# Protocols to Enhance TCP Performance on Mobile Systems

**Pilar Díaz[1], David Airlie[2], Fernando Casadevall[1], John Nelson[2]**

[1]Universitat Politècnica de Catalunya, Barcelona, Spain

[2]University of Limerick, Limerick, Ireland

***Abstract***

*Applications used in Internet run over a TCP protocol that was designed for a wired network, where the main reason for packet loss is congestion on the network. Procedures used to overcome congestion problems are inadequate when packets are lost on the network due to non-congestion reasons, as occurs in wireless networks. This paper lists some solutions already proposed in the literature to alleviate the effects of non-congestion related losses on TCP performance over wireless LANs and proposes some modifications to adopt them for UMTS. The use of IP header compression techniques is considered in this context to obtain some speed benefits. Simulation models to assess TCP performance in different scenarios are also described.*

## 1. Introduction

Many of the applications currently used in the Internet world run over a TCP protocol that was designed taking into account the impairments of the wired network, where the main reason for packet loss is that of congestion on the network. The main procedures used to overcome congestion problems in TCP networks are slow-start and congestion avoidance. However these concepts, which work very well on wired networks, become problems when packets are lost on the network due to non-congestion reasons, which are more likely to occur within wireless networks. With standard TCP schemes non-congestion related packet loss results in an unnecessary reduction in the end-to-end throughput and sub-optimal performance.

Therefore, when these applications are use in a wireless environment, which posses important limitations on the TCP protocol performance, it is necessary to improve the protocol in order to cope with these drawbacks. However, in order to maintain complete compatibility with the already implemented applications, it is desirable to achieve this goal without changing the existing TCP implementation in the fixed network. In this context, it is evident that the only modifications allowed will be addressed to the control of the wireless link. That is, the only elements of the access network, where some modifications can be introduced to cope with the above-mentioned drawbacks, are the Radio Access System (RAS) and Mobile Terminals (MT).

This paper describes first the different protocol categories proposed in the literature to enhance the poor performance of traditional TCP when operated over wireless links. These protocols are proposed in the context of Wireless LAN, the architecture of which is substantially different from UMTS. After this, the simulation models taken into account to obtain the performance of these protocols in a UMTS context is described. Also some ideas on IP header compression techniques, which offer some speed benefits to both TCP and non-TCP throughput, are presented, and finally some conclusions are drawn at the end of the paper.

## 2. Schemes for improving TCP performance in wireless environments

Recently, several schemes have been proposed in the literature to alleviate the effects of non-congestion-related losses on TCP performance over networks that have wireless or similar high-loss links. Many of these schemes can be classified into three basic groups based on their fundamental philosophy: end-to-end protocols, split-connection protocols and link-layer protocols.

In the end-to-end approach, TCP is enhanced with selective acknowledgements (SACK) or explicit congestion notification (ECN) to perform better over wireless links [Jacob88]. End-to-end approaches require changes to the host on the fixed network, although changes such as SACK have started to become standard on many TCP implementations.

In the split-connection approach, the TCP connection between a fixed and a mobile host is split into two separate connections at the base station: one TCP connection between the fixed host and the base station, and the other between the base station and the mobile host [Wang].

In the link-level approach, the wireless link implements a retransmission protocol coupled in some cases with a forward error correction scheme at the data-link level.

There is another protocol, known as the snoop protocol [Bal95], which cannot be classified into any of the three categories mentioned above. It is not an end-to-end protocol, and neither is a split-protocol. Although it is sometimes classified in the literature as a link-layer protocol, in our opinion it is not because it does not work at a data link level.

All these techniques have been proposed in the light of Wireless LAN scenarios, that is, in the absence of a specific Radio Resource Management (RRM) as in UMTS. With an RRM, the radio functionality could contribute much more to the UMTS-Internet architecture. In strict sense, this is the case of the handover procedure, which is under the control of the radio part. It is connection oriented, and consequently no packets should in principle be lost under this concept. Unfortunately this is not always the case because handover may fail, and some of the principles included in the WLAN techniques can be retained.

Additionally, the TCP/IP layers rely on the lower radio layers. Therefore, the above-mentioned protocols should be located in the appropriate entity of the RAS to succeed in improving the performance of the TCP/IP protocols. This is especially important when different types of handover are considered, as will happen in UMTS.

