementation of the "smart selection"' module is done in the application level where packets are collected and released to the TCP level.

For incoming packets, even though we have a "smart selection" mechanism at the sender, for some reasons (such as during the handover), packets are out of order when they arrive at the receiver. Therefore, we implement a module to re-order packets at arrival. Each incoming packet is checked for its sequence number. If it is not expected, it will be queued in the buffer. If the number of out-of-sequence packets is larger than the buffer's capacity, the oldest packet in the queue is removed and the new packet stacks up. During the experiment, it is shown that the buffer is sufficient in reducing the total number of loss packets in TCP transmission.

III. PERFORMANCE RESULTS

A. Testbed

Fig. 3 shows the testbed setup of our proposed framework.

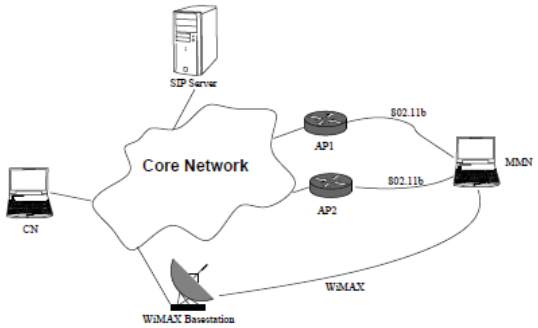

Fig. 3: Illustration of a mobility scenario for multihomed mobile node.

B. Experimental Setup

It consists of a setup with three different links to MMN with the presence of a SIP server in a public IP. CN transfers data to MMN through different links and the MMN moves among the networks (including WiMax, WLAN and UMTS). It means that there are periods when MMN uses all three links or when there is no network access at all. When MMN moves, we run a streaming application using TCP connection to connect the two nodes.

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

We measure the performance of the proposed system based on end-to-end delay and throughput.

C. Experimental Results

The MMN has 3 links, i.e.WiMax, WLAN and UMTS. At first, MMN connects to CN through WiMax which has the longest range. Thereafter, when MMN moves into the WLAN range, it turns on WLAN to connect to CN. Finally, UMTS is used with the other two links to deliver packets to the CN. We used round-robin to distribute the packets over the available links. The results are shown in Fig. 4.

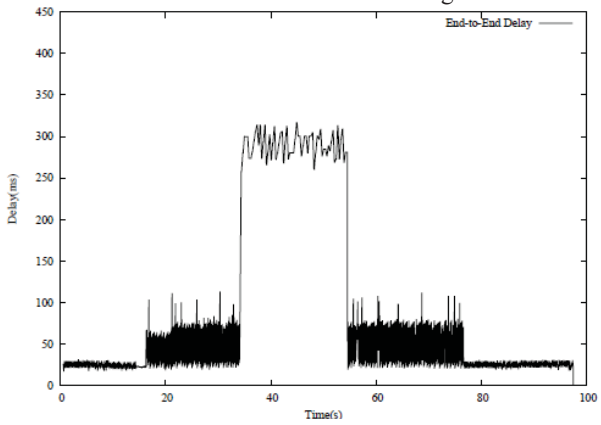

Fig. 4. End-to-end Delay as MMN traverses heterogeneous networks.

At t = 0s, MMN uses only WiMax link, which maintains a reliable and high-speed link. It yields an end-to-end delay of d = 27ms on average. At t = 15s, MMN connects to the WLAN network, using two links at the same time. As we can observe, the end-to-end delay increases to d = 70ms on average. As the WLAN link has lower range and more hops than WiMAX which is only one hop away, the average end-to-end delay is increased. At t = 35s, MMN connects to the UMTS network and it uses all three links at the same time to send packets to CN. It is shown that the average end-to-end delay is increased to d = 300ms. Since UMTS has more hops than WLAN networks, the delay is increased correspondingly. At t = 55s, MMN disconnects from the UMTS network, and goes back to using two links (WiMax and WLAN), resulting in the drop in average end-to-end delay to d = 70 ms. Finally, when MMN moves back to use a single WLAN link, the average delay drops to d = 27ms.

As observed, since all network links have different delays, they affect each other and strangle other packets if we use round-robin algorithm for load-balancing. When a packet using a slow link arrives in the CN, it is queued before the transport layer if it is not in order. The transport layer will send the re-transmission request if the timer runs out. The next time, the packet is sent from the faster link and it arrives in the correct order. Therefore, on average, the delay is increased.

