UNIVERSITÀ DI ROMA "TOR VERGATA"


Degree of Philosophy Doctor in Space Systems and Technologies

# Enhancing Transport Layer in Satellite Systems: Design and Development of an Emulation Platform

**by**

**Cesare Roseti**

Supervisor
Prof. Michele Luglio

Coordinator
Prof. Giancarlo Cardarilli

2007

# Summary

The rapid growth of Internet applications (i.e., email, file transfer, remote access, web browsing, e-learning, e-banking, video-conferencing, etc.) is causing progressive congestion of telecommunication networks. Moreover, new classes of services including telemedicine, information dissemination, emergency support, disaster response and public information, may require broadband and untethered connectivity. Therefore, wireless access is gaining growing importance due to its intrinsic flexibility and ubiquity.

In such a context, the use of satellite systems represents an attractive option they can offer high transmission capacity and large coverage. In fact, a satellite system can guarantee broadband access to the global network in sparsely populated areas where deploying a terrestrial infrastructure remains unappealing or in regions where the deployment of terrestrial facilities remains impractical. Moreover, satellites can be integrated to terrestrial wireless (WiFi, UMTS, etc.) for enhancing mobile access to the Internet by both pedestrians and vehicles while it is often the only option for aeronautical and maritime users.

Most of broadband satellite systems are deployed at traditional geostationary orbit (GEO). In this scenario, IP-based transfers must face some peculiar challenges. In particular, performance of Internet applications, being based on the "Transmission Control Protocol" (TCP) at the transport layer, is strongly affected by several factors introduced by satellite systems (i.e., large latency, losses not due to congestion , high link asymmetry, etc.). Such factors make standard TCP inefficient over satellite links, resulting in poor performance.

The analysis and the design of an optimized transport layer for system including a satellite segment has been the object of the author's Ph.D. program in the

frame of the $19^{th}$ cycle of the Ph.D. course of "Space System and Technologies" at the University of Rome "Tor Vergata". A preliminary activity, including a stage period at the Computer Science Department of University of California Los Angeles (UCLA), has been carried out by the author in the frame of his Master's thesis. Then, author has delved into such a research area over three years within National, EU and ESA founded projects and a stage period spent at both and TEC-SWS Department of the European Space Agency (ESA). Short or medium periods have been spent in external Universities as a visiting researcher to collaborate in joint activities: [12/9/2004-16/8/2004] University of Siena, [3/11/2004-6/11/2004] University of Bologna, [20/3/2005-8/4/2005] Aristotle University of Thessaloniki (GR).

This thesis has a twofold aim: present all the achievements coming from the author's research activity and, based on the collected experience, propose the design of a emulation platform optimized to test TCP/IP protocols over a likely satellite environment.

The work is organized in six chapters. Chapter 1 deals with basic notions concerning satellite systems. In particular, network architectures, services, standard and security aspects are briefly discussed. Chapter 2 summarizes main mechanisms regulating the standard TCP protocol. It aims at highlighting the main TCP protocol characteristics, essential to understand the subsequent chapters which focus on issues of TCP in satellite-based environments. Chapter 3 offers a comprehensive analysis of TCP performance over satellite links. Along with the identification of both the main limitation and the proposed solutions to improve TCP performance, a vast gamut of analytical, simulation and experimentation results, coming from the author's research activity, are presented. Chapter 4 collects all the outcomes of the author's activity, focusing on the enhancement of the transport layer for Satellite-based Internet applications. Chapter 5 describes the main characteristics of an emulation platform, named "Satellite Network Emulation Platform" (SNEP), designed by author to reproduce the variable characteristics of a satellite network (DVB-RCS like) through an active emulation approach. Finally, conclusions are drawn in Chapter 6.

Most of the outcomes of the author's research activities have been collected and discussed in the following publications:

*JOURNAL*

1. M. Luglio, **C. Roseti**, M. Gerla, *The Impact of Efficient Flow Control and OS Features on TCP Performance*, Invited paper on ASSI Satellite Communication Letter (Sat-Comm Letter), $9^{th}$ edition, special issue on Multimedia Satellite Communication, vol. III, n. 1, June 2004, pp. 1-9.

2. C.E. Palazzi,**C. Roseti**, M. Luglio, M.Y. Sanadidi, J. Stepanek,*Enhancing Transport Layer Capability in HAPS-Satellite Integrated Architecture*, Wireless Personal Communications (Kluwer Academic Publisher), vol. 32, n. 3-4, February 2005, pp. 339-356.

3. P. Chini, G. Giambene, D. Bartolini, M. Luglio, **C. Roseti**, *Dynamic resource allocation based on a TCP-MAC cross-layer approach for DVB-RCS satellite networks*, International Journal of Satellite Communications and Networking, Vol. 24, n. 5, Set/Oct. 2006, pp. 367-385.

4. C. Caini, R. Firrincieli, M. Marchese, T. De Cola, M. Luglio,**C. Roseti**, N. Celandroni, F Portontì, *Transport Layer Protocols and Architectures for Advanced Satellite Networks*, International Journal of Satellite Communications and Networking, Vol. 25, n. 1, January 2007, pp. 1-26.

*CONFERENCE & WORKSHOP*

1. C. E. Palazzi, **C. Roseti**, M. Luglio, M. Gerla, M. Y. Sanadidi, J. Stepanek, *Satellite coverage in urban areas using Unmanned Airborne Vehicles (UAVs)*, VTC spring 2004, 17-20 May 2004, Milan, Italy.

2. M. Luglio, **C. Roseti**, M. Gerla, *TCP Performance over Satellite in Case of Multiple Sessions per Links using Efficient Flow Control and Real OS*, 10th Ka and Broadband Communications Conference, Sept. 30th - Oct 2nd 2004, Vicenza, Italy, pp. 253-260.

3. G. Giambene, M. Luglio, **C. Roseti**, *A cross-layer mechanism for efficient management of TCP-based traffic*, in Proceedings of 23rd International Communication Satellite Systems Conference (ICSSC) and 11th Ka and Broadband Communications Conference, Rome, Italy, Sep. 2005.

4. P. Chini, G. Giambene, D. Bartolini, M. Luglio, **C. Roseti**, *Dynamic Resource Allocation based on a TCP-MAC Cross-Layer Approach for Interactive Satellite Networks*, in Proceedings International Workshop on Satellite and Space Communications (IWSSC), Siena, Italy, Sep. 2005.

5. P. Chini, G. Giambene, D. Bartolini, M. Luglio and **C. Roseti**, *Cross-layer Management of Radio Resources in an Interactive DVB-RCS-based Satellite Network*, in Proceedings $20^{th}$ International Symposium on Computer and Information Sciences (ISCIS), pp. 124-135, Istanbul, Turkey, Oct. 2005.

6. **C. Roseti**, G. Theodoridis, M. Luglio and N. Pavlidou, TCP driven CAC scheme for HAPS and Satellite integrated scenario, in Proceedings $1^{st}$ International Workshop on High Altitude Platform Systems (WHAPS), Athens, Greece, Sep. 2005.

7. **C. Roseti**, E. Kristiansen, *TCP behaviour in a DVB-RCS environment*, in Proceedings $24^{th}$ AIAA International Communications Satellite Systems Conference (ICSSC), San Diego, Jun. 2006.

8. M. Luglio and **C. Roseti**, *Network Security and Performance Evaluation of ML-IPsec over Satellite Networks*, in Proceedings $12^{th}$ Ka and Broadband Communications Conference, Naples, Sep. 2006.

9. M. Luglio, F. Mazzenga, **C. Roseti**, *Analysis and Performance evaluation of integrated HAPS/Satellite architectures*, First COST 297-HAPCOS Workshop, York, UK 26-27 October 2006.

10. D. Fanni, M. Luglio, **C. Roseti**, F. Zampognaro, *A Cross-Layer based handover for TCP applications*, $65^{th}$ IEEE Vehicular Technology Conference (VTC), Dublin, Ireland, Apr. 2007.

11. M. Luglio, N. Pavlidou, **C. Roseti**, G. Theodoridis, *CAC-TCP cross-layer interaction in a HAPS-satellite integrated scenario*,$65^{th}$ IEEE Vehicular Technology Conference (VTC), Dublin, Ireland, Apr. 2007.

12. E. Duca, V. Carrozzo, **C. Roseti**, *Performance Evaluation of a Hybrid Satellite Network Based on High-Altitude-Platforms*, 2007 IEEE Aerospace Conference, Mar. 2007.

*BOOK*

1. Ed. G. Giambene, *Resource Management in Satellite Networks*, Ed. Springer. **C. Roseti**, chapter editor of Chapter 9: Resource Management and Transport Layer.[1]

[1] book in press

# Acknowledgements

Un sentito ringraziamento ai miei genitori, che, con il loro incrollabile sostegno morale ed economico, mi hanno permesso di raggiungere questo traguardo.

Un ringraziamento speciale a Giusi per il suo sostegno continuo e sincero.

I would like to thank my supervisor, Prof. Michele Luglio, for the guidance he provided throughout this work.

I would also like to thank Mr. Erling Kristiansen for all the fruitful and stimulating discussions.

Finally, a special thanks to myself for my application.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Satellite systems: architectures, services and standards

## 1.1 Introduction

The ever growing use of Internet applications (e.g., email, file transfer, remote access and web browsing) is causing progressive congestion of telecommunication networks. Moreover, new classes of services including telemedicine, information dissemination, emergency support, disaster response and public information, requires broadband and untethered connectivity. Therefore, wireless access is gaining growing importance due to its intrinsic flexibility and ubiquity. For these reasons, satellite systems presents an attractive option since they offer high transmission capacity and large coverage [1][2]. In fact, a satellite can guarantee broadband access to the global network in sparsely populated areas where deploying a terrestrial infrastructure remains unappealing or in regions where the deployment of terrestrial facilities remains impractical. Therefore, satellite extends mobile access to the Internet, in areas not covered by terrestrial wireless networks (WiFi, UMTS, etc.), for both pedestrians and vehicles, while it is often the only option for aeronautical and maritime users. Most broadband satellite systems are deployed at traditional Geostationary Earth Orbit (GEO).

The setting up of a communication link through satellites requires to carefully take into account the peculiar characteristics of the system, first of all the huge

distance between the terminal on ground and the spacecraft. Since the beginning of the satellite telecommunication age all the related problems have been approached and mostly solved or at least mitigated allowing to achieve ever easier feasibility and better degree of service along with cost reduction.

This chapter addresses the basic concepts concerning satellite telecommunications providing as much as possible a complete, even though brief, overview on both architectures and standards that are envisaged in the research activity, reported in the rest of this thesis.

## 1.2 The Satellite Networks

A satellite network is composed of two segments: the ground segment and the space segment.

The ground segment is mainly composed of the gateway stations or Hub stations, the user terminals and the Network Operation Center (NOC). The NOC includes the Network Control Center (NCC) in charge of configuration management, capacity/bandwidth management, acquisition/synchronization control, performance management, alarm management, security management, billing, and accounting. The space segment includes the satellite equipment and the communication payload.

Satellite systems can be classified on the basis of orbit: GEO, LEO, MEO, HEO or hybrid [3][4].

A GEO satellite is located at an altitude of approximately 36,000 km. GEO satellites can have a large footprint (approximately 1/3 on the world surface), and, in theory, just 3 satellites are sufficient to cover the entire world surface. On the other hand, at this altitude high latency presents a serious disadvantage (about 500 ms round trip delay).

LEO (Low Earth Orbit) satellites orbit around the world at an altitude varying between several hundreds of km and a few thousand km. In this case, the deployment of a constellation of several satellites is needed to achieve contiguous coverage, and efficient handover mechanisms must be implemented to allow service continuity. The propagation delay ranges from several ms up to 80 ms.

The characteristics of MEO (Medium Earth Orbit) satellites fall between GEO

and LEO. Both LEO and MEO satellite systems have the advantage of lower propagation delays when compared with GEO system, but they are more complex to manage since both require continuous hand-off, dynamic routing algorithms and tracking mechanisms.

HEO (Highly Elliptical Orbit) satellites are located in elliptical orbits in which the Earth assumes the position of one of the two foci over planes inclined at 63.4° with respect to the equatorial plane. According to the Kepler's second law, satellites in HEO orbits move slowly around the apogee looking "almost" stationary. As a consequence, HEO satellites cover high latitude regions very effectively with a good (high) elevation angle. Finally, hybrid systems use a combination of more than one type of orbit.

In this thesis, when not otherwise specified, networks based on GEO satellites are considered.

## 1.3 Satellite Architectures

Most of the satellite communication systems are called "bent-pipe" to indicate that satellite simply acts as a signal repeater between two ground stations. Therefore, there is no data processing on board the satellite. On the other hand, some other satellite systems implement on-board processing (OBP) capabilities, which include demodulation, modulation, decoding, coding, packet switching, routing, etc.

An example of "bent-pipe" satellite system architecture is the Direct Broadcast Satellite (DBS), which supports only broadcast services (i.e. TV broadcasting). Instead, to support a full duplex Internet access, a possible solution is based on the use of dial-up telephone connections or by other terrestrial networks. Other solutions provide a return channel via satellite. This is the case of the DVB-RCS. In case of OBP, satellite networks appear like switching packet networks, where satellite switches packets among transmitting beams and receiving beams. Thus, in this case the satellite footprint is composed of multiple spot beams.

Two kinds of topology are usually supported by a satellite network: star and mesh. The former consists of a central hub station and a set of user terminals. The hub acts as the star center through which all the traffic transits; as a consequence

every pair of user terminals are physically interconnected through the hub thus needing a double hop on the satellite (large delay). In the latter topology the user terminals can be directly interconnected among each other, needing only one hop on the satellite, thus minimizing the use of the bandwidth and the delay (one hop). Hybrid configuration, with a subset of stations connected in full mesh mode and the others in star mode, is another possibility, which has trade-offs in performance.

Figure 1.1 shows a typical satellite star architecture, where all the ground terminals are directly connected to the Hub station via the transparent "bent-pipe" satellite. The Hub/NCC station, acting as star center performs all the switching/routing tasks. The terminals belonging to the satellite network communicate by a double-hop link: terminal 1-Hub and Hub-terminal 2.

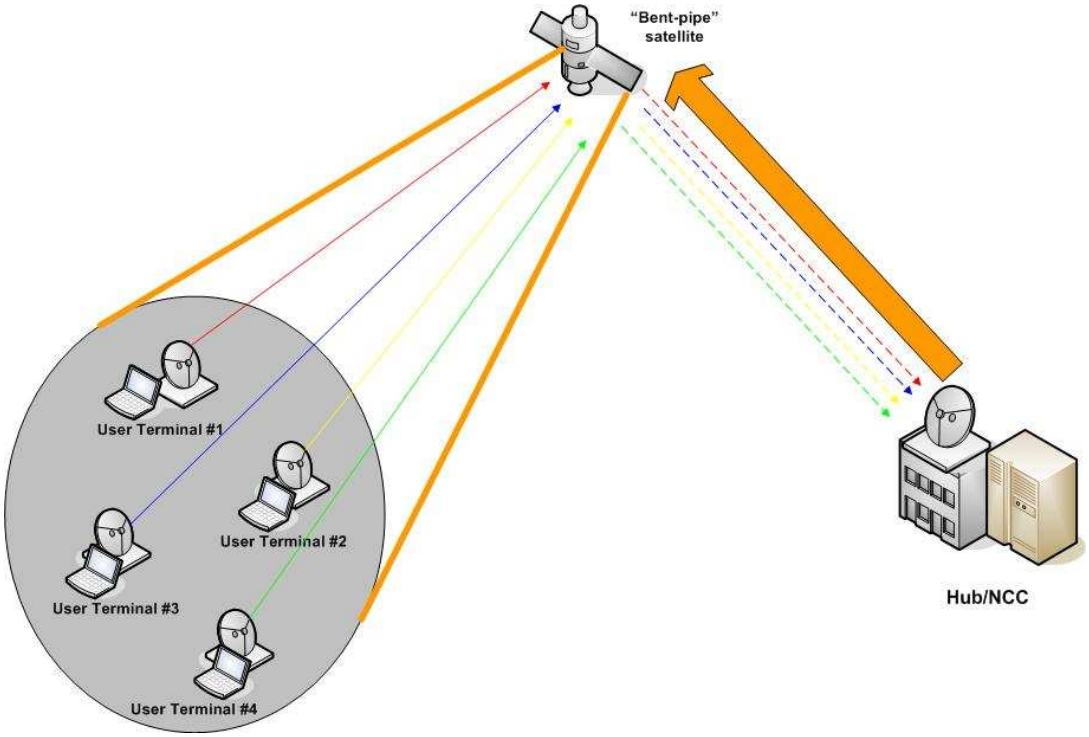


Figure 1.1. Satellite star architecture

Figure 1.2 depicts a satellite mesh architecture, where the OBP satellite payload allows to cross-connect terminals through a multi-spot coverage. In particular, the on-board switching capability provides a direct connectivity among different coverage areas by a single-hop link.

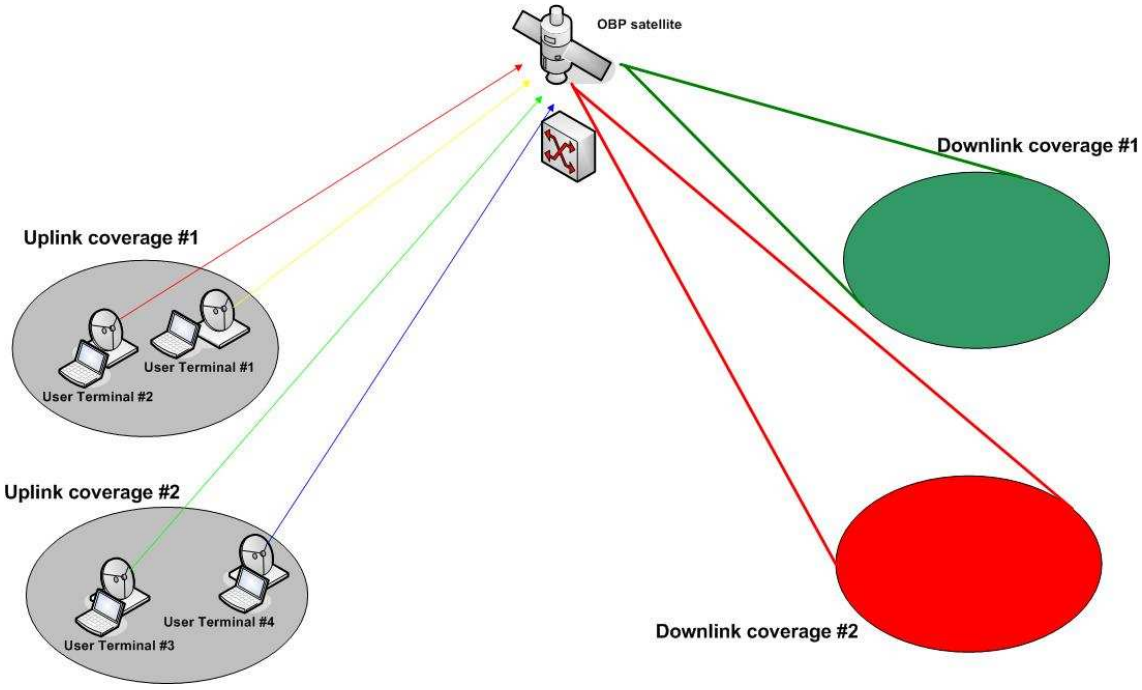


Figure 1.2. Satellite mesh architecture

## 1.4 Services & Applications

There is not a real limitation to provide any service through a satellite network. Every kind of service and relative application can be implemented with satellite but taking into account specific performance. In fact, the propagation delay affects performance of real-time interactive service (e.g., voice, videoconference) but, on the other hand, the suitability to multicast operation makes satellite systems more efficient and cost effective. In particular, both real-time and store and forward services, both fixed and mobile services, both narrow band and wideband services, unicast, multicast, and broadcast services are currently provided.

Satellite systems represent a very attractive solution for a large set of applications including content delivery, data/voice communications and Internet over Satellite. In some case, satellite is the most efficient solution (i.e. provision of broadcast services in a large geographic scale), while in other cases is recommended especially for strategic or economic matters (i.e. Internet access or communications in isolated and sparsely populated areas). The main service/application classes can be classified

as in the Table 1.1.

| **Service** | **Application** |
|---|---|
| Messaging | E-mail<br>Chat<br>Paging |
| On-line Info Retrieval | PC Networking<br>Data Base Access<br>Remote Login |
| Telephony | Voice<br>Sms |
| Video-communication | Videophone<br>Videoconference |
| Video-information | Tele-medicine<br>Tele-education |
| Broadcasting | Analog TV<br>Digital TV<br>HD-TV<br>Analog Radio<br>Digital Radio |

Table 1.1. Services & Applications provided by satellite systems

### 1.4.1 Television & Video Services

The main service provided via satellite over the years has been surely TV broadcasting. In fact, United States, Japan and Europe have largely adopted satellites for distribution of TV programs. In Europe, the Digital Video Broadcasting Project (DVB) worked on designing global standards for the delivery of the digital television [5]. In this frame, DVB-S system for digital satellite broadcasting was developed in 1993 [6]. In general, satellite networks are far more suitable than the terrestrial networks for data broadcasting (i.e. web-casting, network news, TV programs, etc) thanks their ability to cover large areas optimizing radio resources. Furthermore, video distribution applications can usually tolerate delay (but not excessive jitter).

### 1.4.2 Interactive Services

Interactive services concern both voice and data communications. Voice communications are particularly affected by the propagation delay imposed by the GEO satellite systems, which degrades the quality of the interactive communication. On the other hand, most of the current data communications are based on TCP/IP protocol stack, and among these a large part requires reliable data transmission and runs TCP as transport protocol. When using TCP over satellite networks, performance is influenced negatively by 3 factors:

- Long link delay means long round trip time (RTT) for the mechanisms controlling sending rate and congestion management.
- TCP acts defensively to variation in available bandwidth, potentially leading to under-use of communication resources.
- TCP assumes all packet losses are due to congestion. A packet lost due to errors on the link induces TCP to reduce its sending rate unnecessarily, thus wasting resources and/or reducing performance.

The issue of TCP performance over satellite is still a big challenge in satellite network research and it is strongly coped along this thesis.

## 1.5 Standards

Satellite systems are mostly based on proprietary standards, both in terms of hardware and services. This section focuses on Digital Video Broadcasting (DVB) standards, largely referred in the author's research activity.

### 1.5.1 DVB

The DVB system [5] specified by the European Broadcast Union (EBU) is based on the cell-oriented packet transmission system defined by ISO/IEC 13818-1 MPEG-2 Systems Standard [7]. MPEG-2 Systems Standard provides the means of multiplexing several types of multimedia information into one Transport Stream (TS)

that can be transmitted over a variety of transmission media. Figure 1.3 shows the MPEG-2 reference system.

