CHAPTER 1

INTRODUCTION

Three basic ways of getting connected to Internet are Broadband access (by using either a Digital Subscriber Line (DSL) or a cable modem), Wireless Fidelity (Wi-Fi) access, and Dial-up access. The main problems with Broadband access are that it is expensive to deploy and it doesn't reach all areas. Limited coverage area is the main problem with Wi-Fi access. These problems are solved by the Worldwide Interoperability for Microwave Access (WiMAX) technology. WiMAX could replace cable and DSL services (Eklund et al 2002), and thereby provide universal Internet access from anywhere. At the same time, like Wi-Fi, turning the computer ON will automatically provide access to Internet through the closest available WiMAX antenna without any configuration changes.

A WiMAX system consists of two parts, namely, a WiMAX tower, referred to as Base Station (BS), whose coverage area is around 8000 square km, and a WiMAX receiver, termed as Subscriber Station (SS). A WiMAX BS can be connected to the Internet using a high bandwidth wired-connection and also connected to other WiMAX BSs using line-of-sight, microwave-link. The connection to other WiMAX BSs (backhaul) is used to provide coverage to remote rural areas. WiMAX facilitates ubiquitous delivery of wireless broadband service for fixed and/or mobile subscribers. Typical services of WiMAX are data, telecommunications, and IPTV services, Smart Grids, and Metering Services, and are shown in Figure 1.1.
The air-interface of WiMAX technology is based on the IEEE 802.16 standard. The current Mobile WiMAX technology is primarily based on IEEE 802.16e-2005, which supports Scalable Orthogonal Frequency Division Multiple Access (S-OFDMA) air-interface and mobility.

Figure 1.1 Typical applications of WiMAX
1.1 PERFORMANCE OF TCP OVER WiMAX

The Medium Access Control (MAC) layer of IEEE 802.16 supports primarily Point-to-Multipoint (PMP) architecture. MAC is point-to-point and connection-oriented in PMP structure with the BS at one end and an SS at the other end of each connection. Connection is a unidirectional mapping between BS and SS MAC peers. The downlink (DL) channel from the BS to the SS operates on a PMP basis. The BS is the only transmitter operating in this direction and it can transmit without having to coordinate with other stations. SSs share the uplink (UL) channel to communicate with the BS, on a demand basis or on a data-grant allotted to it.

WiMAX’s key focus is to support multiple types of services such as Interactive Gaming, Voice over IP (VoIP), Media Streaming, and Multimedia Broadcasting etc. With these services, Transmission Control Protocol (TCP)-based applications such as Mail, Web-Browsing and File Transfer Protocol (FTP) must also be supported by WiMAX with good performance. These are classified as Best Effort (BE) applications. BS handles the BE traffic on a space available basis.

An SS can transfer the data packets associated with BE traffic only after winning a contention with similar stations for acquiring bandwidth. SSs send a Bandwidth Request (BR) during contention to acquire bandwidth (BW). Moreover, to provide reliability for application, IEEE 802.16 supports Automatic Repeat reQuest (ARQ) at the MAC layer. The basic unit of data managed by ARQ is referred to as a block. After sending a block, ARQ-sender waits for an acknowledgment from the receiver. If the acknowledgment doesn’t arrive or if the wait-interval (in WiMAX, this duration is referred to as ARQ_RETRY_TIMEOUT) for the acknowledgment expires, ARQ retransmits the block. ARQ can carry such retransmissions only
if the \textit{maximum-retries} allows. The delay components incurred by BW request mechanism and ARQ vary with the current channel conditions. Hence, even with ARQ enabled, the wireless-looses are likely to occur. TCP’s congestion-control considers that the packet losses are predominantly an indication of the extent of congestion in the network. The effect of such an assumption by TCP along with the limited bandwidth for Best Effort applications, could lead to its sub-optimal performance in WiMAX environment.

A number of research works have been committed to the issue of TCP’s under-performance over WiMAX, and enormous effort is in progress to alleviate the setback. A hybrid congestion-control mechanism has been proposed by Tay and Noor (2011) to improve the performance of TCP over Mobile WiMAX. This model is developed based on TCP-Vegas and Scalable TCP, and it provides 1.3% to 3% improvement in throughput vis-a-vis the base versions. Anastasopoulos et al (2010) have proposed an Adaptive Modulation and Coding (AMC) scheme for maximizing TCP throughput. Hwang et al (2008) have used the feedback information about the channel state to adaptively control packet size, number of packets to send, and retransmission decision and thereby improves the TCP throughput.

In this work, a scheme is proposed to enhance the performance of TCP, when BE-traffic is carried in the uplink direction. The SSs considered are Fixed Hosts. In the downlink direction, BS has full control over the transmission. But, there is no synchronization among the SSs with BE applications in the uplink direction. Hence, this work focuses on the uplink traffic.
1.2 MOTIVATION AND CONTRIBUTION OF THE THESIS

For applications which require reliable communication service, the dropped blocks by ARQ are to be recovered at the Transport Layer level by TCP, which operates over ARQ. Sayenko et al (2007) have investigated the performance of ARQ over WiMAX and have shown that ARQ mechanism can improve the performance of TCP based applications. However, the mechanisms, TCP and ARQ, do not know the existence of each other and this non-cooperative interaction can cause adverse effect in certain scenarios.