Fig. 5 shows the packet delivery from t = 0s to t = 95s and Fig. 6 zooms in on the handover period which is from t = 10s to t = 20s. As TCP is used, there is no packet loss during the handover. However, Fig. 6 shows that during the handover period, even though there is no loss and all of the packets arrived in the correct order, the interval of arrival has changed and packets are delivered in bundles. This is because before the handover, arriving packets are queued at the receiver end (while the MMN is trying to connect to a different link) and during the handover, these packets are grouped and delivered in bundle at the same time to the application level. This proves that the proposed system could provide seamless connectivity during handover in heterogeneous networks.

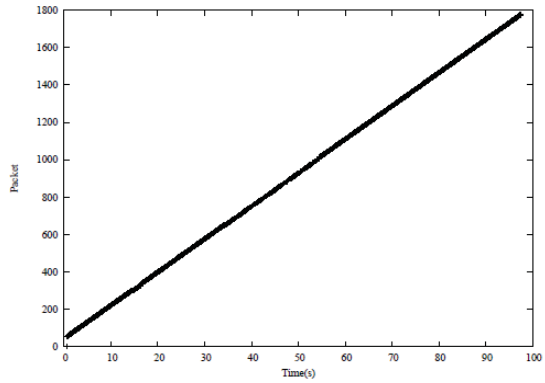

Fig.5. Packet Delivery for entire 100s for TCP

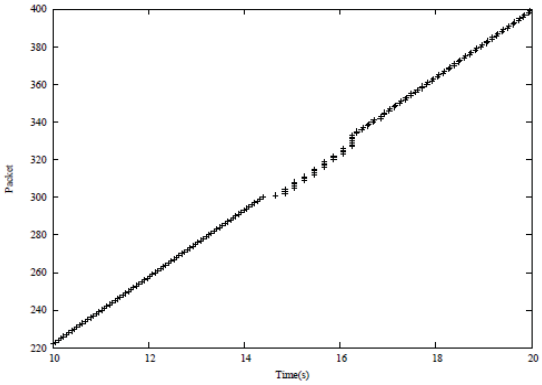

Fig.6. Packet Delivery from t = 10s to t = 20s where handover occurs.

D. Performance Comparison Results

We compare MMS with related approaches here.

Comparison between pTCP and MMS

Compared to pTCP [5] which lies in the transport layer and also supports TCP over multiple interfaces, our solution spans different layers, lying in both the transport and network layers. pTCP uses a virtual TCP session while in fact, spawns the data flow into different TCP pipes, each one of them on a single interface. pTCP's main objective is to achieve aggregated bandwidth using multiple interfaces simultaneously. It is different from our approach in the following manner: (i) Our solution not only achieves aggregated bandwidth but also enables seamless connectivity over heterogeneous handoff. pTCP does not support seamless connection over heterogeneous handoff. (ii) Our system lies in the network layer while pTCP lies in transport layer. (iii) pTCP needs to modify the TCP protocol while as we are at network layer, we do not need to change the TCP protocol and do not need to handle traffic and congestion control.

To compare pTCP and MMS, we create a smart application to measure throughput and packet delivery rate from CN to MMN. The smart application distributes data packets into multiple TCP pipes using MNN's multiple location addresses. The experiment measures data flow

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

during “normal activity period", without any break and measure performance during the switch.

There are two nodes: MMN with three different type of interfaces: WLAN, WiMax and UMTS and CN with one Ethernet connection to the Internet. CN transfers data to MMN via three links. As Ethernet connection has 100MBps bandwidth, it is sufficient enough to feed all MMN's three links (note that WiMax has maximum of 60MBps - the experiment in indoor therefore for safety purpose, it has to be adjusted to a lower level of 6MBps).

Fig. 7 shows that MMS achieves higher throughput than pTCP during single interface (from t = 0s to t = 80s and from t = 160s onwards) and during the time it uses two interfaces. The ideal case is the line presenting the maximum throughput using the single interface and both interfaces. At t = 80s, the MMN switches to use both interfaces and the total throughput is measured on both interfaces. Both pTCP and MMS do not perform as well as the ideal case however, it shows that aggregated bandwidth is achieved for TCP communication.

The experimental results show that MMS is better in achieving aggregate throughput than pTCP without having to modify the TCP protocol.

Evaluation of IKR against Other Packet Scheduling Policies

We compare the different packet scheduling policies, namely, Weighted Round-Robin (WRR), WRR with buffer (WRRB) and IKR and measure the raw throughput at the receiver. WRR is expected to perform worse than other approaches since it does not have any buffer to solve out-of-order packets or intelligent scheduling policy. WRRB could reduce out-of-order packets, however, it introduces waiting time at the queue. IKR's buffering mechanism sorts out packets before distributing them into the different links, and therefore, takes advantage of the multiple interface capability and there is no additional delay incurred during the queuing time. The experimental result is shown in Fig. 8. We use a TCP packet generator to transmit TCP packets of size 1500 bytes each from CN to a MMN with two interfaces (WLAN and WiMax).