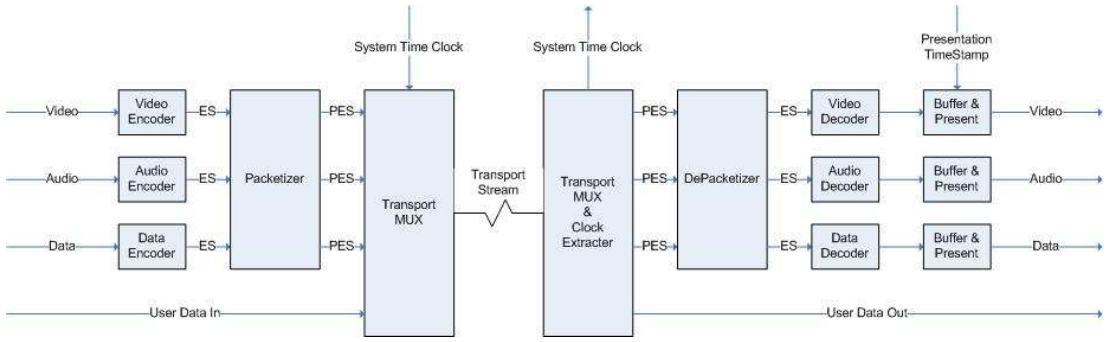


Figure 1.3. MPEG-2 reference system

The basic component of MPEG-2 System is known as "Elementary Stream" (ES). A programme (i.e. TV programme or DVB track) contains a combination of Elementary Streams, typically one for video, one for audio and one for metadata. ESs are generated by video, audio and data encoders. Each ES is an input for an MPEG-2 processor (i.e. video compressor or data formatted), which assembles data into a stream of "Packetised Elementary Stream" (PES). A PES packet is a variable sized block (up to 65536 bytes) including a 6-bytes protocol header. A PES is usually organized to contain an integer number of ES units.

A PES packet structure is shown into Figure 1.4.

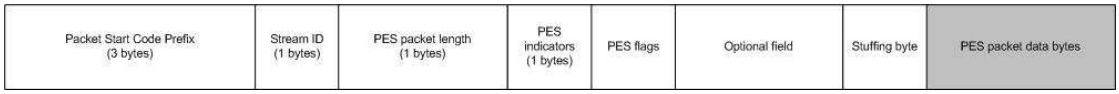


Figure 1.4. PES format

The PES header starts with a 3 byte start code, followed by a 1 byte stream ID and a 2 byte length field respectively. The next field contains the "PES indicators" that provide further information about the stream to assist the decoder at the receiver. Such indicators are:

- *Fixed bits* (2 bytes);
- *PES Scrambling Control* (2 bytes) which defines whether scrambling is used and, in case, the chosen scrambling method;

- *PES Priority* (1 byte) which indicates priority of the current PES packet;
- *Data Alignment Indicator* (1 byte) which indicates if the payload starts with a video or audio start code;
- *Copyright information* (1 byte) which indicates if the payload is copyright protected;
- *Original or Copy* (1byte) which indicates if the PES belongs to the original ES.

A flags field completes the PES header. Each flag defines one of the following optional fields, which (if present) are inserted before the start of the PES payload:

- *Presentation Time Stamp* (PTS) and possibly a *Decode Time Stamp* (DTS) which are used to synchronize a set of elementary streams as well as to control the rate they are replayed by the receiver;
- *Elementary Stream Clock Reference* (ESCR);
- *Elementary Stream rate* which indicates the rate at which the ES was encoded;
- *Trick Mode* which indicates a not "normal" ES (i.e. after DSM-CC has signaled a replay);
- *Copyright Information* which is set to 1 to indicate a copyright ES;
- *CRC* which may be used to monitor errors in the previous PES packet;
- *PES Extension Information* which may be used to support MPEG-1 streams.

The information in the PES header is, in general, independent on the transmission method. Finally, the PES packet payload includes the ES data, and (if needed) a stuffing byte is placed before the payload.

The sequence of variable-length PES composes a Program Stream (PS) that assumes the use of error-free transmission channels (storage). In addition to the PS, MPEG-2 System Standard specifies another multiple stream: the Transport Stream (TS). TS assumes the use of transmission channels (such as in broadcasting and

telecommunications) where errors occur and consists of relatively short fixed length TS packets. The MPEG-2 specification makes possible to convert PS streams into TS streams. Consequential, a PES packet can be divided into segments that, in turn, can be placed in the data section of TS packets. Each TS packet has a fixed length of 188 bytes. This length is highly compatible with ATM cell lengths and has been chosen because of its applicability to efficient and robust error-correction coding. The most important information in the TS header is the packet identifier (PID) that can identify about 8000 types of packets in a 13-bit field. The format of the transport packet is shown in Figure 1.5.

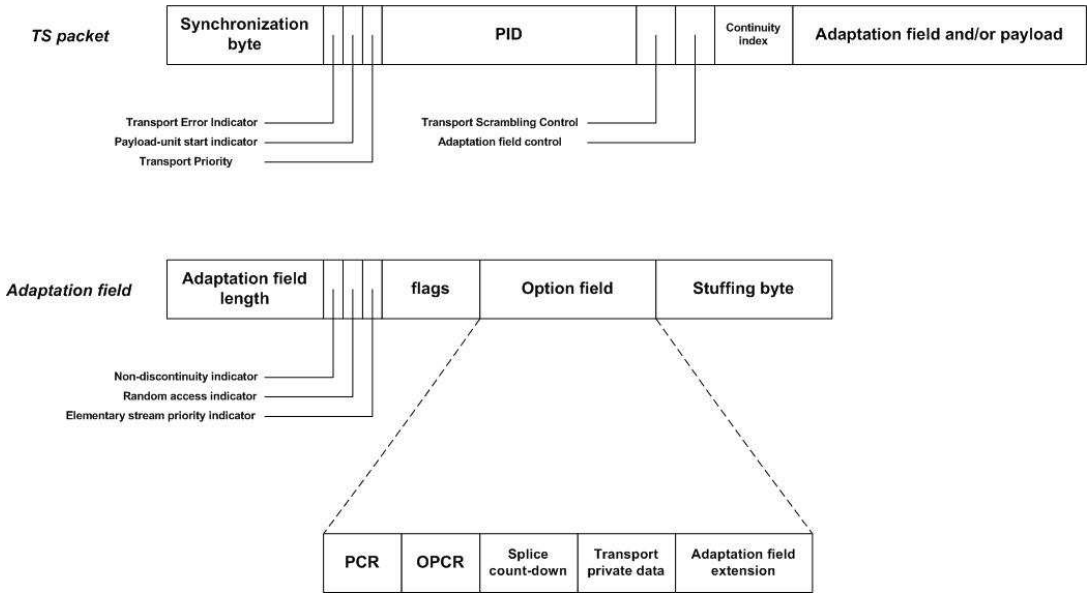


Figure 1.5. TS format

Each MPEG-2 TS packet envisages 184 bytes of payload data prefixed by a 4 bytes header. The header has the following fields:

- *Synchronization Byte* which is a well-known sequence of bits (0x47=0100 0111);
- Three *bit-flags* indicating how the payload should be processed,
  - The first flag indicates a transport error;
  - The second flag indicates the start of a payload;

  - The third flag indicates transport priority bit;

- A 13-bit *Packet Identifier* (PID) which is used to uniquely identify the stream which the packet belongs to. Some PID values are predefined and are used to indicate streams of control information;
- A 2-bit *Transport Scrambling Control* field which is used by conditional access procedures for the encryption of some TS packets;
- A 2-bit *adaptation field control* indicating the type of the TS payload configuration,
  - 01 - no adaptation field, payload only;
  - 10 - adaptation field only;
  - 11 - adaptation field followed by payload;
  - 00 - RESERVED for future use;
- *Continuity Index* (4 bits) which is used to control continuity in the Transport Stream (the value is cyclical incremented).

The adaptation field can contain various types of information such as splice countdown used to indicate Programme Clock Reference (PCR) and edit points. It is also used as stuffing; that is, when the final section of a PES packet is placed in a TS-packet payload, dummy data is added to ensure a fixed-length TS packet. In general a PES packet spans several TS packets. When a PES packet is starting the "payload unit start indicator" bit is set to 1, which means that the first byte of the TS payload contains the first byte of the PES packet header. Only one PES packet can start in a single TS packet. As seen above, TS header also contains the PID so that the receiver can accept or reject PES packets at a high level without leading to a hard processing. An example of how PES packets are placed in the data section of TS packets is shown in Figure 1.6.

#### 1.5.1.1 DVB-S & DVB-S2

DVB-S has been conceived for primary and secondary distribution (Fixed Satellite Service, FSS) and Broadcast Satellite Service (BSS), at present mostly operated in

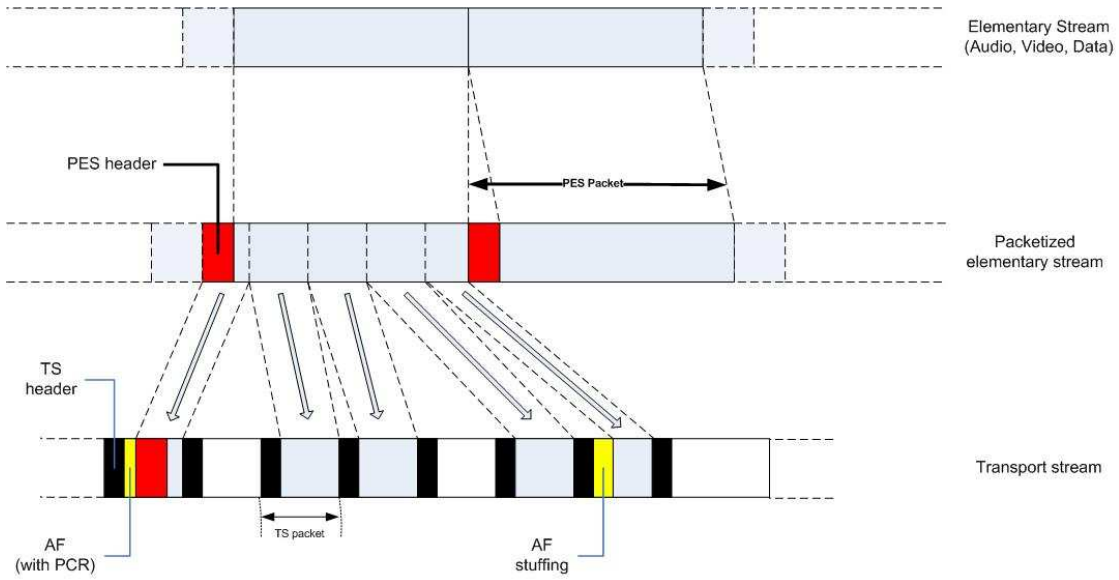


Figure 1.6. Mapping of Transport Stream, PES stream and elementary stream

the Ku band (11/14 GHz) [6]. This standard is designed to provide a direct reception from the satellite (Direct-To-Home, DTH) for both a single user with an integrated receiver-decoder and a collective access. In particular, the DVB-S goal is to adapt the baseband signal from the output of the MPEG-2 transport multiplexer to the satellite channel characteristics. Since DTH services via satellite are particularly affected by power limitations, robustness against noise and interference has been taken as the main design objective.

The satellite channel adaptation concerns the types of modulation and coding schemes to be adopted to meet the target quality of the signal. The quality target is the provision of a "Quasi Error Free" (QEF) channel, corresponding to a Bit Error Rate (BER) ranging from $10^{-10}$ to $10^{-11}$ at the input of the MPEG-2 demultiplexer). The processes involved in this adaptation are transport multiplex adaptation and randomization for energy dispersal, outer coding (i.e., Reed-Solomon), convolutional interleaving, inner coding (i.e., punctured convolutional code), base-band shaping, and modulation (Figure 1.7). The randomization is applied to make the RF spectrum approximate a white noise process. The channel coding is applied through three steps: outer code Reed-Solomon (204, 188, T = 8), convolutional interleaving with the aim to randomize burst errors, and convolutional coding with a variable

code rate (1/2, 2/3, 3/4, 5/6, and 7/8). Gray-coded QPSK modulation with absolute mapping (no differential coding) is adopted. Base-band signal is square-root-raised-cosine-filtered with a roll-off factor of 0.35. The DVB approach offers a good degree of flexibility. A 38 Mbit/s data flux could contain eight standard definition television (SDTV) programs, four enhanced definition television programs (EDTV), or one high definition television (HDTV) program, all with associated multichannel audio and ancillary data services. The maximum data rate for a data stream is typically about 38 Mbit/s accommodated in a 33-MHz satellite transponder.

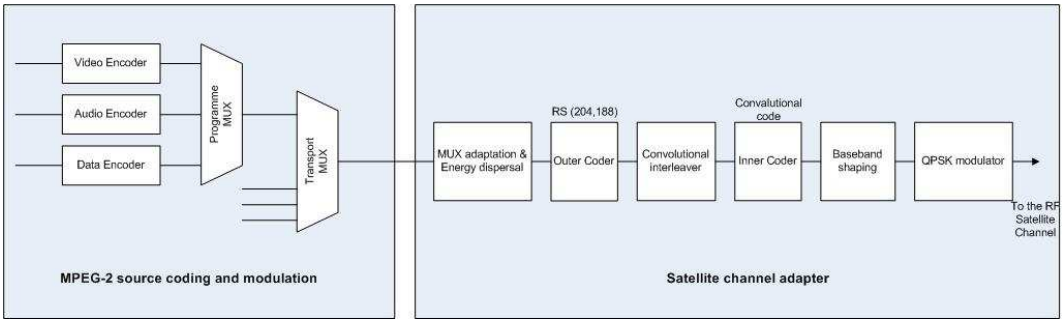


Figure 1.7. Functional block diagram of the DVB-S system

DVB-S2 [8] is the second generation DVB. In the new scheme, which modifies the satellite channel adaptation block only, adaptive modulation and coding (ACM) have been introduced. In particular, low density parity check codes (LDPC) have been selected with Bose-Chaudhuri-Hochquenghem (BCH) code-block length 64,800-bit (or 16,200-bit) periodic structures that facilitate parallel decoding with about 50 decoding iterations. Available coding rates are 1/2, 3/5, 2/3, 3/4, 4/5, 5/6, 8/9, and 9/10. Moreover, five modulation formats, all optimized to operate over nonlinear transponders, have been selected: BPSK (1 bit/s/Hz), QPSK (2 bit/s/Hz), 8PSK (3 bit/s/Hz), 16APSK (4 bit/s/Hz), and 32APSK (5 bit/s/Hz). Then, three roll-off factors are allowed: 0.35, 0.25, and 0.20 (DVB-S: only 0.35). With the new standard, an optimization of spectral resources in the order of 30-40

### 1.5.2 IP over DVB

Within MPEG-2 TS, it is possible to carry defined data containers in addition to the audio and video [9]. These data containers can be used to realize new data services

or to carry IP datagrams.

The DVB specification for data broadcasting [5] defines three different ways of inserting data into the MPEG-2 transport stream. According to the functional block diagram shown in Figure 1.8, there are three possible entry points for IP data traffic over MPEG-2:

**(i)** Data packets can be encapsulated and carried inside the PES packets intended for video and audio streams. This method is referred as Data Streaming.

**(ii)** Data packets can be carried inside the section packets defined for system internal tables in DSM-CC. This method is called Multiprotocol Encapsulation (MPE).

**(iii)** An adaptation layer protocol can segment data packets directly into a sequence of cells. This method is called Data Piping.

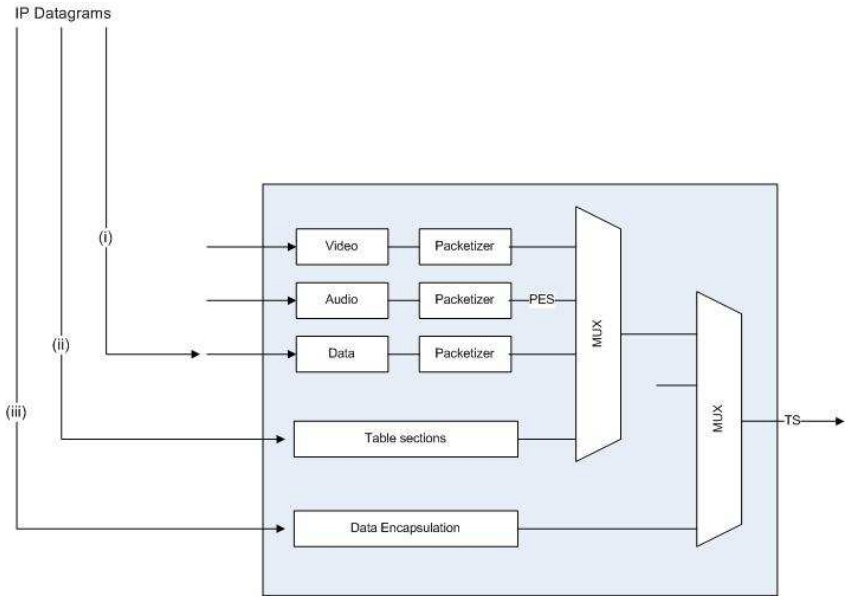


Figure 1.8. Entry points fot IP traffic over MPEG-2

Methods (i) and (ii) are offered by MPEG-2 as standard. In both cases, the segmentation function is performed automatically. The third method of transferring datagrams (data piping) needs an explicit segmentation and reassembly layer.

All three methods involve a certain overhead, which stems from the header fields and the fact that IP datagrams usually do not come in multiples of 184 bytes (TS packet size). The total overhead for a transmission depends on the packet length distribution and the encapsulation method selected. The observed overhead for MPE is typically between 13 and 15 per cent [10].

### 1.5.3 DVB-RCS

The success of the DVB standard due to its capability to transmit huge quantity of data in a very flexible way, with consequent availability of technology at low cost, extends its field of applicability beyond the transmission of only data information. In addition, the compatibility with IP protocol allows the processing of the IP packet in the DVB stream. As a consequence, first the use of DVB for one-way systems (DVB IP with return channel on terrestrial networks) has been introduced. They are based on the use of the DVB flow in the forward link (data received through the same equipment utilized for digital TV), while the return link is exploited through terrestrial networks (i.e., ISDN, PSTN).

A two-way standard based on DVB was introduced with return channel on satellite (DVB-RCS), which allows bidirectional communications up to 2 Mbit/s in the return link and shares the bandwidth on a Multi Frequency Time Division Multiple Access (MF-TDMA) discipline [11][12].

#### 1.5.3.1 Resource management

A certain number of adjacent temporal slots (with the same frequency) are arranged into a frame, with a fixed structure. A frame is composed of different kinds of slots. Data must be placed into a Traffic burst format (TRF) slot. The overall frequency channels available and their time slotting at frame level is announced from the HUB via the forward link on a Service Information (SI) table, as shown in Figure 1.9.

This information is acquired by terminals in the setup phase, and it is important for the transmission synchronization.

The coordination is in charge to the Network Control Center (NCC) which prepares a Burst Time Plan (BTP) so that every slot has an “owner”, and a terminal

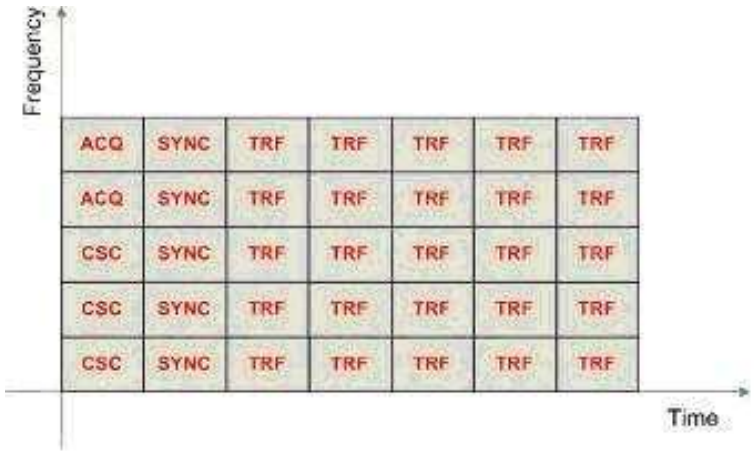


Figure 1.9. Frame Time slotting

is only authorized to transmit in its own slots. How the BTP is scheduled is related to the DAMA scheme discussed in next section.

Several frames are grouped into a super-frame, which contains all frames with the same BTP structure. DVB-RCS supports the use of ATM or optionally MPEG-2 as transport layer above the physical layer.

The BER required for return channel is of about $10^{-7}$ taking into account the mandatory FEC coding. Turbo coding is an option; thus, systems using that FEC coding either would have a better error performance or would need smaller antennas for the same performance.

#### 1.5.3.2 Bandwidth on Demand (BoD)

DVB-RCS supports a Demand Assignment Multiple Access (DAMA) scheme in the return link [11][12]: the NCC assigns bandwidth as response to explicit requests (BoD) coming from RCSTs by announcing in the forward link, for every frame, the Terminal Burst Time Plane (TBTP) on the basis of both the request category and the Service Level Agreements (SLAs) between operator and customers. According to the DVB-RCS standard, RCSTs send their resource requests to the NCC. The NCC, after performing the access control routines, assigns to each RCST a group of bursts (i.e. the related time slots), each of them characterized by a frequency, a bandwidth, a start time and a duration in the form of the TBTP. The DVB-RCS

standard allows fixed or dynamic time slot allocations according to five different schemes.

1. *Continuous Rate Assignment* (CRA): CRA is a fixed and static allocation of resources after an initial set-up phase with a negotiation between the RCST and the NCC. With CRA, a given number of time slots are continuously assigned to an RCST every super-frame until that RCST sends the assignment release message.

2. *Rate-Based Dynamic Capacity* (RBDC): RBDC is rate capacity dynamically requested by the RCST to the NCC. Each request is absolute, corresponding to the full rate currently being requested, overrides all previous RBDC requests from the same RCST, and shall be subject to a maximum rate limit directly negotiated between the RCST and the NCC.

3. *Volume-Based Dynamic Capacity* (VBDC): VBDC is volume capacity dynamically requested by the RCST to the NCC. These requests are cumulative (i.e. each request shall add to all previous ones from the same RCST), indicating a total number of traffic slots that are needed to transmit all data currently available in the MAC queue. These slots can be shared among several superframes. MAC parameters are the minimum and the maximum VBDC capacity that can be assigned to an RCST. VBDC is the default mode in DVB-RCS.

4. *Absolute Volume-Based Dynamic Capacity* (AVBDC): AVBDC is volume capacity dynamically requested by the RCST to the NCC; AVBDC requests are absolute (i.e. each request replaces the previous ones from the same RCST) and indicate a total number of traffic slots that can be assigned in several super-frames. AVBDC is used for loss recovery in the VBDC mode.