One such scenario is considered with an experiment. A simulation was carried out with a WiMAX BS connected to the set of SSs on one side and a wired sink-node on the other side. A BE connection was established between an SS and the BS, and a TCP application which can inject BE traffic was initiated from the SS. The traffic was directed to the sink-node using the established connection. ARQ was configured to retry a block for a maximum of 12 times at an interval of 20 ms, if it doesn’t receive the acknowledgment from the BS. The wired-delay between BS and sink-node was set as 50 ms. Figure 1.2 shows the instances at which the segments were sent by the application and the MAC.

At 16.16534 s, cwnd was 2. At 16.165336021 s, TCP agent generated segments numbered 1779 and 1883. At 16.178632447 s, MAC sent both of them as a single Protocol Data Unit (PDU), which was dropped due to error. Before MAC could recover the lost blocks by way of retransmission, TCP timed-out and retransmitted 1779 at 16.395336021 s. Hence, two copies of the same TCP segment resided in the ARQ queue at the same time. At 16.415311 s, one of the blocks of 1779 got dropped by MAC. Hence, one copy of 1779 reached the receiver. The same scenario got repeated for the segment 2099.
Figure 1.2 Segments Generated by TCP Agent and MAC—Instances

But due to an identical reason, two copies of the segment 2619 were received. This occurred as there is no coordination between TCP and ARQ; i.e., TCP has timed-out while ARQ retransmission was still in progress. The same scenario in the DL direction was exposed by Mehta and Vaidya (1998) in a wireless network, and the authors have suggested that delaying the third duplicate acknowledgment could avoid such early TCP retransmissions. Dawkins et al (2001) have recommended that this scheme can be used only after devising a mechanism to calculate the appropriate amount of delay for an arbitrary network topology. Another possible scenario that could happen is when TCP is waiting for an acknowledgment, while ARQ has dropped the packet already. This could lead to unnecessary delay.

Fairhurst and Wood (2002) have recommended the usage of a backoff retransmission delay which increases with the number of attempts,
while designing ARQ for shared-channels. ARQ supported by IEEE 802.16 uses a static value for the said parameter, `ARQ_RETRY_TIMEOUT`. Moreover, while TCP’s retransmission policy continuously adjusts Retransmission-TimeOut (RTO) based on the current Round-Trip-Time (RTT), ARQ uses the static parameters set during the connection establishment procedure. This non-supportiveness among the protocols hinders end-to-end performance, and this is the prime motive behind studying the impact of the static parameter `ARQ_RETRY_TIMEOUT` on the end-to-end performance of TCP. As per IEEE 802.16-2009 (2009), this parameter accounts for the processing delays associated with the transmitter and the receiver, and any other delay relevant to the system. Hence, from the inference that its value may have considerable impact on RTT, it is chosen for the purpose of study.

The impact of `ARQ_RETRY_TIMEOUT` on the performance of TCP was investigated for different network-loads and error-rates using simulations. The result showed that the TCP throughput is highly influenced by this parameter. Hence, the work has concluded that the parameter needs to be adjusted dynamically based on the current network-load and error-rate.

Similarly, there are other ARQ-related parameters whose values need to be optimized. Tykhomyrov et al (2008) have studied the impact of ARQ feedback interval on TCP. On receiving a data packet from the ARQ-sender, the receiver acknowledges the receipt of the packet. The maximum duration the receiver waits before sending an acknowledgment is termed as feedback-interval. The authors have recommended that ARQ feedback message be used frequently. The authors have also mentioned that if ARQ and Hybrid ARQ (HARQ) mechanisms are enabled at the same time, it is efficient to delay sending the ARQ feedbacks so as to avoid double retransmission. The authors have considered only downlink traffic. In the
uplink direction, delay involved in the BR mechanism should also be considered. Latkoski and Popovski (2009) have also studied the impact of various ARQ parameters on ARQ block mean transmission time, considering the delay incurred by a block before and after its first transmission. To decrease the mean ARQ block delay, and to improve MAC layer throughput, they have recommended that the ARQ parameters should be set according to the traffic behavior as well as the physical layer parameters.

Hence, to enhance the performance of TCP, either the various ARQ parameters are to be adjusted continuously or a scheme is required irrespective of the ARQ parameters chosen. The proposed work in this thesis has followed the latter. ARQ drops a block after a few retries. The packet would have taken many retries, because of wireless-errors or too many SSs are contending for BW. If the cause of the drop could be identified, then TCP can be informed not to invoke congestion recovery for the drops due to wireless-errors. There are lots of congestion notification schemes available such as the one reported by Xu et al (2004), which allow TCP to differentiate wireless-errors and congestion-losses that occur in the network. The reported scheme uses a flag, referred to as Congestion Warning (CW), set by the on-the-way-router, to differentiate the congestion-losses from wireless-losses. Routers set this flag when the queue-length crosses a threshold. But, an appropriate threshold for queue-length can be fixed up, only if the nature of the traffic is known ahead.