Table 1 shows the main advantages and disadvantages of the three protocol categories in the Internet/UMTS context.

| **Approach type** | **Advantages** | **Disadvantages** |
|---|---|---|
| *End to End* | SACK is becoming standard | Not all stacks may support |
| *Link Layer* | Needs no support from end TCP stacks, only link layer | Bad link layer approach can cause degradation |
| *Split Stack* | Isolate the network from the radio part | Violates end-to-end semantics, data is processed more often than end-to-end |

**Table 1.** Advantages/disadvantages of the different approaches in a mobile context.

Taking into account the advantages/disadvantages of the different alternatives, the best scheme is probably a modified link-layer approach that uses a link-layer that supports in-order delivery and has some knowledge of TCP, followed by a mixed SACK/split-stack approach or SACK/snoop approach. In any case, a more detailed study of these protocols in a UMTS context including handover procedures is being conducted in order to determine accurately the real impact of the different strategies on the performance of the proposed mechanism.

## 3. Simulation description

This section describes the simulation scenarios taken into account to assess the performance of the protocols introduced to enhance TCP performance in mobile environments. The aim is to assess the performance of such protocols considering different options of network architecture for UMTS, and analyse the behaviour of TCP in a more realistic environment in comparison to the studies done so far in Wireless-LAN environments.

Among the different protocols proposed in the literature, the study of TCP performance is being carried out considering the snoop protocol and a split-connection approach as the Mobile-End-Transport Protocol (METP) [Wang] for comparison purposes. An end-to-end approach is discarded since this solution would imply changes at both ends of the TCP connection. A pure link-layer approach is not considered as such but in combination with other protocols. In particular, the link-layer considered in this study is the one proposed for the UTRA Wideband CDMA.

### *3.1 Simulation scenarios*

The reference packet network architecture considered in the simulations is illustrated in Figure 1, where the Radio Access Network, composed by Nodes B and Radio Network Controllers (RNC), is pending of GPRS nodes, i.e., SGSN and GGSN. Three different scenarios are being considered: IP down to GGSN, IP down to IGSN (Integrated GPRS Subsystem Node), and IP down to RNC.

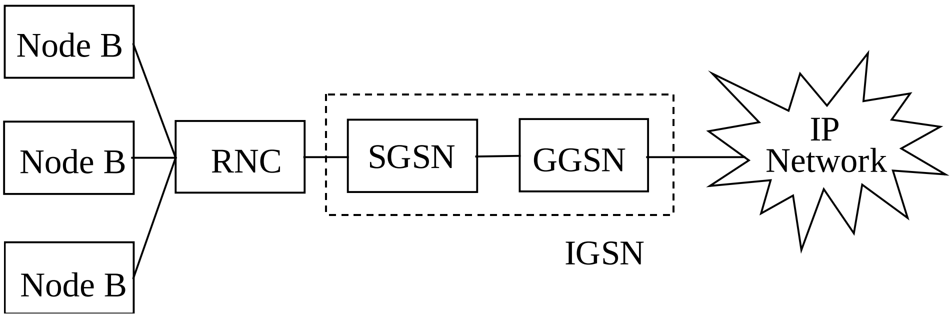

**Figure 1.** Packet network architecture for the mobile system.

The next subsections describe the simulation model considered for each one of those scenarios.

*Scenarios 1 and 2*

Scenarios 1 and 2, in which IP goes down to GGSN or IGSN respectively, are equivalent from a transport level viewpoint, so the simulation model to assess TCP enhancements is similar for both scenarios. This model is represented in Figure 2 for scenario 1, where L1 models the radio channel behaviour in addition to some ARQ mechanism, L2 is a dedicated link representing the tunneling between SGSN and GGSN and L3 is an IP-based link. Since IP goes down to GGSN, the snoop or METP agent is located at this entity as illustrated in the figure.

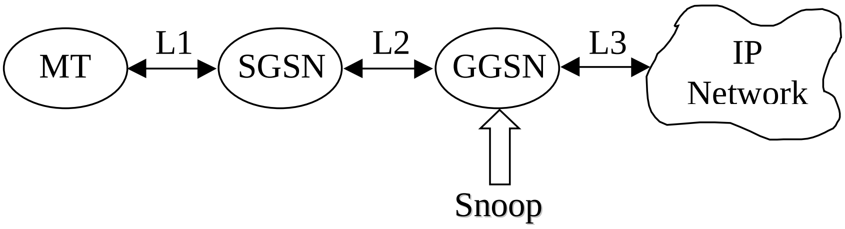

**Figure 2.** Simulation model for IP down to GGSN.

*Scenario 3*

The simulation model for scenario 3, in which IP goes down to RNC, is represented in Figure 3. L1 models the radio channel behaviour in addition to some ARQ mechanism, and L2 is an IP-based link. Since IP goes down to RNC, the snoop or METP agent is located there as illustrated in the figure.