5. *Free Capacity Assignment* (FCA): FCA is volume capacity that shall be assigned to RCSTs from capacity that would be otherwise unused by CRA, RBDC and (A)VBDC methods. The term 'free' in FCA refers to 'spare' system capacity. Such a resource assignment shall be automatic, not involving any request from the RCST. Capacity assigned in this category is intended as a bonus capacity that can be used to reduce delays for any traffic category.

Every super-frame, slots are allocated in an orderly manner based on both the request category (i.e., priority order: CRA → RBDC → (A)VBDC → FCA) and the Service Level Agreements (SLAs) between operator and customers.

## 1.6 Satellite Security

The level of security, in a broad sense, that a satellite system can ensure relies both upon its characteristics and upon the number of expedients that it is possible to implement [13]. The following peculiar characteristics of satellite systems assume a meaningful importance in providing security.

- The coverage area, typically quite extended, allows the network to serve a large number of users even in remote, isolated and impervious locations, where usually terrestrial infrastructures are not deployed (either for economic or feasibility reasons) and where typically some office that requires isolation for confidentiality can be located.

- The broadcast peculiarity of the satellite signal represents an advantage for key distribution to a large set of users spread over the territory. Moreover, this characteristic does not imply the capability to receive information for anybody. In fact, to receive just the electromagnetic signal from just one satellite a highly directional antenna is usually necessary and as a consequence accurate pointing is needed; then, even though the signal is received, to decode information may not be so easy requiring compliance with the adopted air interface. When omnidirectional antenna are adopted on the ground terminal, no pointing is necessary but the decoding information aspect still applies.

- The high latency limits performance of security protocols implementation introducing significant inefficiency if frequent key distribution updating is required, as for example in the multicast scenario [14].

- Facilities and equipment are well localized and not spread over the territory and thus extremely easy to be garrisoned. As a consequence, the unauthorized access to telecommunication resources is practically impossible.

- The practical absence of an open, diffused standard, which actually represents a weakness from market penetration point of view, represents an additional factor of security. In fact, to be able to not only receive the electromagnetic signal but to decode information, it is necessary to be authenticated by the NCC. The authentication process cannot be performed if the utilized standard is not known.

- Both the space segment and terrestrial segment equipment are extremely reliable increasing the confidence for the satellite user about security in terms of service continuity. This feature encourages the use of satellites for secure services that, in most of the cases dealing with, for example, human safety and disaster relief, need also the highest degree of survivability.

# Chapter 2

# TCP protocol

The TCP/IP protocol suite, derived from OSI reference model, is a set of protocols that allow terminals from different networks and with different software/hardware characteristics to communicate each other. Such protocols are developed in layers, with each layer being responsible for a different aspect of communications. TCP/IP is commonly characterized as a 4-layer structure [15]. This chapter deals with the features and the standard implementations of the transmission control protocol (TCP) [16], which is one of the transport layer protocols.

## 2.1 From OSI to the Internet architecture

The network architecture known as Open System Interconnection (OSI) and defined by the International Standards Organisation (ISO) represents a theoretical reference in designing of the real network architectures (i.e., Internet). In particular, the OSI architecture is based on seven layers, as shown in Figure 2.1. This layering on more levels makes the design more simple and facilitates the realization of eventual modifications. In fact, every layer takes care of specific aspects of the communication. Furthermore, every layer provides some service to the upper level, taking advantage of those provided from the lower levels.

The real communication is performed at the physical layer, whereas the upper levels set up a virtual communication with their correspondent. A message that must be exchanged between two terminals crosses all the layers (up to the physical layer)

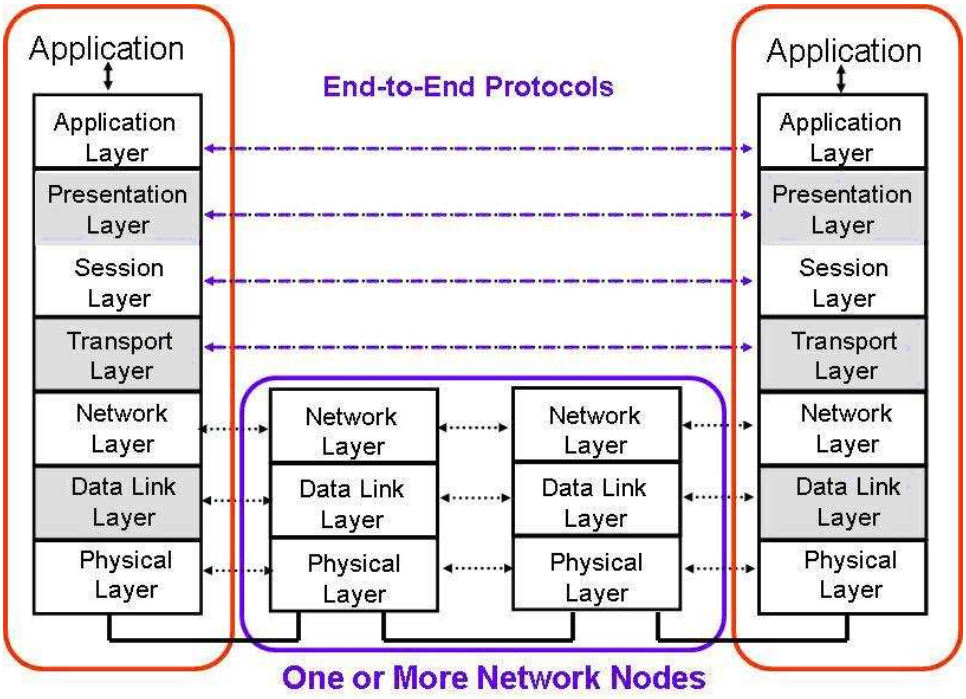


Figure 2.1. OSI reference model

and each layer elaborates the data and adds in the head to the message an header (encapsulation technique). When the message arrives to the terminal receiver, each layer removes the corresponding header and uses it for the own tasks. The Internet architecture presents only four layers, whose correspondences with OSI architecture are shown in Figure 2.2.

The functionalities of each layer can be briefly resumed as follows:

- *Data Link Layer* includes the device drivers and the corresponding Network Interface Card (NIC).Then, it allows the interfacing to the physical media.

- *Network Layer* has in charge both addressing and routing functions. It manage the transfer of the IP packet around the network.

- *Transport Layer* sets up, keeps and releases a connection between the two ends of a communication. Moreover, it has in charge the segmentation and reassembly of the messages.

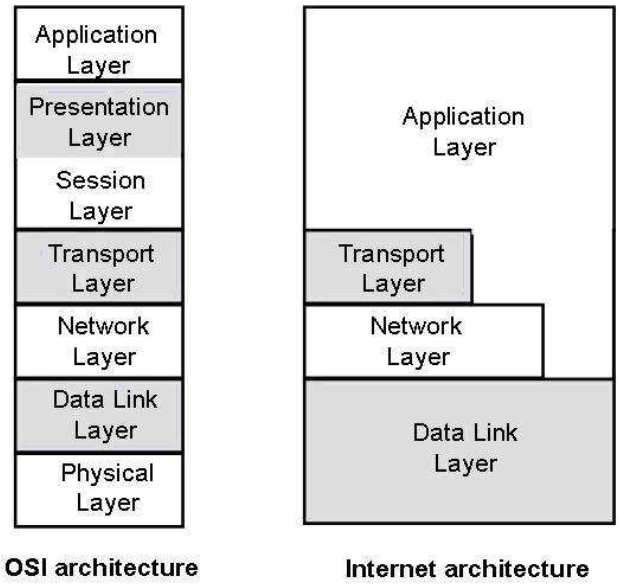


Figure 2.2. Comparison between OSI and Internet architecture

- *Application Layer* provides services that are frequently requested by the application.

## 2.2 TCP services

The transport layer provides a flow of data between two terminals for the application layer above. In the TCP/IP protocol suite, two very different transport protocol are defined: TCP (Transmission Control Protocol) [16] and UDP (User Datagram Protocol) [17].

Even though TCP and UDP use the same network layer (IP), TCP provides a totally different service to the application layer than UDP does.

TCP provides a *connection-oriented*, reliable, byte stream service.

Three functions can be identified in the TCP protocol: flow control, congestion control and error recovery. The flow control scheme allows an efficient exchange of data not exceeding the capacity of the receiver buffer; it is based on the *advertised window* (rwnd) indicating the free space on the receiver buffer [15]. The congestion control scheme governs how TCP avoids and reacts to network congestion. In particular, TCP uses a variable, called *congestion window* (cwnd), to control the

amount of data sent at a certain time. Therefore, the network status is probed by gradually increasing *cwnd* until congestion is experienced and *cwnd* is resized [19]. The minimum between *cwnd* and *rwnd* determines the transmission window. TCP uses cumulative acknowledgements (ACKs) to notify the sender on the last segment correctly received and in order [15]. Then, the timeout expiration at the sender or the receipt of duplicated ACKs indicates the loss of the corresponding segment, thus triggering one of the implemented error recovery schemes.

## 2.3 TCP header

The TCP data are encapsulated into an IP datagram, as shown in Figure 2.3. Then, TCP segment is composed, in turn, of a header and a data payload. Herein, header fields are briefly described, introducing also some basic information on the TCP specific control schemes (i.e. flow control) that are treated in depth in the following.


Figure 2.3. TCP encapsulation over IP

The normal size of the TCP header is 20 bytes, without additional options. Each TCP segment contains the source and destination port number to identify the sending and receiving application (see Figure 2.4). These two values, along with the source and destination IP addresses in the IP header, uniquely identify each connection. The combination of an IP address and a port number is called "socket". The *sequence number* field identifies the first byte of data in the TCP segment. The sequence number is a 32-bit unsigned number then ranging from 0 to $2^{32} - 1$.

When a new connection is being established, the SYN flag is turned on and the *sequence number* field contains the "initial sequence number" (ISN), chosen by this host for this connection. Since every byte that is exchanged is numbered, the *acknowledgement number* refers to the next sequence number that TCP expects to receive. This field is valid only if the ACK is on. The *header length* gives the length of the header in a 32-bit words. This is required because the length of the options field can be variable (TCP is constrained to a 60-bytes header). TCP flow

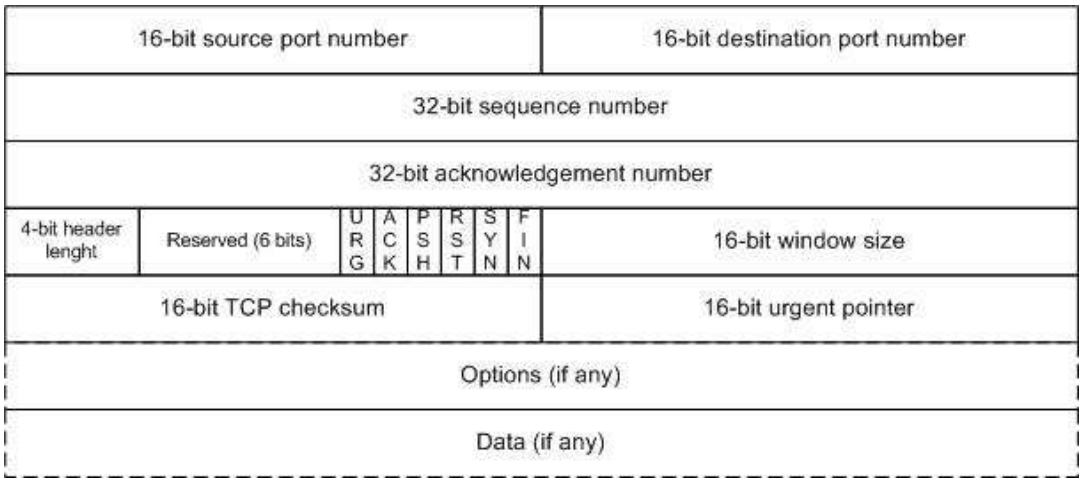


Figure 2.4. TCP header

control is provided by each end host advertising a *window size*. This is the number of bytes, starting with the one specified by the acknowledgment number field, that the receiver is willing to accept. It is a 16-bit field, limiting the window to 65535 bytes. The *checksum* refers to the whole TCP segment: the TCP header and the TCP data. This is a mandatory field that must be calculated and stored by the sender, and then verified by the receiver. Finally, the *urgent pointer* is valid only if the URG flag is set. Such a pointer is a positive offset that must be added to the sequence number field of the segment to yield the sequence number of the last bytes of urgent data. TCP's urgent mode is a way for the sender to transmit emergency data.

The meaning of the six flag bits in the TCP header is shown in Table 2.1.

| | |
|---|---|
| URG | The urgent pointer is valid |
| ACK | The acknowledgement number is valid |
| PSH | The receiver should pass this data to the application as soon as possible |
| RST | Reset the connection |
| SYN | Synchronize sequence numbers to initiate a connection |
| FIN | The sender is finished sending data |

Table 2.1. TCP header bit flags

## 2.4 TCP buffers

A buffer is needed at each end of the TCP connection and such a buffer pair plays an important role in terms of achievable performance. The maximum amount of data in transit in the network is constrained by the bandwidth-delay product, that is, the product of the bottleneck link bandwidth and Round Trip Time (RTT). In order to prevent packet loss due buffer overflow, the receiver needs enough buffer space to store all incoming data. For this purpose, TCP implements a mechanism called "Advertised Receive Window" [15]. This mechanism limits the maximum number of bytes in flight in a given time to the free space in the receiver buffer. In other words, the maximum transmission rate achievable is governed by the following formula:

$$Max\ Tx\ Rate \leq \frac{Advertised\ Receive\ Window}{RTT} \tag{2.1}$$

Therefore, buffer size is fundamental to achieve good performance in case of high latency and broadband links. Studies demonstrates that optimal performance requires a receive buffer size at least twice the Bandwidth-Delay Product (BDP) [18]. Real Operating System (OS) implementations have a default and a maximum size for these buffers that in most cases can be manually changed. Some user-space applications can set TCP buffer size; otherwise the default value is used.

## 2.5 Connection establishment and termination

TCP is a connection-oriented protocol, which uses a mechanism, called *three-way handshake* [15], that envisages of three steps:

1. the "client" sends a SYN segment specifying the port number of the "server" and its Initial Sequence Number (ISN);

2. the "server" responds with its SYN segment containing its ISN and also acknowledges the client's SYN;

3. finally, the "client" acknowledges the SYN segment coming from the "server".

To terminate a connection TCP needs four messages. In fact, since the TCP connection is full-duplex, each end-points must shut down independently by a closing message and the corresponding acknowledgement ("half-close") [15].

## 2.6 Flow control

TCP performs a flow control aiming to provide to the application layer a byte-based, in-order (without errors or duplications) stream of data. TCP general scheme is based on the following steps:

- a sending application writes data into a sending buffer through the socket;
- TCP groups bytes into the TCP packet structure and send them through the network;
- When TCP packets reach the receiver, an ACK is generated and sent back to the sender. Simultaneously, data bytes are accommodated into a receiving buffer.
- A receiving application reads bytes from the sending buffer through the socket.

Such a scheme is depicted in Figure 2.5

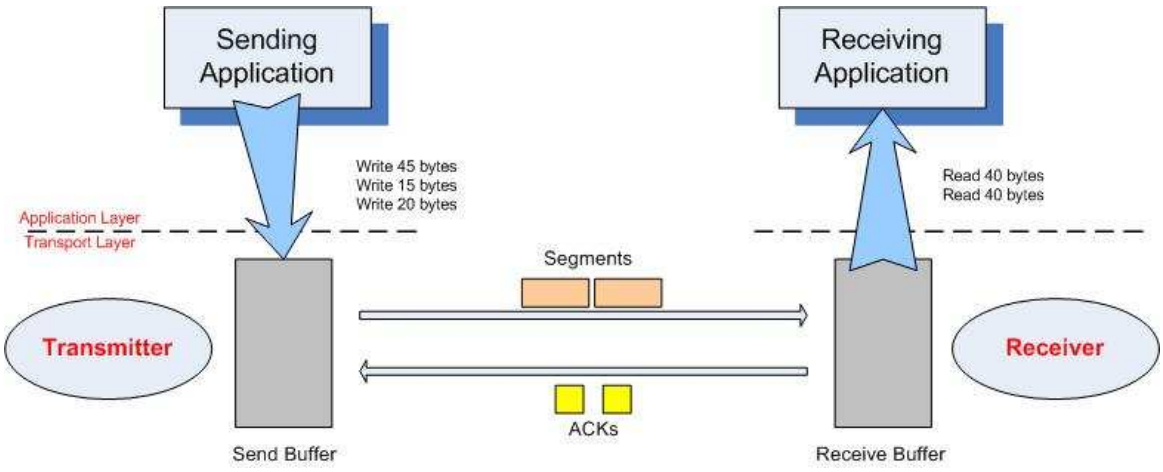


Figure 2.5. TCP flow control scheme

TCP has been conceived to be self-clocking by using ACKs receptions as an indication that a segment has left the network (successfully received) and then there is space to accommodate one more segment (Figure 2.6a). In case ACK is not received within a timeout interval, TCP assumes that packet has been dropped somewhere and retransmits it (Figure 2.6b). This is the basic TCP error control mechanism (Section 2.7.3). TCP manages the start up of clocking and adjusts continuously it by its congestion control scheme (Section 2.7).

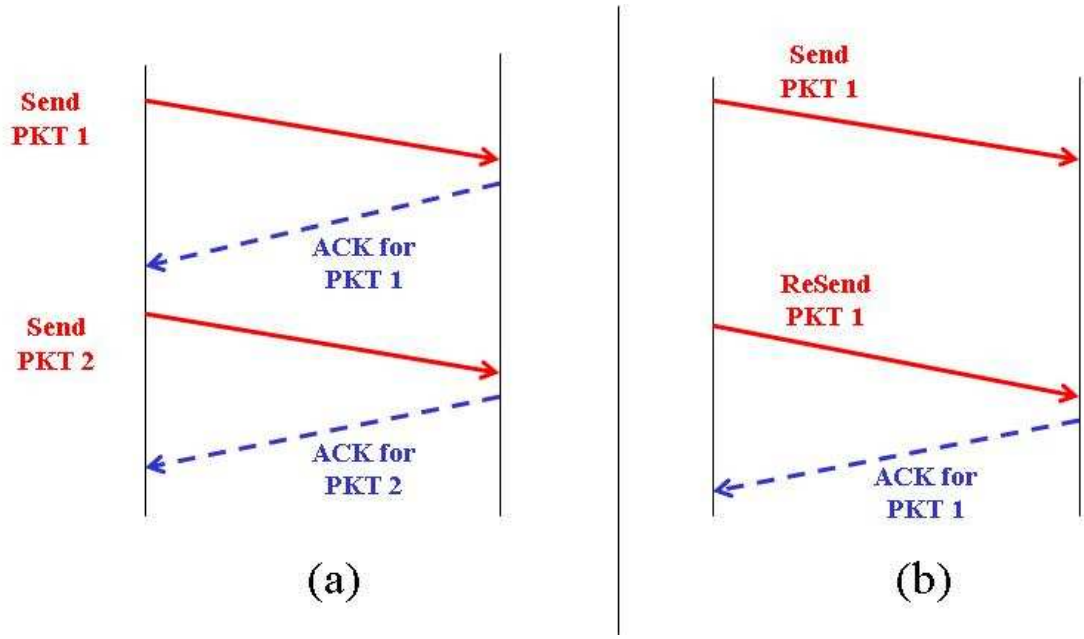


Figure 2.6. ACK for error control

In addition to the ACK-clocking mechanism, TCP flow control uses a *sliding window* mechanism [15] to adapt the transmission rate to the available bandwidth in the channel while not exceeding the capacity of the receiver buffer. Then, TCP flow control manages the shift of such a sliding window over the "segment space", while congestion control (Section 2.7) manages its growth/reduction and the receiver advertises a maximum window size, namely *advertised window*.

In principle, TCP window represents the amount of unacknowledged data, in bytes, that the sender can have in transit towards the receiver. In general, the sliding window moves to the right. In particular, the left edge advances to the right when sent data are acknowledged; as a consequence the right edge moves to the

right as the receiver empties space in the buffer, as shown in Figure 2.7.

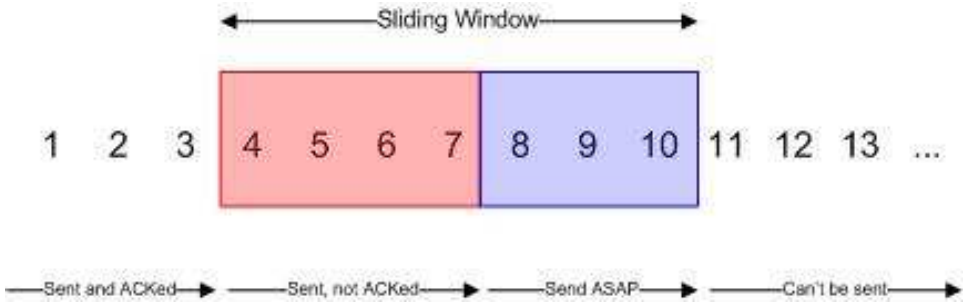


Figure 2.7. TCP sliding window mechanism

## 2.7 Congestion control and error recovery

TCP congestion control manages the *sliding window* increase/decrease. In the last 25 years, TCP congestion control and error recovery schemes have been subject to continuous enhancements coping with the even new communication environments and traffic characteristics. The concepts of *Slow Start* (SS), *Congestion Avoidance* (CA) and *Fast Retransmit-Fast Recovery* (FR-FR) algorithms are presented in [19][20], while the *Retransmission TimeOut* (RTO) mechanism is described in [15][21].

TCP congestion control is mainly based on two state variables:

- the *Congestion Window* (*cwnd*);
- the *Slow Start Threshold* (*ssthresh*).

In more details, *cwnd* limits the amount of unacknowledged data a TCP sender can inject in the network. Some implementations keep its value in unit of bytes, while others in unit of full-sized segment. On the other hand, *ssthresh* represents the switching point between the SS and CA phases.

### 2.7.1 Slow Start (SS)

To prevent congestion problems, TCP sender probes the state of the network by gradually increasing its sliding window until congestion occurs and packet drop as a

consequence. After the three-way handshake is completed and connection is established, TCP sets its cwnd to *Initial Window* (IW) value, less or equal to 2 *Sender Maximum Segment Size* (SMSS), and runs the SS algorithm. SS is characterized by an exponential increase of the *cwnd* as reaction of the ACK receptions (Figure 2.8). For instance, TCP sender initially sends one segment and waits for the corresponding ACK. When it receives the ACK, it sends two segments. In general, during the whole SS, TCP sender increases *cwnd* by SMSS bytes for each ACK received:

$$cwnd_{i+1} = cwnd_i + SMSS. \tag{2.2}$$

Then, as a function of the elapsing time, the *cwnd* growth can be expressed as follows:

$$cwnd(t) = 2^{\frac{t}{RTT}} \tag{2.3}$$

The slow start phase terminates when the *cwnd* exceeds the *ssthresh* or when a packet is lost.