At t = 0s, the MMN uses single interface WiMax. All the policies have almost the same performance. At t = 90s, MMN switches to use two interfaces, WiMax and WLAN with the bandwidth ratio of 2:1. WRR has no buffer and no intelligent scheduling policy, therefore, there is packet loss and it results in the termination of the connection after a several retransmissions. With WRRB, the buffer takes effect and re-orders out-of-sequence packets, however, the total throughput drops as the extra delay causes the reduction of the TCP window size.

IKR shows a better performance as it is hidden from the TCP sender and handles out-of-order packets at the receivers. Comparing with WRRB, it has intelligent scheduling policy, therefore, it reduces the occurence of out-of-order packets, in the long run, it shows a better total throughput (520 KBps on average comparing with 400 KBps of WRRB).

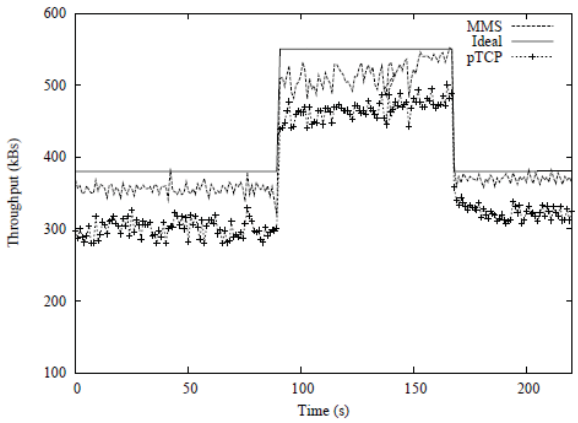

Fig.7. Total Throughput Comparison between MMS and pTCP.

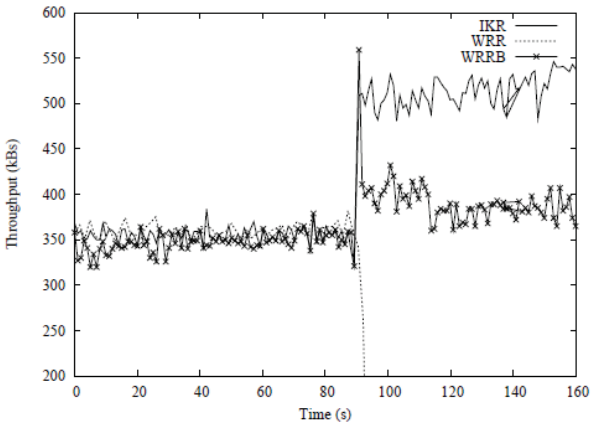

Fig.8. Throughput Comparison between IKR, WRR, BWRR.

IV. Conclusion

TCP is a complicated and reliable protocol with congestion control. It is shown that when applying our proposed system for multihomed devices, it increases the overhead packets and the re-establishment problem or the “slow start" leads to the reduction of window size. The proposed MMS with IKR, has been proven through experiments that it performs slightly better than pTCP in achieving aggregated bandwidth over multiple links without altering the transport layer protocol and can satisfy TCP's complicated requirement to provide seamless connectivity during handover in heterogeneous networks.

References

[1] T. M. Lim, C. K. Yeo, F. Lee, and Q. V. Le, “TMSP: Terminal Mobility Support Protocol,” IEEE Transactions on Mobile Computing, vol. 8, no. 6, pp. 849–863, 2009.

[2] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, “SIP: Session Initiation Protocol,” RFC 2543 (Proposed Standard), Mar. 1999, obsoleted by RFCs 3261, 3262, 3263, 3264, 3265. [Online]. Available: { HYPERLINK "http://www.ietf.org/rfc/rfc2543.txt" }

[3] L. Ubuntu, { HYPERLINK "http://www.videolan.org/." }

[4] D. S. Phatak and T. Goff, “A Novel Mechanism for Data Streaming Across Multiple IP Links for Improving Throughput and Reliability in Mobile Environments.” New York: Proc. IEEE INFOCOM’02, June 2002, pp. 773–781.

[5] H.-Y. Hsieh and R. Sivakumar, “pTCP: An End-to-end Transport Layer Protocol for Striped Connections,” Network Protocols, Proc. 10th IEEE International Conference on, pp. 24–33, 12-15 Nov. 2002.

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)