This work has proposed a scheme, named, TCP-WLAware, to enhance the performance of UL TCP irrespective of the ARQ parameters chosen. TCP-WLAware is an Explicit Loss Notification scheme, which informs TCP regarding the loss events that are not due to congestion, and to identify such losses, it exploits the number of BRs sent by the SS. TCP decouples them from congestion-control and retransmits those segments.
Here, the congestion is confined to wireless-hop only. This scheme allows the SS to identify the non-congestion drops on its own and preserves the end-to-end semantics of the TCP connection.

If ARQ is enabled, the blocks would have got retransmitted many times before getting discarded. The rationale for the drop cannot be assumed to be due to error or congestion as all the retries may not have resulted out of the same reason. Hence, this work checks if the wireless-hop is congested at the time of the drop as the retransmission is going to occur immediately.

As the BS serves the BE applications on a space available basis, with no minimum guarantee, the stations with this kind of traffic will all try to contend for the bandwidth. Colliding stations retry sending the requests for BW after an interval. To spread the contending stations, Truncated Binary Exponential Backoff (TBEB) algorithm is used, and this is a measure taken to avoid congestion. As the spread out stations will contend again and may collide with new stations which are ready at that time, the exponential growth cannot be carried out for a longer time and hence, stops after some retries. If the collisions occur frequently, BRs will not reach the BS, and hence BS would not allocate BW for BE traffic even if the BW is available. Hence, the number of retries made by an SS before getting a BW allotment is an indication of the intensity of contenders, which is a measure of congestion.

Normally, an SS resorts to contention only for a fixed number of retries and drops the PDU. In these cases, ARQ will try to recover those PDU’s. Even then, if a drop has occurred, and if the moving average of the number of retries made by an SS crosses a threshold, then the drop is considered as one due to congestion. The contention procedure maintains a contention window, which directs the SS to decide as to when to send the BR. The proposed work adjusts the contention window dynamically based on the
number of active BE connections, and thereby fixes the threshold for differentiating congestion-losses.

The drops that are not associated with congestion need to be informed to TCP. TCP-WLAware, which is a modified version of TCP-NewReno, just does that. TCP-NewReno is modified to receive the feedbacks triggered by MAC and retransmit the drops by decoupling them from congestion-control while preserving the basic Fast-Retransmit and Fast-Recovery mechanisms. When there are multiple drops within a window (maintained by TCP), TCP-NewReno is an ideal choice and hence, this variant of TCP is chosen as the base. The performance of TCP-WLAware is compared with TCP-NewReno in terms of throughput and is shown that TCP-WLAware outperforms TCP-NewReno.

1.3 OUTLINE OF THE DISSERTATION

This section outlines the structure of the thesis. This thesis comprises of eight chapters.

The second chapter illustrates the IEEE 802.16 reference model. Furthermore, the chapter describes the services of MAC layer and explains the operation of Physical layer variant, OFDMA.

The third chapter provides an overview of the features of TCP. The chapter also elucidates the various variants of TCP in detail. The metrics used to evaluate the performance of TCP are also explained. The issues involved in using TCP over wireless networks are also discussed.

The fourth chapter summarizes the works that are available in the literature, which fall within the domain of the proposed work. This chapter has classified the literature into three groups and has discussed the merits and
demerits of the schemes. The three groups considered are (i) schemes which explore the performance of TCP over WiMAX and enhance the performance of TCP by circumventing a limiting factor (ii) schemes that separate congestion-losses from wireless-losses and thereby enhancing TCP (iii) schemes that adjust the values of the contention resolution parameters.

The fifth chapter discusses the issues related to having TCP over ARQ. The modifications carried out so as to extend WiMAX Simulator Release 2.6 (WiMAX patch, 2009) for NS-2 (http://www.isi.edu/nsnam/ns) are presented. The performance of TCP in terms of throughput for various ARQ_RETRY_TIMEOUT at different loads and error-rates are obtained and compared.

The sixth chapter gives an overview of Code Division Multiple Access (CDMA) based contention mechanism. The chapter also presents the analytical calculation for finding out the initial Start/End for the backoff-window, and the effectiveness of the analytical model is verified through simulations. Then, the chapter also depicts as to how the adaptive window could be used to identify the drops associated with congestion. The simulation results are also discussed.

The seventh chapter proposes an Explicit Loss Notification scheme to notify the loss to TCP, named TCP-WLAware. The modified TCP-NewReno algorithm is also outlined and discussed. The performance of TCP-WLAware is compared with TCP-NewReno through simulations.

The eighth chapter provides concluding remarks and discusses the direction for future research.