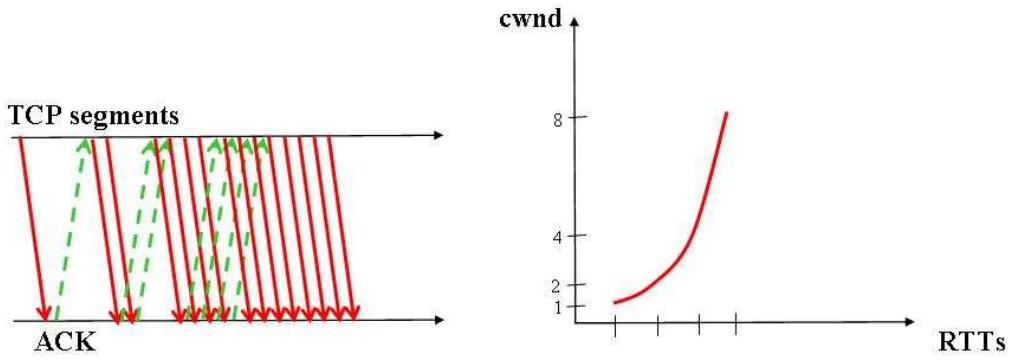


Figure 2.8. Slow Start dynamics

## 2.7.2 Congestion Avoidance (CA)

When *cwnd* gets over *ssthresh* value, the CA algorithm is triggered [20]. During CA, for every incoming non-duplicate ACK, the cwnd in bytes is update according to the following rule:

$$cwnd_{i+1} = cwnd_i + \frac{SMSS}{cwnd_i} \tag{2.4}$$

Accordingly, the *cwnd* growth over time is:

$$cwnd(t) = \frac{t - t_{ssthresh}}{RTT} + ssthresh \tag{2.5}$$

The above formulas can be interpreted as a *cwnd* increase of 1 SMSS per RTT (Figure 2.9).

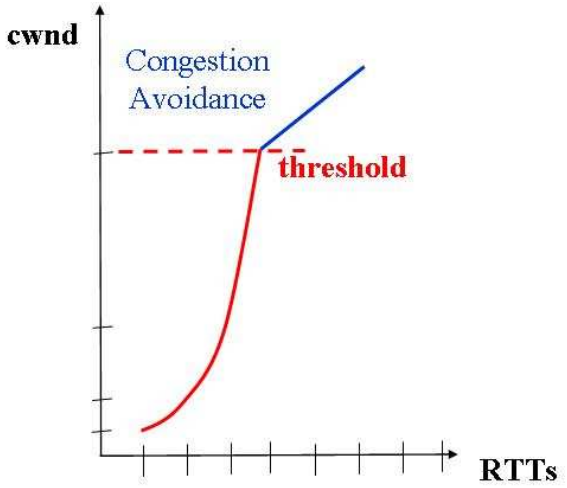


Figure 2.9. Congestion Avoidance dynamics

### 2.7.3 Retransmission TimeOut & Delayed ACKs

TCP uses a retransmission timer to ensure data delivery in the absence of any feedback from the remote data receiver. The duration of this timer is referred to as RTO (retransmission timeout). To compute the current RTO, a TCP sender makes use of two state variables:

- SRTT (smoothed RTT);
- RTTVAR (RTT variance).

Another important role is also played by the timer granularity, indicated in the following as $G$. The rules governing the computation of SRTT, RTTVAR, and RTO are described in [21] and listed in the following:

1. Until an RTT measurement has been made for a segment sent between the sender and receiver, the sender sets RTO to 3 s, though the "backing off" on repeated retransmission still applies.

2. When the first RTT measurement $R$ is made, the TCP sender sets SRTT equal to $R$, RTTVAR to $\frac{R}{2}$ and RTO to $SRTT + max(G,K \cdot RTTVAR)$, where $K = 4$.

3. When the next RTT measurement $R'$ is available, a host sets

$$RTTVAR_{new} = (1 - \beta) \cdot RTTVAR_{old} + \beta \cdot |SRTT_{old} - R'| \tag{2.6}$$

$$SRTT_{new} = (1 - \alpha) \cdot SRTT_{old} + \alpha \cdot R' \tag{2.7}$$

   where the subscripts "new" and "old" indicate the new updated value and the old one, respectively. Moreover, usually $\alpha = \frac{1}{8}$ and $\beta = \frac{1}{4}$.

4. After the computation, a host updates RTO following the relation

$$RTO = SRTT + max(G,K \cdot RTTVAR) \tag{2.8}$$

On the basis of the computation of the RTO value, the following procedure manages the retransmission timer:

1. For each data transmission (or retransmission), if the timer is not running, it starts so that it will expire after RTO seconds.

2. When all outstanding data has been acknowledged, the retransmission timer is turned off.

3. When a non-duplicated ACK is received, the timer is restarted so that it will expire after RTO seconds.

When the retransmission timer expires:

1. The earliest segment that has not been acknowledged by the TCP receiver is retransmitted.

2. The host performs the back-off operation by setting RTO to $RTO \cdot 2$ ("back off the timer"). This RTO value is maintained until either a fresh RTT estimate is available or a new RTO expiration occurs.

Many TCP's acknowledge only every $K^{th}$ segment out of a group of segments arriving within a short time interval; this policy is known generally as *delayed ACK* [23]. The data-sender TCP must measure the effective RTT, including the additional time due to delayed ACK's, or else it will retransmit unnecessarily. Thus, when delayed ACK's are in use, most Berkeley-derived implementations of TCP measure only 1 RTT per window.

### 2.7.4 Fast Retransmit & Fast Recovery

In early TCP implementations (old TCP Tahoe) [22], whenever a loss occurred the sender had to wait for a timeout interval before retransmitting the missing packet. Since the length of this timeout is conservative, the idle time after a loss event grows continuously. Also, after a retransmission timeout the sender wastes bandwidth re-entering the "Slow Start" phase. To mitigate such problems TCP Reno (1990) introduced the "Fast Retransmit" and "Fast Recovery" algorithms [15][19] which allow the TCP sender to retransmit a lost segment before timeout. In fact, since TCP generates duplicate acknowledgements (dupACKs) [15], the Fast Retransmit algorithm considers the arrival of three dupACKs as a clear indication that a segment has been lost and retransmits it immediately. In addition, the Fast Recovery algorithm interprets the dropped packet as an indication of network congestion and reduces the congestion window. In detail, The fast retransmit and fast recovery algorithms are usually implemented together as follows:

- After receiving 3 dupACKs, TCP sender:
  - Retransmits the lost segment;
  - Sets $ssthresh = \frac{currentcwnd}{2}$
  - Sets $cwnd = ssthresh$ , and $ndupacks = 3$;
  - Sets $cwnd = cwnd + ndupacks$;
- If further dupACKs arrive:

- $++ ndupacks$;
- Transmits new segments, if allowed;

- If new ACK arrives:
  - $ndupacks = 0$;
  - Exit fast recovery.

Figure 2.10 shows a typical time evolution of the congestion window for TCP Reno as a function of RTT.

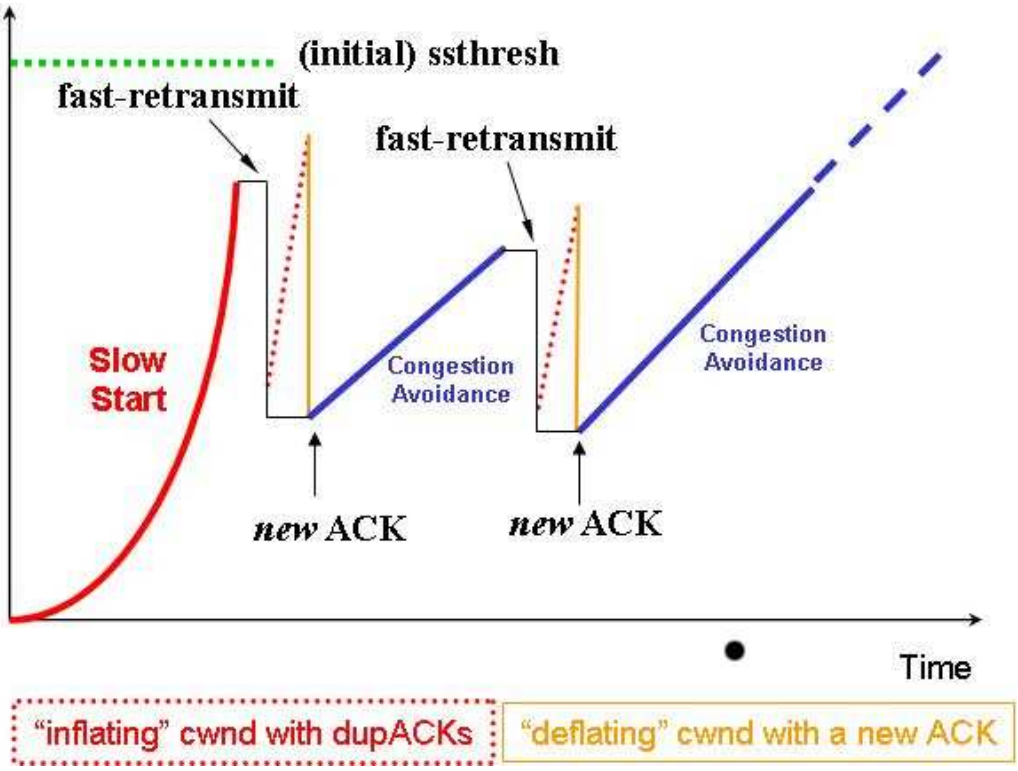


Figure 2.10. TCP congestion window evolution

# Chapter 3

# Analysis on TCP over Satellite

The core of TCP/IP protocol stack has been designed and developed from the mid 1970's coping with issues usually encountered in wired networks [20][24]. Therefore, some of the assumptions of the first TCP design fail in satellite links leading to poor performance [25][26][27][28][29].

This chapter points out the factors impacting TCP performance over satellite links by analyzing TCP dynamics in related environments. Furthermore, all main strategies to counteract the satellite constraints are classified and analyzed.

The overall analysis is enriched by results, coming from both simulation and real measurements, achieved by author.

## 3.1 Taxonomy

A large number of solutions have been proposed in the literature to improve TCP performance over satellite links. Nevertheless, the constant upgrading of satellite standard, the technology development and the definition of new network topologies lead at arising new problems to face, and accordingly new solutions become necessary. In addition, there are many solutions originally proposed for different environments (i.e., Wireless terrestrial networks), that, although partially, improve TCP dynamics over satellite links.

Figure 3.1 proposes a taxonomy, which classifies both problems and solutions for TCP over satellite into groups and sub-groups.

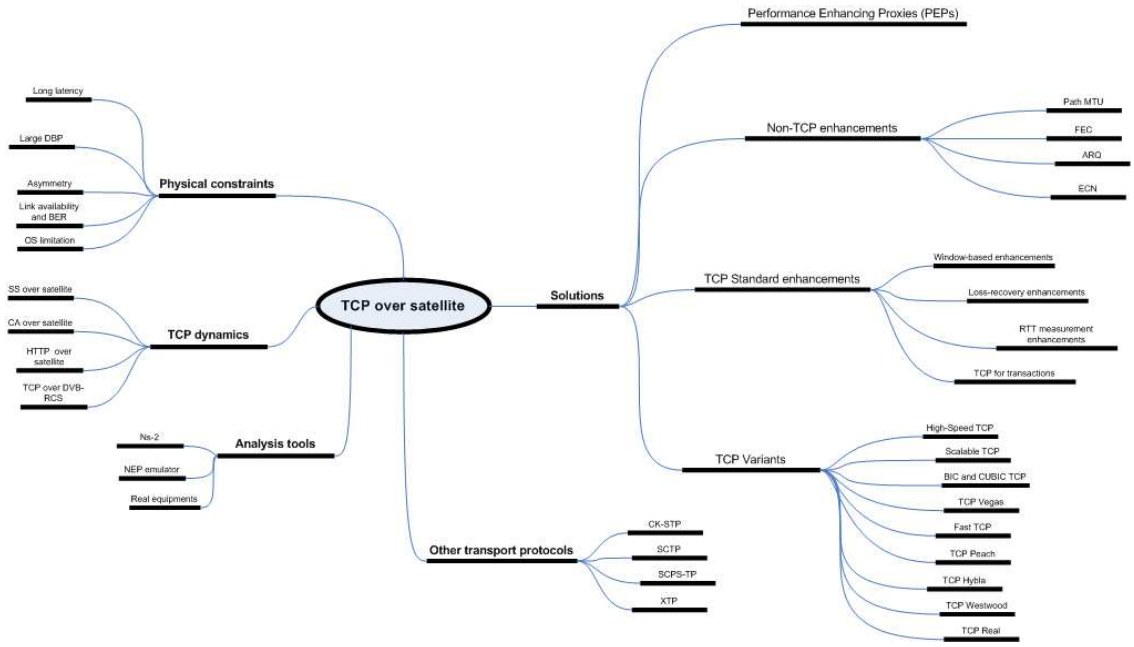


Figure 3.1. TCP over satellite: taxonomy

First, factors impacting TCP performance have been distinguished from the consequential alteration of TCP dynamics when run over a satellite link. The rationale is that this kind of differentiation can result useful for two main matters:

1. Only a sub-set of such factors can be involved in the target satellite scenario. As an example, some satellite link can be designed in order to provide a "Quasi Error Free" ($BER \simeq 0$). Then, corruption losses are not experienced by TCP. More in general, the weight of each factor in the achievement of TCP performance is strongly dependent on the specific scenario.

2. Both performance and the impact of solutions depend on the application. As an example, Web traffic is particularly affected by start-up problems, while large FTP downloads are limited by steady-state problems.

The solutions can be classified as follows:

- *Non-TCP enhancements.* They aim to improve TCP performance by providing enhanced lower layers. In other words, TCP can benefit from mechanisms running at different protocol layers.

- *TCP standard enhancements.* Extensions compliant to the standard TCP protocol specifications [16].
- *TCP variants.* Modifications of the standard flow, congestion and error recovery schemes.
- *Performance Enhancing Proxies* (PEPs). Modifications of the TCP/IP semantics. PEPs can envisage also architectural modifications (i.e. addition of sofware/hardware agents along the end-to-end path).

For sake of completeness, also a brief survey of transport protocols, used in commercial product to replace TCP, is provided.

Finally, an important role in the TCP performance analysis is based on the definition of a common metric and the selection of suitable analysis tools: simulation software, emulation platform, real equipments.

## 3.2 Factors limiting TCP performance

The main characteristics of an end-to-end path, including a satellite link, impacting TCP performance are:

- large latency;
- High Bandwidth-Delay Product (for wideband multimedia services);
- Exploitation of asymmetric connections;
- Losses due to corruption errors.

In addition to link conditions, some default options of real operating systems (i.e., Linux, Windows, BSD/OS, etc.) limit TCP performance [18].

### 3.2.1 Latency

Three basic factors contribute to latency: propagation delay, processing delay and queuing delay. For satellite scenarios, the contribution of propagation delay often is predominant, unless the used access scheme implies huge processing delay, as for

DVB-RCS DAMA scheme. A network segment using a GEO satellite introduces a propagation delay in the range 240-300 ms end-to-end that must be doubled if RTT is considered. In addition, satellite systems adopting DAMA schemes at the MAC layer, introduce a further delay contribution, named "access delay", which is defined as the time elapsed between the arrival of a segment in the MAC buffer and its actual transmission. As a consequence, a TCP sender may take a long time to receive ACKs. For this reason, the growth of the TCP transmission window, based on the reception of the acknowledgement from the receiver, proceeds very slowly.

### 3.2.2 Large bandwidth-delay product

The bandwidth-delay product (BDP) defines the amount of data the sender must transmit at any instant to fully utilize the available bandwidth. The BDP is the product of the bottleneck link bandwidth and Round Trip Time (RTT):

$$BDP = bottleneck\ BW \left(\frac{bit}{s}\right) \cdot RTT(s) \tag{3.1}$$

When the BDP is huge, TCP needs to increase considerably the transmission window in order to keep a large number of packets "in flight" (packets sent but not acknowledged). This implies a long time spent by TCP in probing the available capacity, and consequentially file transfers can be entirely performed without ever reaching the optimal TCP window (equal to BDP). This leads to a non optimal network resource utilization.

Furthermore, keeping a large number of packets "in flight", the risk of multiple losses in the same window increases. During the recovery of massive losses, round-trip time (RTT) measurements and acknowledgment clocking may be lost triggering consecutive timeouts.

Finally, working with large window sizes, traffic becomes extremely bursty (i.e. a single acknowledgment can acknowledge a large number of packets), opening up the window in a large burst. In addition, sometimes the sender CPU is busy for a long period to serve interrupts of incoming packets, allowing outgoing packets to accumulate at device output queue, to be transmitted in a large burst when the CPU becomes available. Routers or other intermediate elements can be easily overburdened, leading to multiple losses.

### 3.2.3 Connection Asymmetry

The asymmetry concept can e referred to different bandwidths, but also latency, media-access and error-rates in forward and return links. The throughput may depend on the characteristics of both links (forward and return) and asymmetry may impact TCP performance [33]. TCP assumes that congestion can occur only in the data direction (forward link), and RTT increases are always interpreted by the TCP sender as a signal of congestion in the forward link, and consequentially the transmission rate is reduced [20]. Unfortunately, this happens even if congestion affects the ACK flow, leading to an unnecessary rate reduction in the data transmission.

For example, in DVB IP one-way systems the user terminal receives incoming traffic via the satellite channel and sends all outgoing traffic over a terrestrial link. The whole link is asymmetric both in terms of capacity and latency. Another common case of asymmetry occurs when the satellite handles both directions, but the capacity in the return link can be smaller than the capacity in the forward link (e.g., a DVB-RCS system).

### 3.2.4 Link availability & BER

Wireless links can be subject to lower link availability than wired networks [34][34][36] due to random variations of atmosphere conditions, especially at high frequencies. In addition, errors often occur in bursts and, in case of mobile services, multipath and shadowing impact link dimensioning.

BER (Bit Error Rate) depends on service requirements and impacts physical link parameters. When the link availability is low, the long term average BER decreases. Unfortunately, since there is no feedback information about the reason a packet is lost in the network (congestion or corruption), TCP interprets any packet loss as a notification of network congestion and reduces its transmission window in order to alleviate congestion.

### 3.2.5 Real Operating System limitations

Optimum performance is obtained when the data pipe between sender and receiver is always kept full. On the other hand, the receiver needs enough free buffer space to

store all incoming data waiting to be processed. For this purpose TCP implements the already mentioned "Advertised window" field in ACK packets. Unfortunately, the "Advertised Receive Window" field was initially designed as a 16-bit field so that $2^{16}$=64 KBytes is also the maximum receiving buffer size. 64 KBytes are considered as a reference maximum value, even if new options for window scaling [37] are commonly being adopted to extend this value to GBytes. Anyway, as pointed out in Table 3.1, not always the limit of 64 KBytes is adopted (a common value is 32 KBytes) and the operations to perform to tune the OS are usually for very skilled users: for instance in Windows XP to use 64 KBytes or to go further using the windows scaling option, it is necessary to add new keys in the registry manually.

| **OS** | **Max. size** | **Def. size** | **App. Def. size** |
|---|---|---|---|
| FreeBSD | Avail. RAM | 256 KB | 32KB |
| Linux 2.6.7 | | Auto tuning up to 4MB. | |
| Linux - older | Avail. RAM | 64kB | 32KB |
| MacOS X | Avail. RAM | 256KB | 32KB |
| Sun Solaris 10 | Avail. RAM | 1MB | 48KB |
| Windows 98/NT4.0 | 64KB | n.a. | 8KB |
| Windows 2000 | Up to 1GB | n.a. | 8KB |
| Windows XP/2003 | Up to 1GB | n.a. | 16KB |
| Windows Vista | | Novel tuning system | |

Table 3.1. TCP receive buffer size in common OS

Such a limitation can strongly affect the performance in high-speed networks, where the TCP transmission window could potentially grow up to several MBytes. Figure 3.2 shows the maximum transmission rate achievable as function of the round-trip time when the buffer is set to 64KB. Considering a reference available capacity of 2 Mbit/s, links with large RTT ($>$ 250 milliseconds) could under use it.

## 3.3 TCP enhancements

In recent years, several modifications of standard procedures and/or on tuning TCP parameters have been proposed for improving TCP performance over both wired and

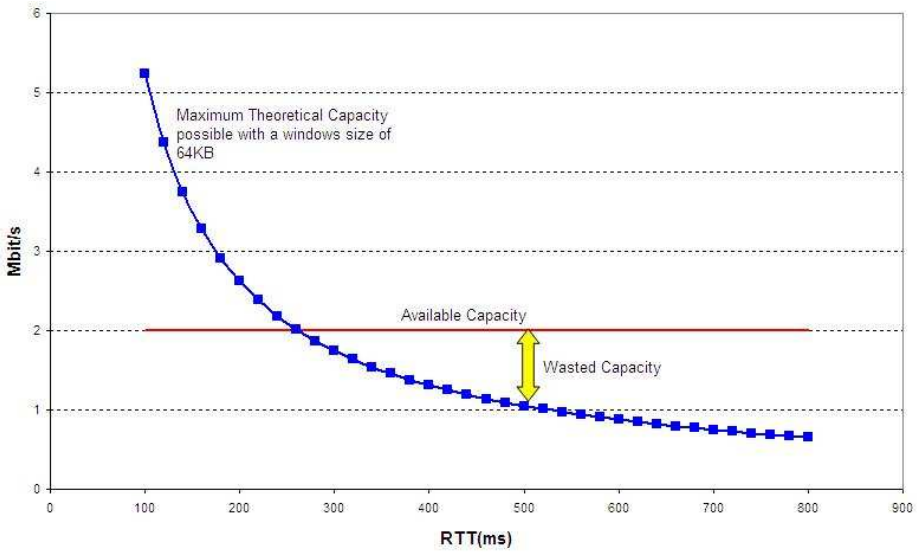


Figure 3.2. Maximum transmission rate vs. RTT

wireless links, including satellite connections. In the following, the TCP enhancements addressing, directly or indirectly, satellite semiretirement are summarized [37][47][48][49][50] [51][52][53].

### 3.3.1 Window-based enhancements

This section collects all the enhancements aiming to improve the TCP window management. In particular, some techniques/options address the use of large windows (in case of large pipes), while others deal with the improvement of the ACK-clock timing.

#### 3.3.1.1 Window scaling

To fully exploit large pipe available in high-speed networks, a TCP option, called "Window Scaling" [37], has been defined to expands the definition of the TCP window to 32 bits and then uses a scale factor to carry this 32-bit value in the 16-bit window field in the TCP header.

#### 3.3.1.2 Large initial window

To improve the TCP start-up behavior, [51] specifies an increase in the permitted upper bound for the TCP initial window from one segment up to four segments. In particular, the upper bound is given by:

$$min\,(4 \cdot MSS, max(2 \cdot MSS,\ 4380\ bytes)) \tag{3.2}$$

where MSS is the maximum segment size. In this way, file with sizes under approximately four thousand bytes can be transmitted in only one RTT.