This model is very similar to the one described in the previous subsection. The main difference arises when mobility is taken into account. If a handover process involves the node where the snoop (or

MEPT) agent resides in, context information must be shifted from the old node to the new one. When the node is RNC, as in scenario 3, instead of GGSN, as in scenario 1, this shift takes place more often. This results in a similar model but with different handover rates from the viewpoint of the snoop agent.

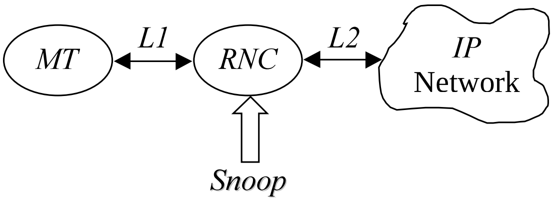

**Figure 3.** Simulation model for IP down to RNC.

### *3.2 Results*

To compare the performance introduced by the snoop protocol and the MEPT, values of throughput and delay (mean and standard deviation) and TCP retransmission rates will be obtained from the simulations. Results will be available in a final RAINBOW deliverable.

## 4. IP Header Compression

Although not directly related to enhancing TCP performance, the use of IP header compression over the final radio-based IP link can bring speed benefits to both TCP and non-TCP throughput. IP Header compression works only over a single point-to-point link with a combination compressor/decompressor module located at each end of the link. On low to medium speed links, IP header compression provides the following advantages.

- Improved interactive response time
- Allows use of small packets for bulk data with good line efficiency
- Allows use of small packets for delay sensitive low data rate traffic
- Decrease in header overheads
- Decrease in packet loss on lossy links

### *4.1 Basic Compression System*

The compression system is based around packet groupings known as packet streams. A packet stream is a sequence of packets that have similar headers and share a context. For TCP connections and many UDP streams, all packets share common source addresses, destination addresses and ports in the header so these are easily identified. The compressor contains a packet grouping mechanism to make sure that the packets that are grouped together have as little difference between them as possible, or else the scheme becomes inefficient. An identified packet stream is assigned a context identifier, CID, which denotes to the de-compressor which stream any incoming packets belong to.

For an individual packet stream the compression scheme is initiated by the transmission of a full header including the CID. This header is taken by the compressor and decompressor as the basis for further packet compression. The constant fields in the context are marked and these will no longer be transmitted over the link, also fields with minor differences are identified, so that only deltas have to be transmitted.

During compression, if a constant field changes a full header must be transmitted by the compressor with a new CID in order to update the context at the decompressor. If the full header is corrupted by the link and is discarded, the compressor and decompressor lose synchronisation and further compressed headers will be decompressed incorrectly. When using header compression with IPv6, it is recommended that strong checksums and perhaps some form of FEC/ARQ is used at the link-layer to avoid packet loss on the point-to-point link. This ensures that the synchronisation is not lost. Also reordering of packets on the link can cause problems with synchronisation so if possible this should be avoided or a mechanism should be employed to detect lost synchronisation and resynchronise the streams. These mechanisms are quite different for TCP and non-TCP streams and are dealt with in the next section.

The compression system also uses a slow-start mechanism to allow quicker recovery from the loss of a full context-changing header. Full headers are transmitted periodically with an exponentially increasing period, this avoids having to exchange messages between the compressor and decompressor as these exchanges can be costly on wireless links in terms of resources.

The compression method, specified in [Degerm], uses four link-level packet types in addition to the standard IPv4 and IPv6 packet types. These are

- FULL_HEADER - this indicates a packet with an uncompressed header, including a CID, and if it is non-TCP, a generation count. It is used to initialise the compression scheme or update the context.
- COMPRESSED_NON_TCP - this indicates a non-TCP packet with a compressed header, which consists of a CID to identify the context, a generation count, which is used to detect an inconsistent context, and the randomly changing fields of the header.
- COMPRESSED_TCP - this indicates a compressed TCP packet, containing a CID to identify the context, a flag octet indicating the changed fields and the changed fields encoded as a delta versus the previous value of the field.
- COMPRESSED_TCP_NODELTA - this indicates a packet with a compressed TCP header where all fields that are normally transmitted as deltas are instead sent as they are. This is only transmitted in response to a header request from the decompressor.