#### 3.3.1.3 Byte counting

A possible approach to speed up an initial connection is known as "byte counting" [53]. Basically, byte counting requires a modified sender that increases the congestion window considering the amount of acknowledged data (in bytes) rather than the number of acknowledgements received. This algorithm improves performance by mitigating the impact of delayed ACKs (Section 2.7.3) on the growth of *cwnd* . At the same time, the algorithm provides cwnd growth in direct relation to the probed capacity of a network path.

#### 3.3.1.4 Timestamp option

Accurate RTT estimates are necessary to adapt the transmission rate to the changing traffic conditions and to avoid instability events, known as "congestion collapses" [54]. Since TCP keeps only 1 timer, RTT measurements are based upon a sample per sent window. This may results in an inaccurate RTT estimate in case of large windows. A solution to this problem is based on the use of the "Timestamp option" [37]. Basically, the sender places a timestamp (representing the transmission time of the segment) in each data segment header, and the receiver reflects this value back in the corresponding ACK header. Then, an RTT sample is computed for every received ACK.

#### 3.3.1.5 Protect Against Wrapped Sequence (PAWS) numbers

TCP "Protect Against Wrapped Sequence" (PAWS) option [37] works by including the "Timestamp option" in all TCP headers in order to help the validation of the packet sequence numbers. TCP senders use an internal timer to compare the value of the timestamp associated to incoming packets against the last valid timestamp recorded. If the segment timestamp is larger than the value of the last valid timestamp and the sequence number is less than the acknowledged one, TCP sender updates its internal timer with the timestamp value and passes the packet on for further processing.To opposite, if TCP sender detects a timestamp smaller than the last recoreded one while the sequence number is greater than last acknowledged one, it rejects the packet.

### 3.3.2 Loss recovery enhancements

This section provides a look at the impact of loss recovery enhancements on TCP performance.

#### 3.3.2.1 NewReno option

The FR-FR mechanism explained in section 2.7.4 does not provide good performance when multiple segments are lost within the same window. In fact, with such a scheme, after the recovery of the first lost segment, *cwnd* is halved and the probability to receive enough dupacks (typically 3) to detect the next loss decreases. Consequentially, the probability of a timeout expiration grows. To this purpose, tests demonstrate that:

- with two losses in a window, Reno will occasionally timeout;
- with three losses in a window, Reno will usually timeout;
- with four losses in a window, Reno is guaranteed to timeout.

To mitigate this problem, TCP NewReno introduces the possibility to send, besides to cumulative ACKs, some partial ACKs [55]. In particular, partial ACKs do not take out of Fast Recovery, while, when received during Fast Recovery, are

treated as an indication that the segment immediately following the acknowledged segment in sequence space has been lost, and should be retransmitted. Thus, when multiple segments are lost from a single window of data, NewReno can help to recover them without experiencing timeout. As a drawback, the recovery of each loss needs an RTT.

#### 3.3.2.2 SACK option

The "Selective Acknowledgement" (SACK) TCP extension [50] allows TCP to report and recover a large number of missing segment in one only RTT. Such an extension uses two different TCP options: the first is an enabling option, "Sack Permitted", which indicates that the SACK extension can be used once the connection is established; the second option is sent by the receiver to inform the sender of non-contiguous blocks of data that have been received.

### 3.3.3 Connection start-up enhancements

Internet applications tend to be transaction-oriented, rather than use a virtual circuit style of communication. Transaction-oriented applications particularly suffer the overhead of opening and closing TCP connections. In fact, TCP needs at least 7 segments for a whole communication including connection setup, data transmission and close (data can be piggy-backed with connection setup and/or close).

Some of the enhancements presented above (i.e. large initial window, byte counting, etc.) help in speed up the slow start phase. But, in case of very short transfers (i.e. web traffic), such improvements are completely hidden by the inefficiency of the TCP connection management.

In such a frame, T/TCP [49][57] defines an extension for TCP to improve performance in case of short transactions. Finally, the minimum number of massages to exchange is reduced from 7 to 3, thanks to a mechanism called "TCP Accelerated Open" (TAO).

## 3.4 TCP Variants

Plenty of new TCP-based protocols are currently available. Some of them introduce "modified" congestion control scheme, while others replace congestion control with a rate control scheme. Among the many proposals in literature, this section gives attention to those most suited to cope with satellite characteristics. Of course, all the presented TCP variants address only a sub-set of the satellite constraints (often changing scenario by scenario) above mentioned, nevertheless they can help in identifying a set of strategy to fix inefficiency of the standard TCP over satellite links.

### 3.4.1 High-Speed TCP

High-Speed TCP [58][59] aims to improve the loss recovery time of the standard TCP. Its behavior is equivalent to TCP standard TCP for small congestion windows (below a fixed threshold) while it is more responsive on big windows. It accomplishes this with the use of a modified "Additional Increase Multiple Decease" (AIMD) behavior during congestion avoidance, dependent to the current window size. In fact, the additional and multiplying factors are proportional to the window size in this way:

$$\begin{cases} cwnd = cwnd + \frac{a(cwnd)}{cwnd} & \forall\ ACK\ received \\ cwnd = (1 - b(cwnd)) \cdot cwnd & if\ 3\ dupACKs \end{cases} \tag{3.3}$$

with $a(cwnd)$ proportional to $cwnd$ and $b(cwnd)$ inversely proportional to $cwnd$. Then, a faster $cwnd$ is allowed in CA while after a loss $cwnd$ is less than halved.

### 3.4.2 Scalable TCP

As the High-Speed TCP, Scalable TCP modifies the standard TCP only in the management of large $cwnd$ during the congestion avoidance. The concept and the implemented algorithm of this version are so close to the High-speed TCP, that they are defined in the same RFC [58].

### 3.4.3 BIC & CUBIC TCP

BIC (Binary Increase Congestion control) is implemented in the standard Linux kernel since the 2.6.7 version, and it is also used as default TCP for Linux machines. The protocol combines two schemes called "additive increase" and "binary search increase". In the former, congestion control is considered as a searching problem in which the system gives `yes/no` feedback through packet loss as to whether the current sending rate is larger than the network capacity. The boundary points for such a searching are named "minimum window size" ($W_{min}$) and "maximum window size" ($W_{max}$). In particular, $W_{max}$ is *cwnd* value just before the last *fast recovery* and $W_{min}$ is the *cwnd* just after the *fast recovery*. The algorithm repeatedly computes the midpoint between $W_{min}$ and $W_{max}$, set the next *cwnd* size (1 RTT later) to the midpoint, and checks for feedback in the form of packet losses. Based on this feedback, the midpoint is taken as the new $W_{max}$ if a loss is detected, and as the new $W_{min}$ otherwise. This process is repeated until the difference between $W_{max}$ and $W_{min}$ falls below a preset threshold, called "minimum increment" ($S_{min}$). This technique allows bandwidth probing to be more aggressive initially, and becomes less aggressive as the current *cwnd* gets closer to $W_{max}$. Then, the increase function is logarithmic. In order to ensure faster convergence, the "binary search increase" is combined with an "additive increase" strategy. When the gap between the midpoint and the current *cwnd* is larger than a preset "maximum increment" ($S_{max}$)), *cwnd* is increased by $S_{max}$. A "Slow Start" strategy is adopted from the "old" $W_{max}$ to $W_{max} + S_{max}$.

CUBIC is an enhanced version of the TCP BIC less aggressive at startup avoiding the additive increase. CUBIC does not perform the binary search while enforces a single algorithm for window adjustment (namely a cubic function) that is:

$$W_{cubic} = C \cdot (t - K)^3 + W_{max} \tag{3.4}$$

where $C$ is a scaling factor, $t$ is the elapsed time since the last window reduction, $W_{max}$ is the window size just before the last window reduction and $K = \sqrt[3]{\frac{W_{max} \cdot \beta}{C}}$ where $\beta$ is a multiplicative decrease factor after a packet loss event.

### 3.4.4 TCP Vegas

TCP Vegas [60] proposes a new sender-based congestion control mechanism aiming to prevent congestion events by monitoring the difference between the expected rate and the actual rate. Specifically, TCP Vegas is based on three mechanisms:

1. *A new retransmission mechanism.* A TCP Vegas sender retransmits more aggressively. In particular, it measures the RTT for every sent segment. Such a RTT measurement is used to set a RTO timer for each segment. Vegas checks whether RTO has expired and then decides for a retransmission in the following situations:

   **i.** when a duplicate ACK is received;

   **ii.** when the first or the second non-duplicate ACK after a retransmission is received. Basically, this mechanism aims at reducing the time to detect multiple packet losses.

2. *A new Congestion Avoidance mechanism.* To predict the congestion, TCP Vegas considers not only the dropped packets, but also the variations of the estimated RTT. To implement this congestion control, it defines the "BaseRTT" as the RTT without congestion (minimum of all measured RTT), and then the expected throughput is calculated as the ratio between the transmission window size and BaseRTT. On the other hand, TCP Vegas calculates the current sending rate by dividing the amount of outstanding bytes by the RTT. Finally, it compares the current and the expected throughput and adjusts the congestion window accordingly:

$$Diff = Expected - Current. \tag{3.5}$$

   In fact, by defining two threshold values, $\alpha$ and $\beta$, TCP Vegas implements the following algorithm:

$$\begin{cases} If(Diff < \alpha) & linear\ cwnd\ increase\ in\ next\ RTT \\ If(Diff > \beta) & decrease\ cwnd\ in\ next\ RTT \\ If(\alpha < Diff < \beta) & cwnd\ unchanged\ in\ next\ RTT \end{cases} \tag{3.6}$$

3. *A modified Slow Start mechanism.* If the standard Slow Start mechanism is not limited by buffer dimensions at the end hosts, the congestion window size will double every RTT (if there are no losses), until an overrun of connection capacity. Therefore, the losses may be on the order of half the current congestion window. TCP Vegas prevents congestion events during the Slow Start allowing an exponential window growth only every other RTT. During this time, the congestion window is fixed. Thus, a valid comparison of the expected and actual rates can be performed.

### 3.4.5 Fast TCP

Fast TCP [61] is a sender-side only modification to the Vegas TCP congestion avoidance algorithm. The congestion control mechanism of FAST TCP can be divided into four components. These four components are functionally independent so that they can be designed separately and upgraded asynchronously. The *data control* component determines which packets to transmit, *window control* determines how many packets to transmit, and *burstiness control* determines when to transmit these packets. These decisions are made based on information provided by the *estimation* component. Window control regulates packet transmission at the RTT timescale, while burstiness control works at a smaller timescale. Under normal network conditions, the "window control" algorithm periodically updates the congestion window w based on the average RTT according to:

$$W = min\left\{2 \cdot W,\ (1-\gamma) + \gamma \cdot \left(\frac{baseRTT}{RTT} \cdot W + \alpha\right)\right\} \tag{3.7}$$

where $\gamma \in (0;1]$, RTT is the current average round-trip time, baseRTT is the minimum RTT observed so far, and $\alpha$ is a protocol parameter that controls fairness and the number of packets each flow buffered in the network. It is proved in [61] that, in the absence of delay, this algorithm is globally stable and converges exponentially to the unique equilibrium point where every bottleneck link is fully utilized and the rate allocation is proportionally fair.

### 3.4.6 TCP Peach

TCP-Peach is composed of two new algorithms, namely Sudden Start and Rapid Recovery, as well as the two traditional TCP algorithms, Congestion Avoidance and Fast Retransmit [62]. Sudden Start and Rapid Recovery are designed to replace respectively the Slow Start and Fast Recovery algorithms of standard TCP. The new algorithms are based on the novel concept of using dummy segments to probe the availability of network resources without carrying any new information to the sender. Dummy segments are low-priority segments generated by the sender as a copy of the last transmitted data segment. If a router on the connection path is congested, it discards the IP packets carrying dummy segments first. Consequently, the transmission of dummy segments does not cause a throughput decrease of actual data segments. If the routers are not congested, then the dummy segments can reach the receiver. The sender interprets the ACKs related to dummy segments as evidence that there are unused resources in the network, and increases its transmission rate accordingly. By exploiting dummy segments, Sudden Start provides a fast opening of the congestion window at the beginning of the connections, irrespective of the actual RTT, while Rapid Recovery is used to react to losses [62]. The main requirement of TCP Peach is that the network routers implement some form of DiffServ policy, to differentiate between high and low priority traffic.

### 3.4.7 TCP Hybla

TCP Hybla [63][64] has been conceived with the primary aim of counteracting the performance deterioration originated by the long RTTs typical of satellite connections. It consists of a set of procedures which includes an enhancement of the standard congestion control laws for both the Slow Start and the Congestion Avoidance phases, the mandatory adoption of the SACK policy, the adoption of Hoe's channel bandwidth estimation, the use of timestamps and the implementation of packet spacing techniques. The modification of the standard congestion control rules is dictated by the TCP Hybla ideal aim of obtaining for long RTT connections the same instantaneous segment transmission rate of a comparatively fast reference TCP connection (e.g. a wired one). To this scope, TCP Hybla introduces a new parameter: the normalized round trip time, defined as the ratio between the actual

RTT and the round trip time of the reference connection denoted by $RTT_0$:

$$\rho = \frac{RTT}{RTT_0} \tag{3.8}$$

Then, the standard congestion control laws are replaced by the following, which represent the TCP Hybla congestion control rules when receiving an ACK:

$$W_{i+1} = \begin{cases} W_i + 2^\rho - 1 & SS \\ W_i + \frac{\rho^2}{W_i} & CA \end{cases} \tag{3.9}$$

### 3.4.8 TCP Westwood

TCP Westwood [65][66] was introduced for the purpose of limiting the consequences of the losses introduced by a wireless channel, which are always erroneously ascribed to congestion by the TCP protocol. To this end, TCP Westwood introduces a modification of the Fast Recovery algorithm called "Faster Recovery". Instead of halving the *cwnd* after three duplicate ACKs, and fixing the *ssthresh* to this value, TCP Westwood sets the *ssthresh* as a function of the actual available bandwidth. In this way, channel losses do not cause the dramatic slow-down of the transmission rate as in the standard TCP. The bandwidth is estimated by measuring and averaging the rate of returning ACKs. Then, after a loss detection, TCP Westwood sets the *ssthresh* and *cwnd* as follows:

$$ssthresh = \widehat{b} \cdot RTT_{min} \tag{3.10}$$

$$cwnd = \begin{cases} ssthresh & if\ cwnd > ssthresh \\ 1 & after\ a\ timeout \end{cases} \tag{3.11}$$

where $\widehat{b}$ is the estimated bandwidth.

### 3.4.9 TCP Real

TCP Real [69] aims to abolish three shortfalls of TCP for reliable multimedia applications over heterogeneous networks:

1. ineffective bandwidth utilization;

2. unnecessary congestion-oriented responses to wireless link errors;

3. wasteful window adjustments over asymmetric, low-bandwidth re verse path.

The basic idea of TCP Real is to incorporate into TCP the concept of "wave" from the Wave and Wait Protocol [70] without changing the semantic of TCP and without violating the Additive-Increase/Multiplicative-Decrease (AIMD) based congestion control. The concept of "wave" allows receiver to efficiently estimate network congestion through the knowledge of the sender transmission pattern. Since TCP sender sends packets within the *cwnd* every RTT, from receiver point of view, the sender sends packets in "waves". In addition, TCP Real leaves to the receiver the task of controlling *cwnd* size.

Then, receiver measures the data receiving rate and adjust the wave level accordingly. Furthermore, receiver notifies the sender about the new wave level, using an optional field in the ACKs. When sender receives the ACKs, it extracts such a value and set *cwnd* to the new advertised wave level.

## 3.5 Performance Enhancing Proxies (PEPs)

The improvement of TCP performance has been also addressed by a class of approaches involving additional or enhanced network infrastructures. The basic idea is that intermediaries perform processing of behalf of TCP endpoints to the greater benefit of performance. In this case, performance optimization is achieved at the cost of increased hardware complexity and (at least in some cases) also of severe semantics violations. These solutions are generally referred to as Performance Enhancing Proxies (PEPs) [71]. Hereafter, two of the most promising PEP architectures for satellite environments are briefly presented: namely spoofing and splitting.

### 3.5.1 Spoofing

TCP spoofing [72] is a viable solution for mitigating the performance penalization that stems from long RTTs. It consists in accelerating the growth of the TCP transmission window by masking the long delay experienced on the satellite link (Figure

3.3). More specifically, the satellite gateway, transparently to the end hosts, is responsible for generating automatically fake ACKs in correspondence to the incoming TCP segments. In this way, the latency needed for the hosts to transmit new segments is the sole delay experienced by the terrestrial link, excluding the satellite portion. The main drawback of this approach is that the end-to-end semantics is no longer respected, creating problems and inefficiencies for the applications, which on the contrary rely upon this characteristic. Moreover, the adoption of this scheme implies that the "true" ACKs generated by the destination host have to be intercepted by the satellite gateway and then suppressed in order to avoid spurious duplicate ACKs arriving at the source side; in the case of lost segments, the gateway will be also be responsible for retransmitting the missed segments. For this task, it is also necessary that a symmetric routing be performed; in other words, the TCP data segments and the related ACKs have to pass through the same paths. If this is not the case, the satellite gateway, responsible for performing spoofing operations, will not be able to effectively manage the TCP connections on the satellite link.

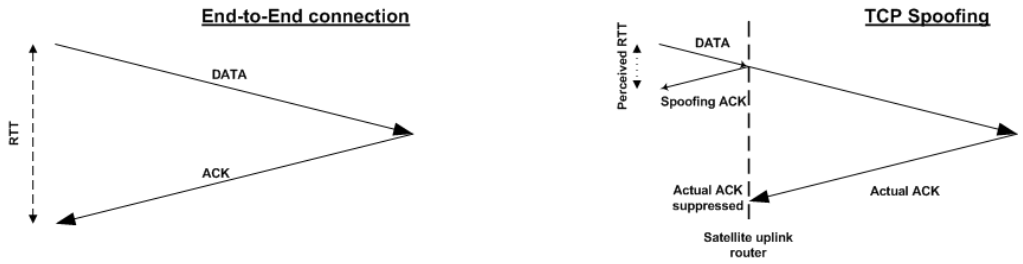


Figure 3.3. TCP Spoofing application over satellite

### 3.5.2 Splitting

The rationale behind the splitting approach is to separate the satellite portion from the rest of the network in order to have dedicated and distinct connections (Figure 3.4).

In practice, this approach can be realized by considering splitting functionalities acting on the satellite gateways. In this way, the TCP connection established between the hosts is split into three separate connections, of which two are over the paths linking the terminal hosts with the satellite gateways; and the third between

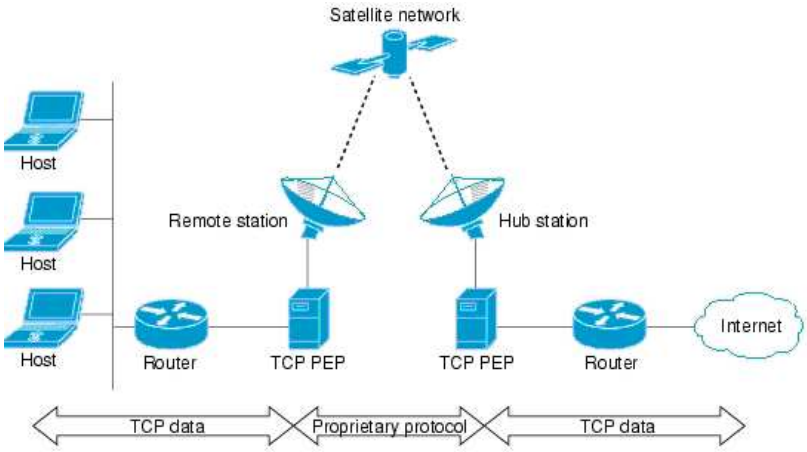


Figure 3.4. TCP Splitting architecture

the gateways over the satellite portion. From the network point of view, a new architecture is defined and its added value consists in the possibility of hiding the satellite link to the end users, which continue to use the TCP stack for connecting to satellite gateways, while between them it is possible to take advantage of a specifically designed transport protocol [73]. Examples of TCP-based protocols optimized for satellite environment (even though designed for different aims) are:

- the "Complete Knowledge-Satellite Transport Protocol" (CK-STP) [73];
- the "Stream Control Transmission Protocol" (SCTP) [74];
- the "Space Communications Protocol Standards-Transport Protocol" (SCPS-TP);
- the "Xpress Transport Protocol" (XTP) [79].

## 3.6 Non-TCP Enhancements

"Non-TCP enhancements" are here meant as all the techniques, implemented out of the transport layer frame, directly or indirectly enhancing TCP performance. Of course, their effect can be more or less significant to TCP performance depending on the target scenario. In the following, four techniques particularly suited for

either satellite or hybrid network (including satellite segment) are briefly described: Path MTU discovery, Forward Error Correction (FEC), Automatic Repeat Requests (ARQ), Explicit Congestion Notification (ECN).

The Path MTU discovery option [42] allows a TCP connection to explore a network path for the largest allowable Message Transfer Unit (MTU). For satellite-based TCP connections, the use of the largest segments possible has two main benefits:

1. it reduces the overhead;
2. it allows the transmission of the maximum amount of data possible under the constraints of the TCP *cwnd*.

FEC schemes [43] can be employed to improve the Bit Error Rate (BER) and them to reduce the losses due to corruption errors misinterpreted by TCP as congestion signals. The cost for a lower BER is a reduced efficiency in the use of the available spectrum [44] due to the higher redundancy.

In case of links with high error rates (regardless the use of FEC schemes) the use of link layer retransmission protocols, called "automatic repeat request" (ARQ), can improve performance. In detail, an ARQ protocol, as defined in [45], carries out the retransmission of the lost packets via ARQ algorithms at the link layer. In this way, the IP layer sees an error-free sequence of packet and TCP can work efficiently. However, a long recovery phase could lead to coarse timeout at the TCP layer.

Finally, a last technique, named Explicit Congestion Notification (ECN) [78], aims to prevent congestion events through explicit notification. In practise, intermediate router use the ECN bit once the queue occupancy grows above a given threshold. Then, the sender once aware of an imminent congestion could perform some defensive actions.

## 3.7 Analysis tools

This section presents the basic characteristics of the tools used by author as a support to analyze TCP performance over satellite and gives details on the metric adopted to compare outcomes.

### 3.7.1 A simulation platform

A simulation platform, modeling the DVB-RCS access schemes, has been built on the satellite network extensions of the network simulator Ns-2 [30]. In particular, the author has used Ns-2 satellite extensions to model a geostationary "bent-pipe" satellite node connected to Satellite Terminals (STs) and a satellite gateway (GW) through asymmetric links, a population of STs and a satellite network interface stack for each ST. In addition, modifications on the internal *C++* code and the creation of *tcl* procedures have allowed to model a DVB-RCS-like MAC layer, supporting all relevant DAMA disciplines (Section 1.5.3.2).

The structure of the implemented model is represented in the figure 3.5.

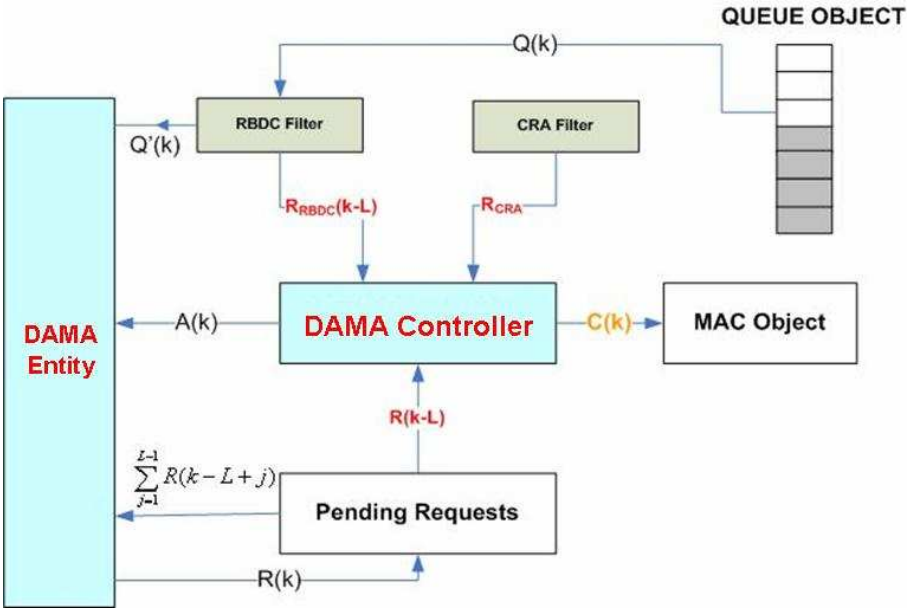


Figure 3.5. Ns-2 allocation scheme

First, to configure a particular allocation discipline, several configuration parameters must be set for each ST:

- Maximum capacity allowed (in slots);
- Number of slots assigned by CRA ($N_{CRA}$);
- Maximum number of slots managed by RBDC ($N_{RBDC}$);

- Sampling weight $\gamma$ for the RBDC algorithm.

CRA is implemented by a module acting as a "setup filter". It runs at the beginning of the simulation and communicates to the *DAMA controller* the number of slots per frame ($N_{CRA}$) permanently granted to a given ST. As a consequence, the DAMA controller decreases the number of slots that it will dynamic assign "on demand".

The RBDC module computes capacity requests based on the principle of making input and output rate in the ST queue equal. To this purpose, it acts as a filter that, every superframe, gets as input the amount of data in the queue not served by CRA, computes the RBDC request on the basis of both the $N_{RBDC}$ value and the following algorithm:

$$R_{RBDC}(k) = (1-\gamma) \cdot R_{RBDC}(k-1) + \gamma \cdot Q_k \tag{3.12}$$

where $Q_k$ is the amount of data queued at the $k^{th}$ superframe and $\gamma$ is a weighting factor that determines the amount of "memory" of past history.

The VBDC discipline is managed through a *DAMA Entity module*. Every superframe, a capacity request is computed by using the following algorithm [31]:

$$R_{VBDC}(k) = \lceil Q^{'}(k) - A(k) - \sum_{j=1}^{L-1} R(k-L+j) \rceil^{+} \tag{3.13}$$

The inputs to the VBDC algorithm are:

- the amount of data ($Q_k^{'}$) not processed by other access schemes (i.e., CRA, RBDC);
- the capacity allocated for the considered ST in the current superframe ($A(k)$);
- the requests sent in the previous superframes but not yet taken into account by the DAMA controller.

The parameter $L$ indicates the "resource allocation periods" between the transmission of the capacity request and the activation of the corresponding capacity (advertised by the Burst Time Plane - BTP), named "System Response Time" or "access delay". It includes the propagation delay and the processing delays at both

STs and GW sides. A dedicated module, called *Pending Requests*, stores such requests.

A *DAMA Controller*, logically located in the GW, manages the request scheduling and, every superframe, assigns the available slots on the basis of the received requests. Depending on the assigned capacity, the *DAMA Controller* regulates the transmission time of the queued data (managed by the *MAC Object*).

The implemented allocation scheme has been designed to respect typical values concerning both the frame structure and processing delays. In the default setting, the frame supports a capacity of 2048 kbit/s and is divided into 32 units/slots. The frame duration is 24 ms, while a superframe is constituted by 4 frames (the duration is 96 ms). The "System Response Time", defined as the time between the request sending and the corresponding capacity assignment, is set equal to 11 superframes (1056 ms).

A vast gamut of statistics (graphs or data files) can be obtained as simulation outputs. In particular, such outcomes can be divided into three categories:

- *Lower layer statistics.* These concern the allocation dynamics at the MAC layer.
  - The "System Response Time" measurement;
  - Trace of the slots allocated in each superframe;
  - Calculation of the "Saturation Time", the time at which the maximum sustainable transmission rate has been achieved.
- *End-to-end statistics.* Statistics gathered in the end-systems.
  - RTT;
  - Throughput.
- *TCP statistics.* Monitoring of the evolutions of the TCP internal parameters.
  - *Cwnd* and *ssthresh* trend;
  - Trace of the retransmissions.

### 3.7.2 Network Engineering Platform

Network Engineering Platform (NEP) is a network emulation tool developed by Alcatel Alenia Space, initially for internal use, but later upgraded for more general use. It was used as emulation tool. NEP aims at emulating DVB-S/DVB-RCS satellite access systems for what concerns network, access and resource management layers are concerned. It provides a real-time environment to assess and benchmark the performance of both popular internet applications and satellite dedicated applications. NEP is composed of 6 interconnected Linux PCs covering all the functionalities of a real satellite access network DVB-RCS like:

- 3 PCs acting as STs;
- 1 PC implementing GW and NCC functions;
- 1 PC providing the Terrestrial Interconnection Sub-System (TISS);
- 1 configuration station implementing the "Man Machine Interface" (MMI) and coordinating NEP operation.

### 3.7.3 Real Measurements

Some real TCP tests have been performed through commercial hardware. Two systems have been used:

- Newtec terminals connected in a DVB-RCS network operated commercially by Belgacom;
- Viasat terminals connected in a satellite network operated commercially by Eutelsat.

### 3.7.4 Metric and comparison criteria

In this section the main criteria and the most important guidelines to evaluate and compare (to the standard TCP) a TCP-based protocol are briefly described.

#### 3.7.4.1 Performance

Performance at the transport layer can be expressed in terms of:

- Throughput (packet/s, bytes/s);
- Perceived latency (RTT measurements);
- Steady-state time (time needed to saturate the available capacity).

#### 3.7.4.2 TCP friendliness

Friendliness addresses the capability of TCP-compatible flows to behaves under congestion like a flow produced by a standard TCP. In other words, a new TCP-compatible protocol should not take away too much bandwidth from standard TCP flows while utilizing all the unused bandwidth.

#### 3.7.4.3 Fairness & Channel Utilization

Fairness addresses the capability of a TCP-compatible flow to fairly share the overall capacity with other flows. It is classified as:

- inter-protocol fairness;
- intra-protocol fairness.

The former concerns the presence of both standard TCP flows and TCP compatible flows. Then, it falls into TCP friendliness. The latter concerns fairness among connections using the same protocol. In this case, fairness measures how flows using the same transport protocol use the shared capacity. Intra-fairness can be quantified through the Jain's index [32]:

$$Fairness\ Index = \frac{(\sum x_i)^2}{n \cdot \sum (x_i)^2} \tag{3.14}$$

where $x_i = \frac{T_i}{O_i}$ is the normalized average throughput for the connection $i$ ($T_i$ = measured average throughput, and $O_i$ = fair average throughput), and $n$ is the number of connections.

Another important parameter to evaluate related to fairness is the total channel utilization. It can be easily calculated through the following formula:

$$Total\ utilization = \frac{\sum T_i}{B} \tag{3.15}$$

where $B$ represents the channel bandwidth.

#### 3.7.4.4 Practical Implications

To implement some TCP-based mechanisms, modification at operating system kernel are required. Others can be implemented as a user level library easily recalled by applications. Complexity in either adding features to the standard TCP or replacing TCP with another transport protocol is an important parameter of quality.

## 3.8 TCP dynamics over satellite links

This section reports the most relevant results of both the theoretical and experimental analysis performed by the author concerning TCP dynamics over satellite links.

### 3.8.1 TCP mechanisms over satellite links

Without losses, the time necessary to reach a window of size $W$ segments can modeled as follows:

$$Ramp\ up\ Time = RTT \cdot (\log_2 W + 1) \tag{3.16}$$

Considering a broadband satellite with a long RTT (i.e. 500 ms), and assuming 1500 byte segments and an initial *ssthresh* value (set as the a*dvertised window* value) equal to 256 segments ($W$), the SS algorithm takes about 4.5 s to reach the advertised window, while in the same conditions TCP running over a terrestrial network with an RTT of 40 ms takes only 320 ms. As shown in Figure 3.6, TCP performance can drastically deteriorate in a satellite environment, especially in the cases when the data transfers are much shorter than the delay-bandwidth product of the network (e.g., downloading a Web page).

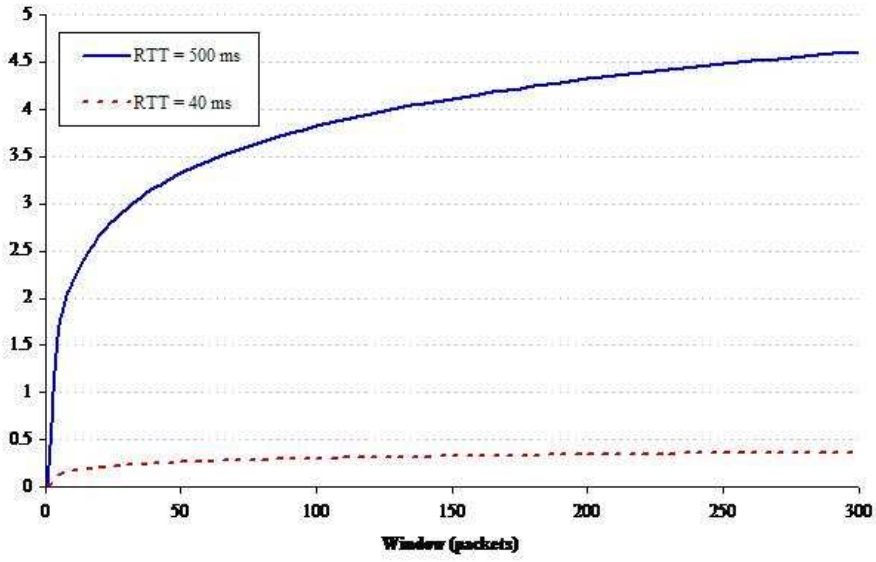


Figure 3.6. TCP rump up time vs. window size

At the end of the Slow Start phase or after a loss is detected by duplicate ACKs, TCP uses the Congestion Avoidance algorithm to probe more gradually the additional link capacity. In fact, *cwnd* is increased respecting the Equation (2.4). Of course, as with Slow Start, the amount of time the congestion avoidance algorithm takes to increase *cwnd* using a satellite link is longer than for a terrestrial link. To quantify such impairment, we consider the same case previously described. In particular, it is supposed that a packet loss occurs when the window size is equal to 64 packets. In this case, as seen in section 2.7.4, the Fast Recovery algorithm halves the cwnd size (32 packets) and the connection enters Congestion Avoidance mode (to reach a window of 100 packets TCP takes 8 s over a terrestrial link with a RTT of 40 ms while it takes 46 s over a satellite link with a RTT of 500 ms as shown in Figure 3.7).

#### 3.8.1.1 Impact of real Operating System features on TCP performance over satellite

As discussed in the Section 3.2.5, TCP flow control is regulated by the "Advertised Window", which is depending on the receiver buffer size. Figure 3.8, obtained through Ns-2 [30] simulations, shows the average throughput achieved for different

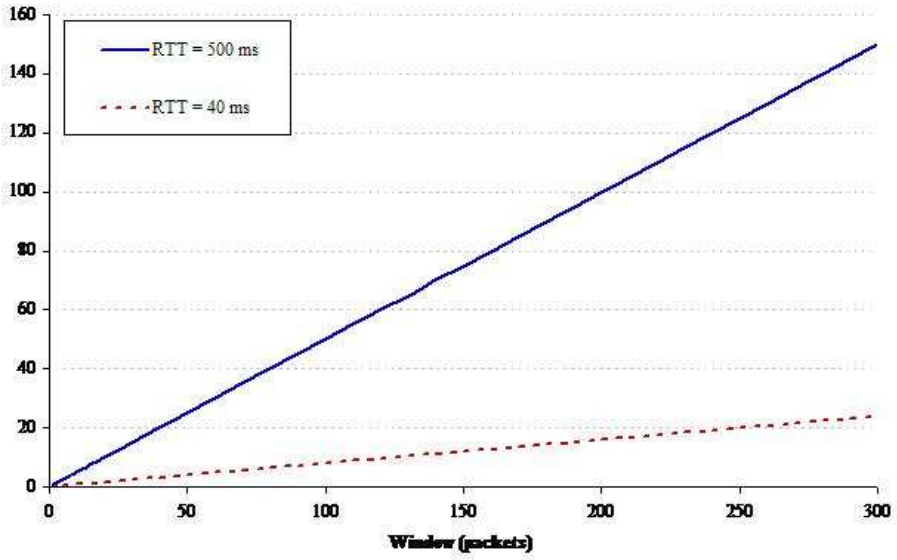


Figure 3.7. TCP congestion avoidance time

values of Packet Error Rate in the satellite link and different values for the buffer size in the receiver host.

In detail, the main simulation parameters are following listed:

- Physical RTT = 508 ms;
- Packet Error Rate [0,$10^{-4}$,$10^{-3}$];
- Bandwidth= 4 Mit/s

In the error free case, the average throughput grows linearly up to the buffer size catches up the data pipe value ($\cdot$ 170 packets), and remains constant for greater values. The presence of errors make the average throughput floating for buffer sizes close to data pipe value. Anyway, the "saturation point" is always kept to about the pipe size.

### 3.8.2 HTTP over satellite

The World Wide Web (WWW) page retrievals represent one of the most important "source" of traffic in today's Internet. The impact of satellite-specific factors on the standard HyperText Transfer Protocol (HTTP) [38][39] performance is meaningful.

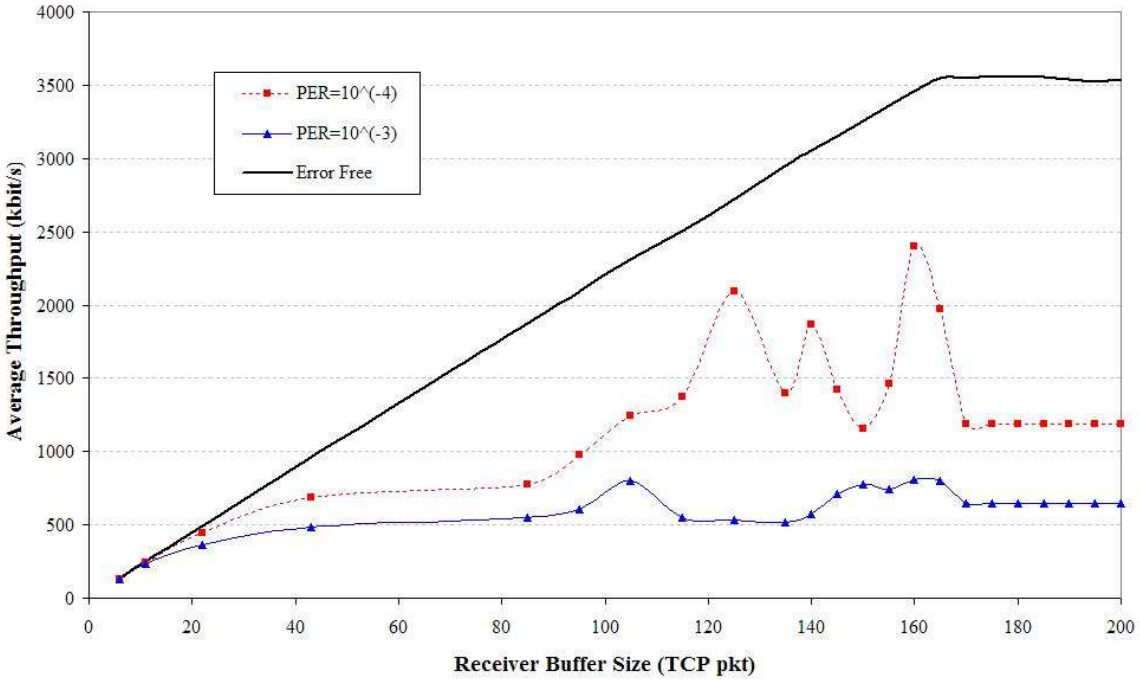


Figure 3.8. Receiver Buffer Optimization

Two HTTP versions are generally used in servers and browsers on the Internet:

- HTTP/1.0 [38];
- HTTP/1.1 [39].

HTTP/1.0 establishes a separate TCP connection to retrieve each HTTP object (i.e., HTML page, Java script, image, etc.). Therefore, the download of a typical WWW page requires many short TCP connections in parallel. This inevitably leads to relevant protocol inefficiencies due to standard TCP mechanisms (i.e., three-way handshaking, congestion control).

To improve on these inefficiencies, the "Keep-Alive" extension to HTTP/1.0 allows multiple requests to be sent over a single persistent TCP connection. However, the use of persistent connections does not preclude the triggering of multiple parallel TCP connections in many browsers.

The use of persistent connections was formally defined in HTTP/1.1 [39] and has been shown, in some cases, to halve the time to download a Web page [40].

HTTP/1.1 does not use multiple connections, but rather each element in turn on the same TCP connection. This leads to an interval of 1 RTT between subsequent object requests. This problem is faced by the "pipelining" option that allows HTTP/1.1 client to send simultaneously multiple requests (keeping the same connection) while receiving responses from the server. Due to the use of a single TCP connection, the HTTP/1.1 with pipelining can be easily modeled as follows:

1. the browser issues an HTTP `GET` command for the base HTML document;
2. One RTT later, the first base data document is received;
3. The browser then issues further `GET` commands for each page element referenced in the base document.

With reference to Figure 3.9, the cost of a HTTP transfer is:

- 1 RTT for the control-channel OPEN (TCP three-way handshake);
- $\frac{1}{2}$ RTT to send the request for the base data document;
- $\frac{1}{2}$ RTT to receive the base data document;
- time to get the whole document ($T_{transfer}$) (including time for sending the `GET` commands.

Then, the overall time to get a file in HTTP is:

$$2 \cdot RTT + T_{transfer} \tag{3.17}$$

where $T_{transfer}$ depends on TCP initial setting (i.e. initial *ssthresh*, initial *cwnd*), TCP mechanisms (i.e. Slow Start) and the file size (sum of all the element sizes). However, since typical WWW page have a limited size (hundreds of KBytes). To improve HTTP performance two main aspect must be addressed:

- reduction of the set-up phase (i.e. sending some data even during three-way handshake);
- optimization of the initial settings (i.e. use a large initial *cwnd*).

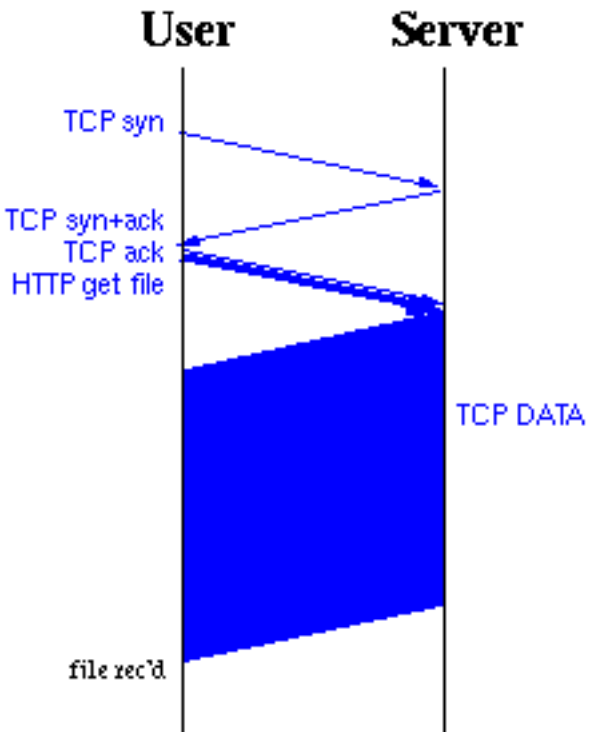


Figure 3.9. HTTP/1.1 with pipelining file transfer

### 3.8.3 DVB-RCS impact on TCP performance

In a DVB-RCS environment the use of DAMA mechanisms may introduce real RTTs well above the propagation delay, and the available bandwidth may vary strongly and abruptly. These two factors are essential in determining the achievable performance. To opposite, packet loss due to link errors is not a real problem, since DVB-RCS uses strong forward error correction at the physical layer, providing to the network layer and above a quasi error free channel. When TCP operates over a DAMA satellite link, this effectively leads to two nested control loops:

- TCP's rate control loop
- DVB-RCS's bandwidth allocation control loop.

The control bandwidth of both loops is of the same order of magnitude: somewhere between half a second and a few seconds. This leads to the two interacting control loops.

#### 3.8.3.1 DAMA performance

In terms of performance:

- CRA has a constant RTT of around 500 ms for a GEO satellite channel. Bandwidth is reserved regardless actual utilization. All the unused bandwidth is wasted.
- VBDC has a much longer RTT, typically around 1.5 s, due to the request/allocation cycle. Normally, capacity will only be requested for data that is already in the buffer, so the bandwidth utilization is 100%.
- RBDC has RTT similar to CRA, except some extra delay on first request and on adjustment of requested rate. The bandwidth provided will rarely match precisely the bandwidth needed, so that some bandwidth will be wasted most of the time, but less than for CRA.

Since real implementation details of VBDC and RBDC are not reported in DVB-RCS standard and information from commercial manufactures are not accessible, some assumptions are done. VBDC is rather straightforward, and there are not many choices available. RBDC, on the other hand, is much more open to different algorithms. Herein a relatively simple model based on a running average of recent traffic demand is chosen. It is reasonable that VBDC is intended for best effort traffic like TCP, while RBDC is intended for stream traffic that has more or less constant bandwidth requirements over extended time periods.