Regular IP packet types are used whenever the compression scheme decides not to compress a packet header. There is also an additional special case packet used to speed up the repair of TCP streams that the decompressor can send to the compressor

- CONTEXT_STATE - this communicates a list of TCP CIDs for which synchronisation has been lost. It requires no IP header as it is only transmitted over the point-to-point link.

### *4.2 Dealing with packet loss in TCP streams*

TCP packets are compressed by sending the difference between the current packet and the previous packet in the stream, therefore the loss of a packet can cause subsequent decompression to be incorrect. When a compressed TCP header is lost, the sequence numbers in subsequent decompressed headers will be off by an amount k, where k is the size of the lost segment. The TCP receiver should take care of these headers as the TCP checksum reliably catches ‘off-by-k’ errors. TCP’s standard retransmission scheme will cause the discarded segment to be resent, and the compressor checks the headers for these re-transmits and sends a full header to resynchronise the compressor and decompressor. There are also some mechanisms that can be used to speed up the context repair for TCP streams, which are advantageous in wireless environments.

### *4.3 Dealing with packet loss in non-TCP streams*

When a UDP packet or other non-TCP packet is incorrectly decompressed they are not as well protected by checksums. UDP checksums only cover the payload, UDP header and source/destination address fields in the main IP header. This means many parts of IPv6 headers are not covered by this checksum. In order to avoid incorrect decompression, each version of the context for non-TCP packets contains a generation field, which is carried by the full headers at startup and context refresh. This field is carried by compressed headers and, if the generation field differs, the decompressor either has to discard the packet or wait for a full header to update the context. As there is no delta-coding done for non-TCP streams, compressed headers do not change the context, so the loss of a compressed header does not invalidate subsequent packets.

### *4.4 Application of IP Header Compression to UMTS/IP and Effects of Handover*

IP Header compression is used only over a single IP link, this means that wherever IP is terminated in the network, the compression system must be located, along with corresponding functionality in the UE. The compression system is only affected by handover that moves the entry point to the compressor. For scenarios involving IP to the IGSN (or equivalent) this means only in intra-IGSN, or inter-domain cases, for the RNC level, inter-RNC handover and above and for the TDD mode inter-BTS handover will affect the compression system. The current contexts stored in the decompressor module would need to be transferred to the new decompressor, or a message would need to be sent to the compressor in the UE to restart, and the compressor in the network would need to restart by transmitting new full headers and context information.

### *4.5 Simulation Structure*

The simulations in this area look at three scenarios, with the compressor located at three different points, with compression turned on and off. The simulation work is being carried out in OPNET using a point-to-point link with radio-like characteristics and estimated handover rates. This work had just gotten underway at the time of writing, and results are hoped to be available in a final RAINBOW deliverable.

## 5. Conclusions

Many of the applications used in Internet run over a TCP protocol designed to be used within a wired network, where the main reason for packet loss is congestion on the network. Procedures used to overcome congestion problems are inadequate when packets are lost on the network due to non-congestion reasons, as occurs in wireless networks.

This paper presents some solutions already proposed in the literature to alleviate the effects of non-congestion related losses on TCP performance over wireless LANs, as the snoop protocol, an end-to-end approach, an split-stack approach or a link-layer solution. If adopted for UMTS, the best scheme is probably a modified link-layer approach followed by a mixed SACK/split-stack approach or SACK/snoop approach. A more detailed study of these protocols in a UMTS context including handover procedures is being carried out in OPNET. The use of IP header compression techniques is also considered in this context to obtain some speed benefits.

## References

[Bal95] H. Balakrishnan, S. Seshan, R.K. Katz, "Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks", Wireless Network, Vol.1, pp.469-481, 1995.

[Degerm] Degermark, Nordgren, Pink, “IP Header Compression”, RFC-2507, Internet RFC, February 1999.

[Jacob88] V. Jacobson, R.T. Braden, “TCP Extensions for Long Delay Paths”, RFC-1072, Internet RFC, October 1988.

[Wang] K.Y. Wang, S.K. Tripathi, “Mobile-End Transport Protocol: An Alternative to TCP/IP over Wireless Links”, Infocom 98.