#### 3.8.3.2 An analytical model for TCP over DVB-RCS

At the beginning, TCP performs a "three-way handshake", in which the end-systems exchange the initial sequence number by sending SYN segments. The overall time needed for this procedure is equal to one RTT:

$$T_{TH} = RTT. \tag{3.18}$$

Then, data transfer starts when an application places data in its sending buffer and, assuming the absence of packet losses, can be divided into 2 phases 2:

1. Slow start phase;

2. Congestion Avoidance phase.

In SS phase, the receiver sends a cumulative ACK for each $b^{th}$ received segment (delayed acknowledgement mechanism [15]); then, each RTT the sender will increase its *cwnd* of $\frac{cwnd}{b}$ segments:

$$cwnd_{i+1} = cwnd_i + \frac{cwnd_i}{b} = \left(1 + \frac{1}{b}\right) \cdot cwnd_i = \gamma \cdot cwnd_i \tag{3.19}$$

more in general:

$$cwnd_i = cwnd_0 \cdot \gamma^i \tag{3.20}$$

where $cwnd_0$ is the initial congestion window.

Therefore, let $d_{ss}(ssthresh)$ be the number of packets expected to be transmitted in the SS phase, where *ssthresh* is the initial value of the slow start threshold, according to (3.20) the SS duration is approximately:

$$T_{SS}(ssthresh) = RTT \cdot \left[\log_\gamma \left(\frac{d_{SS}(ssthresh)}{2}\right) - \log_\gamma(cwnd_0)\right] \tag{3.21}$$

where RTT is the average round trip delay.

Then, TCP switches in CA phase the *cwnd* increase becomes linear (not delayed ACK in CA are assumed):

$$cwnd_{i+1} = cwnd_i + 1 \tag{3.22}$$

In addition, let $d_T$ be the total number of segments. Then, the amount of data to send in CA phase is:

$$d_{CA} = d_T - d_{SS} \tag{3.23}$$

According to the Equation 3.22, $d_{CA}$ can be expressed as function of both the number of RTT (N) needed to transmit it and the considered *ssthresh*:

$$d_{CA}(N,ssthresh) = N \cdot ssthresh + \frac{N \cdot (N-1)}{2} \tag{3.24}$$

Therefore, the overall time needed to transfer the $d_{CA}$ segments is:

$$T_{CA}(ssthresh) = RTT \cdot N(d_{CA}) \tag{3.25}$$

Definitively, the time needed to transfer a file of $d_T$ TCP segments is:

$$T_{TRANSFER} = RTT \cdot \left[ 1 + \log_\gamma \left( \frac{d_{SS}(ssthresh)}{2} \right) - \log_\gamma (cwnd_0) + N(d_{CA}) \right] \tag{3.26}$$

DAMA schemes, supported by the DVB-RCS standard, require an exchange of signaling messages to allocate capacity on a superframe basis. Basically, as long as new data arrive in the ST queue, a DAMA entity computes and submits capacity requests to the DAMA controller located in the GW. On the other side, the DAMA controller takes into account the requests coming from all the active STs and assigns capacity considering:

- the allocation category;
- the priority of the STs;
- the maximum allotment enabled in the different allocation categories;
- time limitations.

In this way, it builds the Burst Time Plan (BTP) and broadcasts it to all the STs. Such an allocation process is repeated every $n_s$ TDMA frames, corresponding to the superframe duration or "resource allocation period". The number $L$ of "resource allocation periods" between the transmission of the capacity request and the activation of the corresponding capacity (advertised by the BTP) is named "System Response Time" or "access delay". It includes the propagation delay and the processing delays at both STs and GW sides. Requests that cannot be satisfied in a certain resource allocation period are stored and served with priority in the next one. Therefore, if a volume-based allocation discipline (VBDC) is considered, the amount of resources per frame requested for the $k^{th}$ resource allocation period is given by [31]:

$$R_{VBDC}(k) = \left\lceil \frac{Q(k) - n_s \cdot a(k) - n_s \cdot \sum_{j=1}^{L-1} R(k-L+j) - n_s \cdot w(k)}{n_s} \right\rceil^{+} \quad (3.27)$$

where:

- $Q(k)$ represents the amount of data stored in the ST queue at the beginning of the $k^{th}$ resource allocation period;
- $a(k)$ is the amount of resources per frame assigned to the ST for the current superframe;
- $\sum_{j=1}^{L-1} R(k-L+j)$ is the sum of all the previous requests sent but not yet served;
- $w(k)$ indicates the owed resources by the DAMA controller to the STs, because in one or more of the previous allocation periods, the controller assigned fewer or ever none of the requested resources.

This criteria to compute the requests and the corresponding allocation process lead to an "access delay" quite constant and equal to the system response time, as long as congestion occurs ($w(k) \neq 0$).

Then, the average RTT perceived by TCP can be expressed as:

$$RTT(t) = RTT_0 + L \cdot n_s + \beta(t) \cdot n_s \quad (3.28)$$

where $RTT_0$ is the physical round-trip delay, $L \cdot n_s$ is the system response time and $\beta(t)$ is a component due to the congestion state of the network ($\beta(t) = 0 \Rightarrow w(k) = 0$).

In addition, DVB-RCS supports also CRA and RBDC allocation disciplines either as stand-alone access schemes or combined to VBDC discipline. In the latter case, ST could have slots allocated with different allocation disciplines. Thus, (3.28) can be re-formulated as follows:

$$RTT(t) = RTT_0 + L \cdot n_s + \beta(t) \cdot n_s - \alpha(t) \cdot n_s \quad (3.29)$$

The parameter $\alpha$ is the CRA/RBDC gain, and depends on both the number of slots assigned in CRA/RBDC discipline and the amount of data stored in the ST queue. In general, CRA and RBDC allow to decrease the average RTT ($\alpha$>0) as a consequence of the fact that CRA discipline allocates statically slots without dynamic negotiations, while RBDC computes request on the basis of the rate the data feed the ST queue and then, once that slots are allocated, they are booked also for a certain number of successive superframe.

Now, since TCP control mechanisms depend on the perceived RTT (3.26), which depends on the supported DAMA access scheme 3.29, the merge of 3.26 and 3.29 models the DVB-RCS/TCP nested control loops providing a theoretical tool to evaluate end-to-end performance under the different access schemes supported by the DVB-RCS standard.

Finally, the RTT presented in the 3.26 is assumed constant, while, as highlighted in 3.29, DAMA schemes could lead to meaningful variations in the RTT. Nevertheless, in the steady-state conditions, when TCP keeps constant its maximum window size ($\simeq$ bandwidth-delay product), the RTT becomes stable. In the final model, RTT(t) is approximated with its steady-state value obtained by replacing in the 3.29 the CRA/RBDC time-variant gain with the corresponding steady-state value. Therefore, assuming a network not congested ($\beta(t) = 0$), the time needed to transmit a file corresponding to $d_T$ TCP segments is given by:

$$T_T = (RTT_0 + L \cdot n_s - \overline{\alpha} \cdot n_s) \cdot \left[1 + \log_\gamma \left(\frac{d_{SS}}{2}\right) - \log_\gamma(cwnd_0) + N(d_{CA})\right] \quad (3.30)$$

where $\overline{\alpha}$ is the CRA/RBDC steady-state value.

### 3.8.4 Performance evaluation of TCP over DVB-RCS

In this section, performance evaluation is carried out using three different approaches:

1. Emulation through NEP (Section 3.7.2);

2. Measurements on a real system;

3. simulation through Ns-2 (Section 3.7.1).

In particular, emulation and real measurement outcomes have been used to validate the implemented simulation platform (Section 3.7.1). Once validated, the simulator has been configured to perform a large set of tests and collect a vast gamut of results.

#### 3.8.4.1 Emulation results

By using NEP, some tests have been performed by running Iperf [41] to create a TCP flow from a user terminal to the GW. Considering that the maximum achievable throughput was limited by the default buffer size, the optimum value for the buffer size was preliminary researched.

Figure 3.10 compares the RTT measurements obtained by considering the main aforementioned access schemes. In CRA and VBDC cases, a perfect match with the expected values can be observed (physical delay in the CRA case, physical delay + access delay in the VBDC case), excluding some higher values detected during the start-up phase. These higher values are likely to be artifacts of the detailed implementation of the NEP. As far as RBDC is concerned, some aspects in the RTT pattern need to be commented:

1. At the start up, the RTT values are very similar to those of VBDC. The rationale is that, due to the bursty nature of the TCP flow, RBDC needs some time to adjust to the real source activity. In fact, when the new rate samples are zero for several superframes, the RBDC gradually reduces the resources granted, and when new data feed the queue it behaves exactly like VBDC.

2. Since TCP traffic is bursty , RBDC is slow in establishing a stable estimate of the transmission rate, and the oscillations in the rate estimate lead to oscillations in the perceived RTT.

#### 3.8.4.2 Field Trials

Commercial systems use proprietary solutions as far as both request/allocation algorithms and TCP acceleration are concerned. Nevertheless, tests over real link are

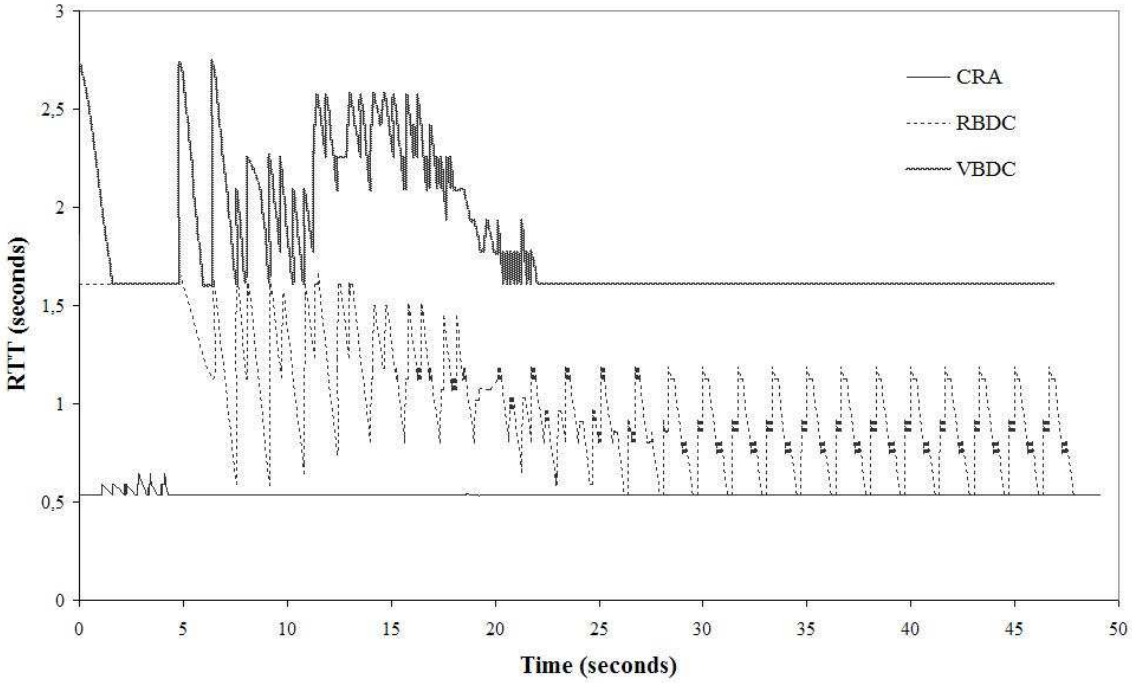


Figure 3.10. NEP tests: RTT measurements

useful to gather information concerning statistics of some physical parameters and subsequently use them to enrich simulated models. To this purpose, some tests have been performed to evaluate the Probability Density Function (PDF) of the satellite delay over a DVB-RCS network operated commercially by Belgacom and available at ESA premises. In particular, tests have allowed to obtained the probability density function of the observed RTT between an ST and the GW as shown in figure 3.11.

#### 3.8.4.3 Simulation results

Using the important input coming from emulation and real measurements, a deep analysis of the TCP behavior over DVB-RCS has been carried out using the simulator described in Section 3.7.1. The main goal was to highlight the dynamics of TCP over a DVB-RCS network and then identify the best access scheme for a targeted QoS. In particular, simulation activity has addressed the possibility to combine the different standard access schemes in order to study the correspondence between the

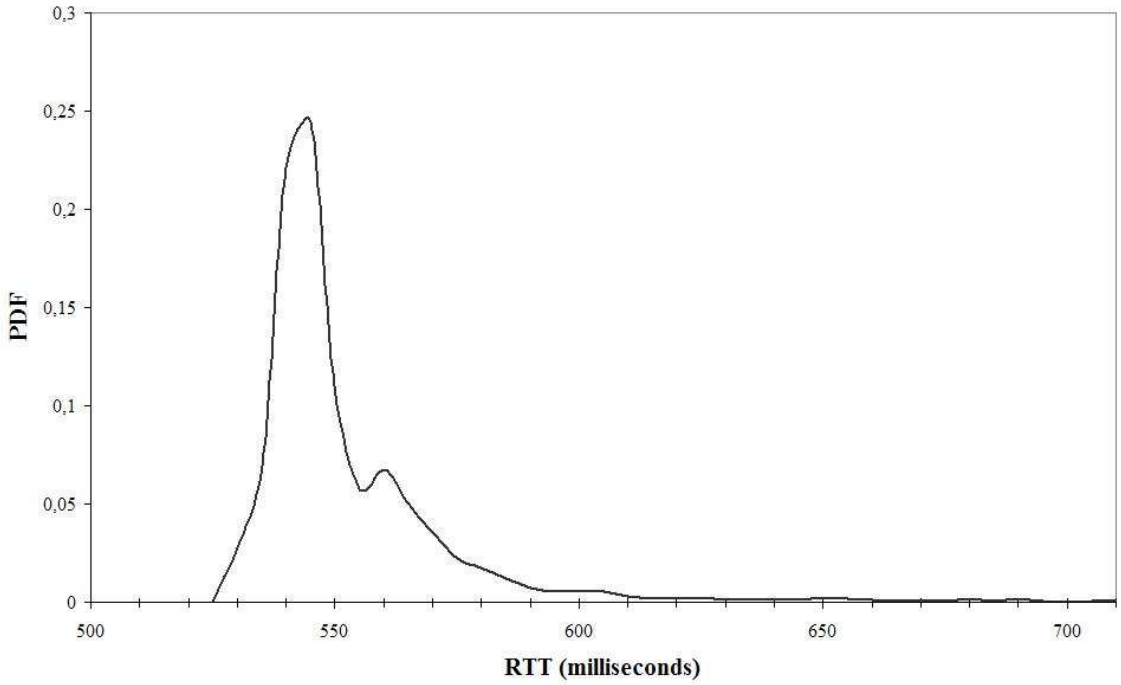


Figure 3.11. RTT distribution over CRA scheme

access policy and the achieved performance over a large set of alternative schemes. In the considered scenario, TCP source is located on the ST, while TCP receiver is located on the GW. Then, the analysis can be divided in two parts:

1. Configuration of DAMA profile based on the capability of the created simulation platform in handling layer 2 parameters/statistics (this is the main limitation when real equipments are used).

2. Evaluation on the performance perceived at the end-systems (higher layer statistics) correspondent to the provided QoS.

Figures (3.12)(3.13)(3.14) show graphs related to different access schemes. In particular:

- Figure (3.12) shows dynamics of the slots allocated per superframe, in the case a hybrid access scheme ("1 CRA slot" + "31 VBDC slots") is adopted.

- Figure (3.13) overlaps the graphs of data queued in the ST buffer and capacity (in bytes per superframe) allocated, when RBDC ($\gamma$ = 0,1) is considered as access scheme. The oscillatory behavior observed at the beginning is due to the joint effect of the request-assignation loop and the bursty nature of the TCP transmission. At saturation, the amount of enqueued data always exceeds the maximum capacity, then all the capacity is constantly allocated, while queue occupation grows.
- Figure (3.14) traces the RTT in the case an RBDC ($\gamma$ = 0,1) access scheme is considered.

Then, due to the flexibility of the simulation approach, it appears evident how to use the simulation for obtaining the most accurate statistics and to have complete control of all the parameters at both the MAC and the transport layers.

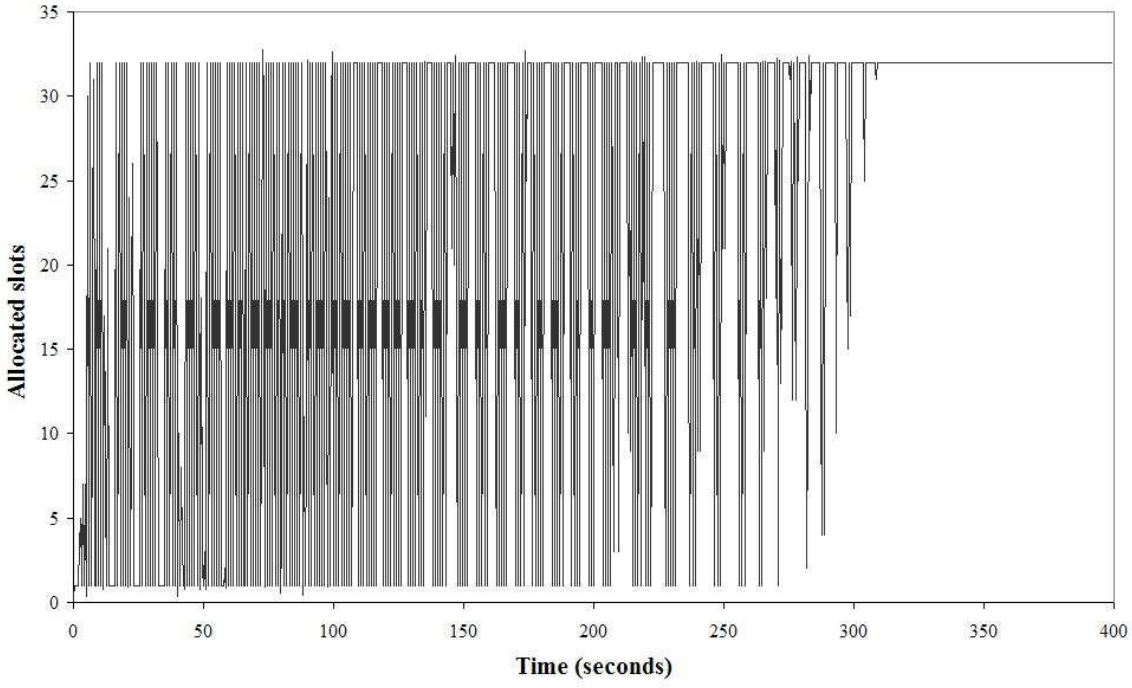


Figure 3.12. DAMA allocation dynamics (31 VBDC+ 1 CRA)

As seen in the theoretical analysis (Section 3.8.3.2), the key parameter for TCP performance is the RTT. For this reason, by associating to each access scheme an

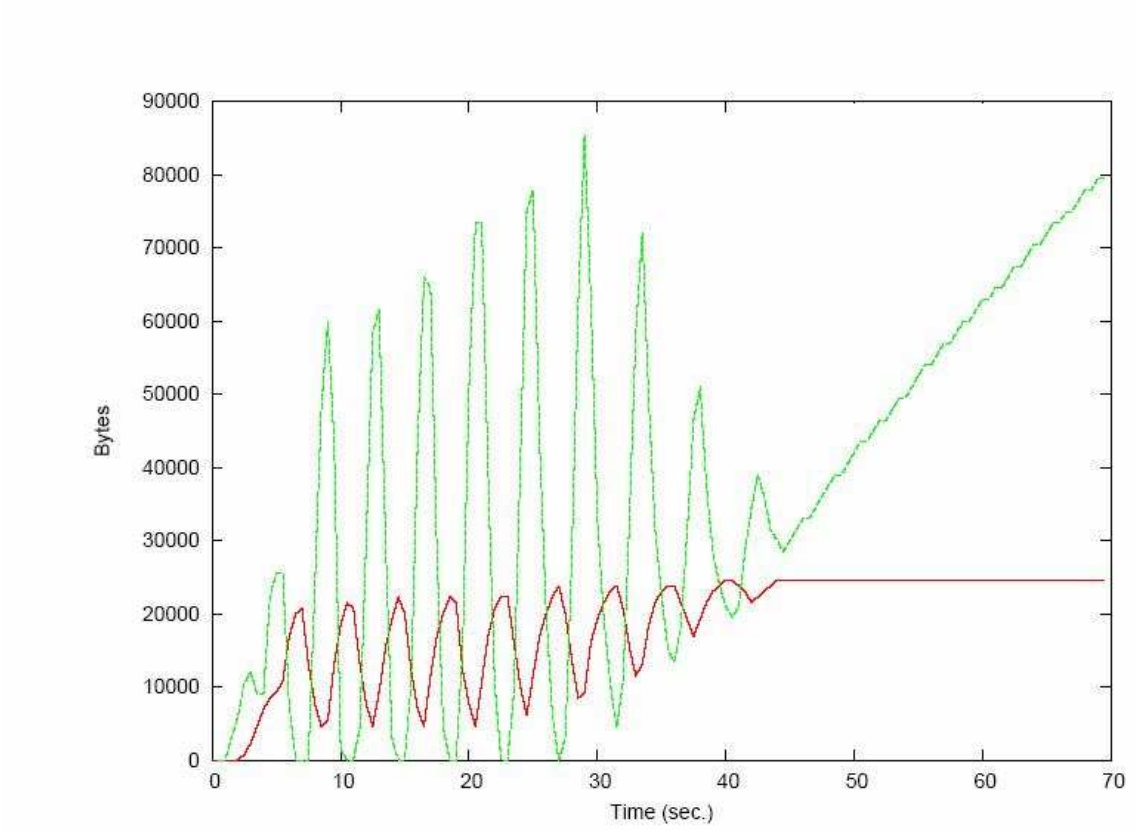


Figure 3.13. RBDC capacity allocation dynamics

estimated RTT value, it is possible to deduce the corresponding theoretical performance perceived by the end-systems.

Figure 3.15 shows the RTT values for various access schemes, representative of the main allocation classes herein treated and the corresponding throughput achieved by the TCP source. In particular, it refers to the RTT perceived by TCP over 4 different access schemes:

- CRA;
- VBDC;
- RBDC ($\gamma = 1$);
- Combination of VBDC and CRA (5 slots statically assigned in CRA, and the

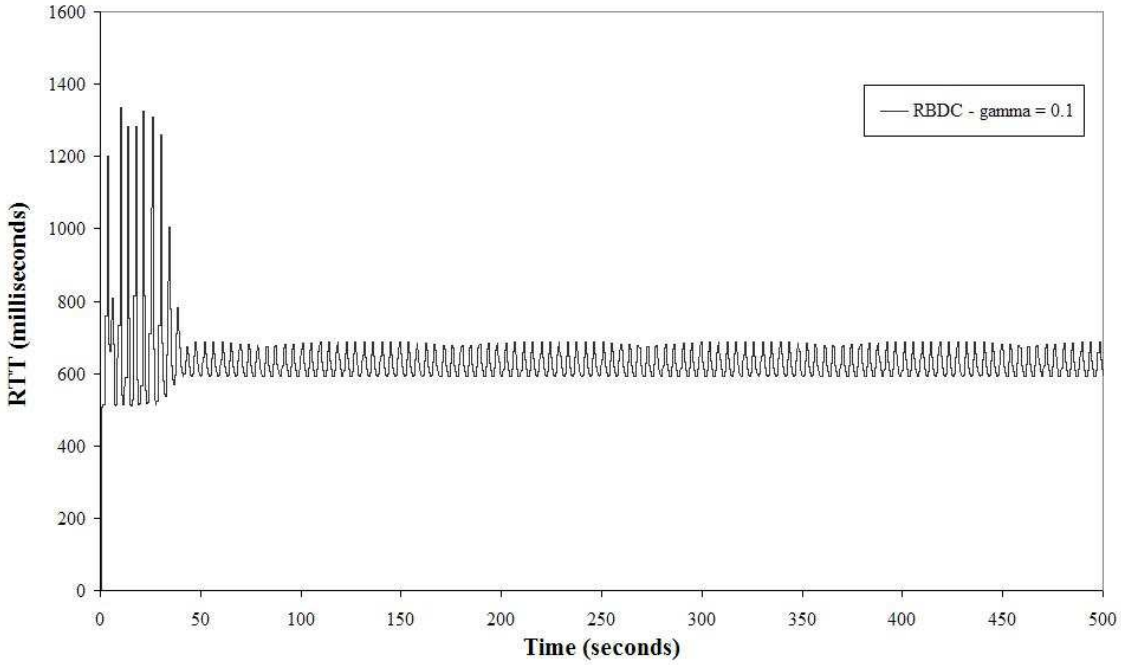


Figure 3.14. RTT over RBDC

remaining 27 slots negotiable by VBDC).

In the simulations, the receiver buffer size is set equal to about 1 MByte in order to avoid eventual contributes in the perceived RTT due to ST "internal" congestions, occurring when the amount of data feeding the buffer exceeds the bandwidth-delay product.

Observing the trend of the graphs, some comments derive:

- VBDC and CRA are characterized by constant RTT values, respectively equal to the physical delay and the physical delay + the access delay (as defined and discussed in Section (3.8.3.1));
- RBDC leads to oscillations in the RTT at the beginning due to the bursty nature of the TCP traffic, while when saturation occurs (the maximum TCP window is achieved) the RTT becomes quite stable. In particular, the oscillations are caused by the periodicity of two phases in the request computation ($R_k = 0$):

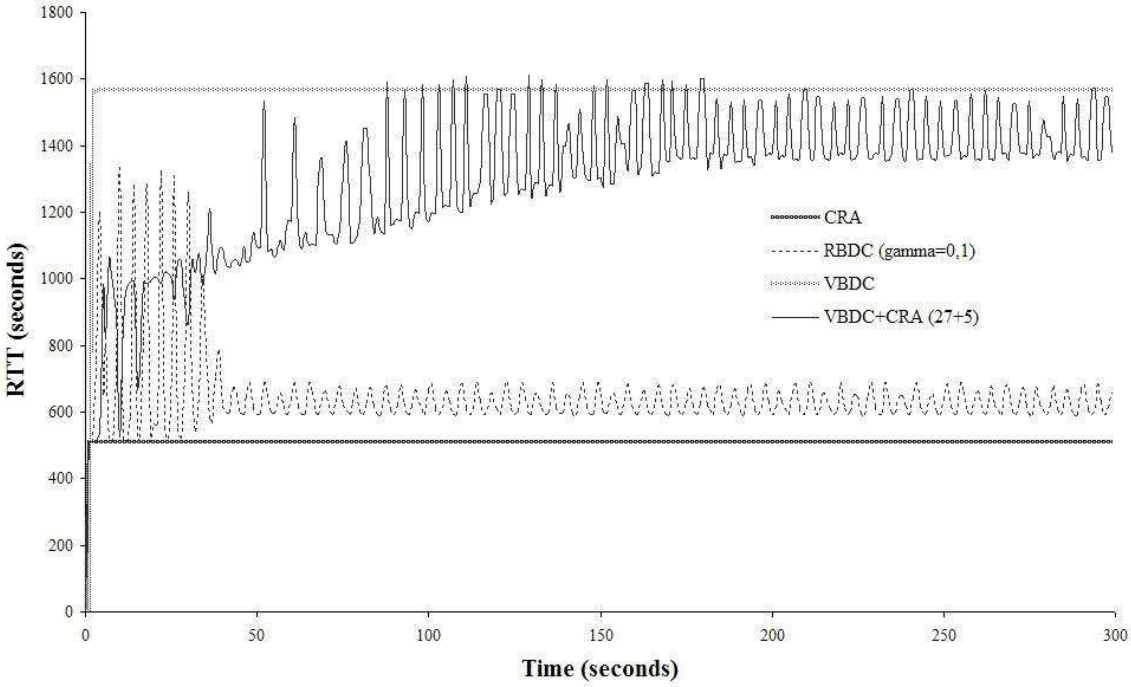


Figure 3.15. RTT evolutions with different DAMA schemes

1. In the previous superframes $R_{k-1} = 0$. Then, when new data feed the queue, the time spent in the request-allocation process is very close to that one needed in the VBDC.

2. In the previous superframes $R_{k-1} \neq 0$. Then, with reference to Equation (3.12), the new request will be not null even if there is no data in the queue. Consequently, when further data will feed the queue, a given amount of capacity could be already assigned.

- In the "hybrid" VBDC/CRA access schemes, since 5 slots are statically assigned, the dynamic allocation process concerns just a part of the stored data. Therefore, when the amount of data is less or equal to the assigned capacity, the RTT involves just the physical delay; in contrast, when the amount of stored data grows, a part of capacity is requested by the VBDC algorithm (Equation 3.132), and the RTT is increased as well, until it approaches a saturation value that, on average, is less than that perceived whit the VBDC

scheme.

Figure 3.16 shows the throughput variations related to the four access schemes considered in Figure 3.15. These graphs summarize all the conclusions achieved so far, highlighting the time needed to exploit the total capacity (throughput = 2048 kbit/s), called saturation time, and the dynamics of the allocation algorithm at the start-up (oscillations in the throughput).

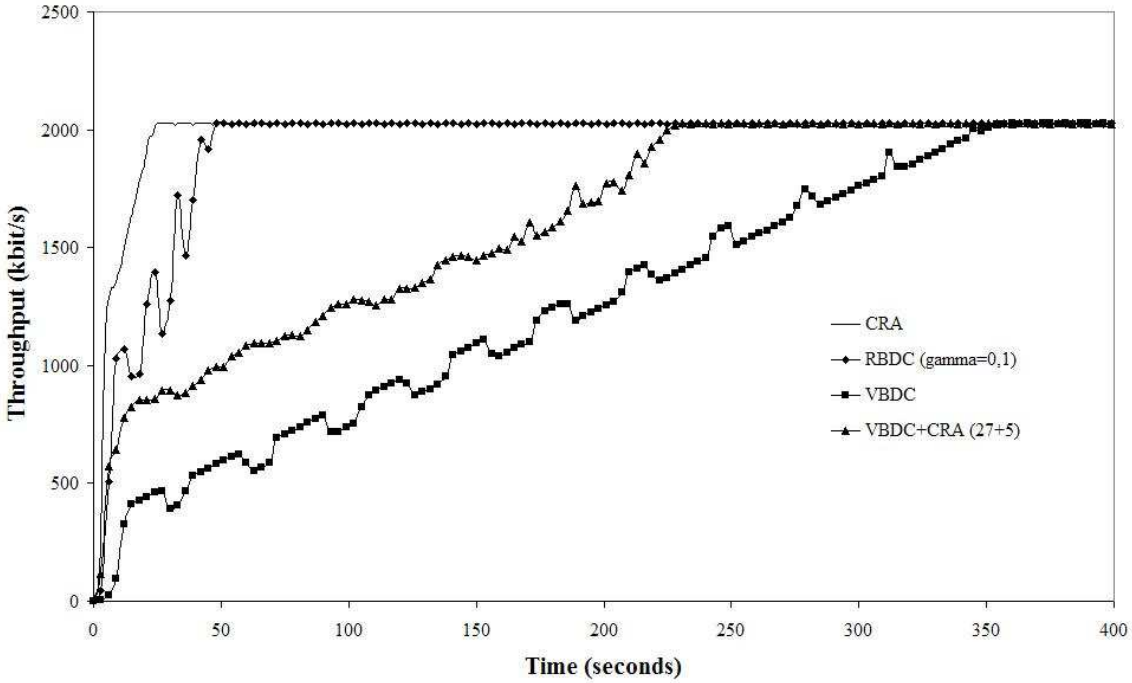


Figure 3.16. Throughput measurements for different DAMA schemes

The last set of results, illustrated in the figure 3.17, concerns the simulation of file transfers by FTP from an ST to the GW. All the standard DVB-RCS access schemes ,all the possible combinations between VBDC and CRA and different weights $\gamma$ (0,1-0,5-1) in the RBDC algorithm (Equation 3.12) have been considered. Moreover, the tests have been carried out with a small file of 500 kbytes and a large file of 10 Mbytes in order to distinguish short and long term behaviors. The transfer time has been normalized to the transfer time detected in the CRA case.

Analyzing the figure, some comments derive:

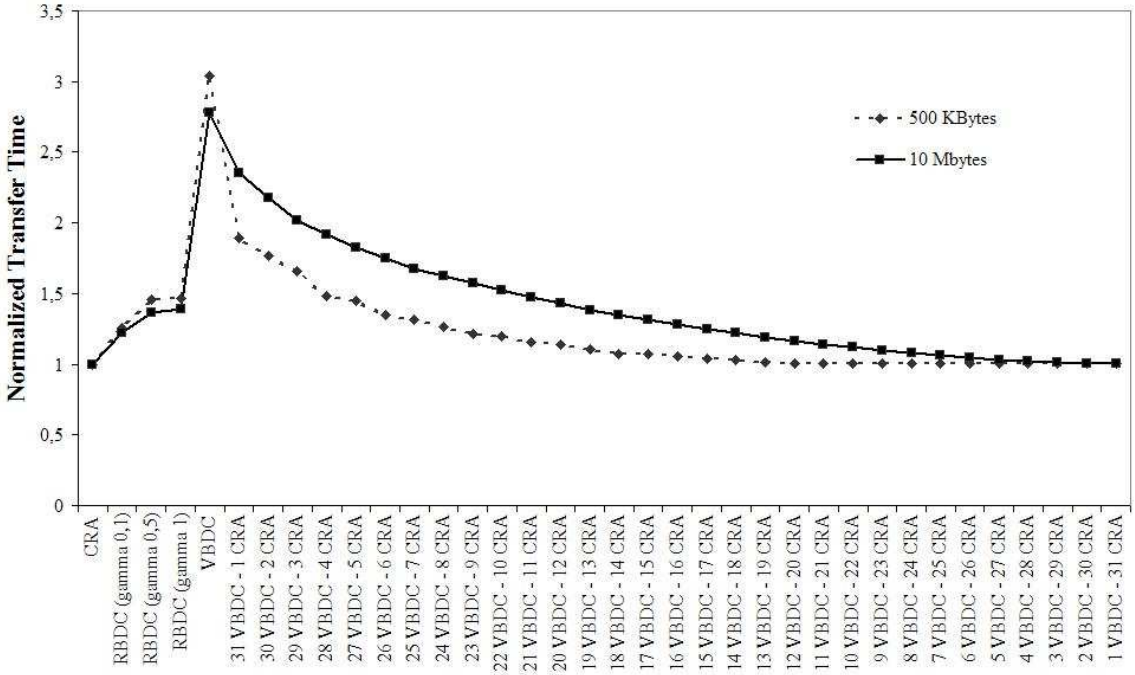


Figure 3.17. Normalized file transfer time

- The normalized transfer time over the VBDC access scheme is equal to 3. This validates the conclusions achieved by theoretical model expressed in (3.30), according to which the transfer time is proportional to RTT. In fact, the RTT for VBDC is about 3 times the RTT of CRA.

- RBDC scheme presents better performance for large files than for small files. This confirms the fact that RBDC suffers from the bursty nature of the TCP traffic during the start up phase.

- Lower $\gamma$ values makes the RBDC algorithm more stable (despite the network efficiency) by leading to better performance.

- The hybrid CRA-VBDC scheme provides higher gains in the transfer time in the case of short files. The rationale is that the percentage of the data served by CRA is higher.

# Chapter 4

# Enhancing Transport Layer for Satellite-based Internet

## 4.1 Enhancing TCP in HAPS-Satellite Integrated Architecture

To ensure telecommunication capabilities in emergency scenario requires the use of challenging architectures. The concept of using unmanned objects [82][83] flying (HAPS/UAV) or stationary (Sky Station) at relatively low altitudes to provide backup or support capacity to high traffic areas (hot spots) has been previously introduced. The satellite is intrinsically suitable to provide service in such a scenario, not depending on ground situation. If the satellite is used in combination to other systems, for example HAPS/UAV, capable to easily offer and enhance hot spot coverage with limited latency, the combined architecture results to be even more suitable. In fact, while the HAPs/UAV can provide short range wireless connectivity even with small user terminals, being the link not so critical, the satellite can ensure large bandwidth for very long range connectivity, to reach a remote headquarter all over the world. The two systems can be jointly used with the HAPS/UAV collecting information at high rates from user terminals thanks to the very short range and high elevation angles achievable even in urban areas, while the satellite provides interconnection with the terrestrial infrastructure.

In this scenario, the performance of some conventional protocols developed for

Hot Spot environments present some constraints when used in this rather unconventional environment. In fact, with the HAPS/UAV + Satellite connection an higher loss rates is expected on the ground to UAV link than in conventional Hot Spots because of distance, obstacles and UAV motion. To characterize such a scenario classical channel models, developed for satellite environments, can be suitably applied [84][85]. Moreover, the satellite link has a large propagation delay and may introduce random loss of its own. On the basis of the analysis carried out in Chapter 3, applications running on TCP particularly suffer from the characteristics of this environment.

In this framework, two different ways of maintaining TCP connections have been studied:

1. Optimization of the end-to-end connections, from ground user to Internet server. Replacing of the legacy TCP NewReno (TCPNR)(Section 3.3.2.1) with the Adaptive Bandwidth Share Estimation (ABSE) [66] version of TCP Westwood (TCPW) (Section 3.4.8, optimized for leaky links with large pipes.

2. Proxy server on board of the UAV. The idea is to split the TCP connection (Section 3.5.2) on the HAPS/UAV and thus reduce the problem into two more tractable subproblems, i.e., a very lossy link with short propagation delay; and a more reliable (or at least, more predictable) link with very large propagation delay. In this case, different TCP "flavors" may be needed for the differing links characteristics.

### 4.1.1 System architecture

The overall system envisages an urban scenario with mobile users (pedestrians and cars) connected by a very efficient communication infrastructure comprising the cellular system as well as a network of Hot Spots placed in strategic locations (i.e. busy street crossings, tall buildings, parks, etc.) to access multimedia services. In particular, the data services (mainly, access to the Internet from mobile users), if the trend is confirmed, will see a dramatic growth in the next few years, owing to the introduction of new mobile services such as location based resource discovery (i.e., nearest drugstore), navigation support, web access etc. In this environment,

that is becoming increasingly dependent on communications, in case of emergency situation when power goes out and all the Hot Spots and Cellular Base Stations shut off, communications come to a standstill until power and telecommunication systems are restored. In addition, communications are most needed to control vehicular traffic and to allow users to "navigate" their way out of the traffic congestion. At the same time repair crews, police and medical teams need efficient communications to coordinate their work. A similar emergency scenario in an urban area occurs when there is a chemical, nuclear disaster caused by human error, plant break down, act of war, or terrorist attack. Again in such situations the communication infrastructure has been impaired while the need for efficient communications remains critical. In the depicted scenario, to deploy in a short time a system composed of several HAPS/UAVs to establish an emergency telecommunication infrastructure is very cost effective and easy. Moreover, being simple and unmanned flying objects, HAPS/UAVs can be deployed very rapidly and flown even in environments very dangerous to humans, as those polluted by chemical spills or nuclear slag. Once in place, the HAPS/UAV will act like a Hot Spot to the customers on the ground permitting them to communicate among one another but also to access the Internet, or a remote headquarter, via Satellite. The HAPS/UAV may fly through the "urban canyons" acting as repeaters and propagating the received signal to a wireless receiver on the ground, even exploiting paths that involves also other HAPS/UAVs, or to a GEO satellite connected to a gateway located very far from the disaster area. Another realistic scenario involves a mobile satellite station, a truck provided with antennas both for Wi-Fi and for satellite, as depicted in Figure 4.1.

In this case HAPS/UAVs can be relieved of any functions, and consequent machinery weight, involved with satellite transmissions. HAPS/UAVs can thus be as simple as possible, requiring very little maintenance and rare upgrading, while the latest technology innovation can be easily implemented in the on-wheels satellite station. Not only the presence of several flying units can greatly enlarge the area covered by the proposed architectures but also the exploitation of path diversity can be particularly functional to improve availability in an urban environment. Indeed, this two level "satellite empowered" architecture includes the advantages of very small mobile terminal technology, the ability of handling high data rates, and the attitude to establish very long range connections.

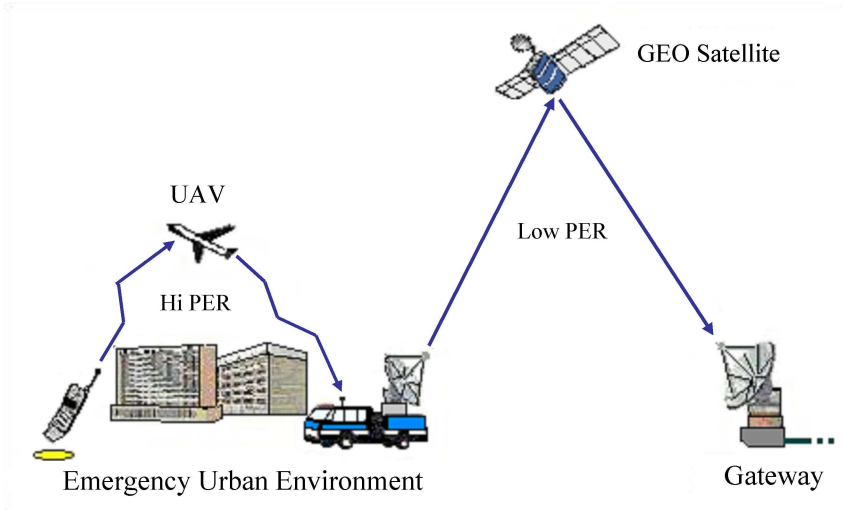


Figure 4.1. Reference scenario

Looking for effective bandwidth access in the presented scenario, efficient algorithms must be implemented to guarantee wireless error resilience.

#### 4.1.1.1 High Altitude Platforms

The proposed solution for an emergency urban environment demands for highly efficient broadband and multimedia services. The ideal goal is to build a wireless network able to efficiently cover a wide area with low propagation delays and having a negligible multi-path fading. In this context, radio communications could be based on higher frequency bands looking for larger bandwidths. Unfortunately, the use of this spectrum of frequencies implies line-of-sight propagation between the base station and each customer terminal since local obstructions cause problems.

The concept of HAPS is not new. What is quite new is the willingness to utilize such platforms as a structure to locate a communication payload. In this way the capability to offer connectivity over large areas utilizing a device located at a few kilometers (in the near space), represents a trade off solution between very distant satellites and terrestrial repeaters. In fact, they are so close to the Earth surface to not requiring complex launch, they can be easily moved from a position to another, they can be easily repaired and re-launched. Moreover, free

space losses and propagation delay are not critical at all. The flying objects usable for this scope can be classified as stationary (blimps) and moving, both usually conceived as unmanned (even though someone has proposed also manned vehicles). The balloons have the advantage to appear fix with respect to an Earth observer but the mechanical stabilization is extremely critical. On the other hand, the moving platforms needs to be tracked from ground even though they can fly over limited areas. The coverage area of a platform can typically have a diameter up to 200 km divided in cells of 1-10 km. High Altitude Platforms (HAPs) located in the stratosphere at 17-22 km can carry communication relay payloads and operate in a quasi-stationary position. The payload can be a complete base-station, or simply a transparent transponder. Therefore, HAPs can combine the best features of both terrestrial and satellite systems leading to a powerful integrated network. A single HAP can replace a large number of terrestrial resources covering a wide area by a flexible cellular frequency re-use structure and providing services more economically. Basically, HAPs are aircraft or airships (essentially balloons, termed "aerostats") and can be manned or unmanned; in the latter case, autonomous operations can be coupled with remote control from a control ground station. Throughout the evolution of HAPs, three types of vehicles can be distinguished:

1. *Unmanned Airships.* These can range from small expendable balloons flying at modest altitudes to large airships operating at c. 21 km altitude. Typically, the balloons are solar-powered and filled with helium.

2. *Unmanned Aircrafts.* This type of HAP is an unmanned solar-powered plane, which can fly against the wind or in a roughly circular tight path to maintain their position as stable as possible. These platforms are power-limited and need to store sufficient energy for station-keeping throughout the night. A fueled unmanned version of these aircrafts, flying generally at modest altitudes, is specifically known as Unmanned Aerial Vehicle (UAV).

3. *Manned Aircraft.* These are represented by the conventional fueled aircrafts.

### 4.1.2 Simulation scenario

In order to verify the performance gain obtained by the combined use of a splitting architecture and TCP Westwood (TCPW), the NS-2 simulator [30][17] hab been chosen as the testing tool. This platform, in fact, is widely adopted for network scenario thanks to the reliability of its outcomes and the validity of its models, especially in new transport protocols investigation.

In particular, new C++ code has been added to simulate the behavior of TCPW and the splitting scheme at the HAPS/UAV. In addition, a OTcl script has been written to configure all the set of simulations required to test the diverse network set-up. This allowed to obtain meaningful and fully understandable results by precisely configuring each single component of the scenario:

- Packet size;
- Queue size at nodes and cache size at the proxy (when present);
- Location of the proxy (when present);
- Propagation delay of each link;
- Capacity of each link;
- Error rate on each link;
- Number of flows present on the channel;
- Protocols involved.

Finally, the possibility to use different random seed in generating some network conditions, as the time occurrence of the wireless errors, allowed to run several different simulations of the same network set-up and then to statistically average the results.

A simplified summary of the general topology adopted in the simulations is depicted in Figure 4.2, which includes some information regarding configuration parameters. Each of the circles in the picture corresponds to one of the actors in the proposed scenario, and specifically:

- $W$ - generic wireless device in the urban area;
- $U$ - HAPS/UAV acting as a signal repeater;
- $T$ - Truck with a satellite station (eventually with proxy on board);
- $S$ - GEO satellite (eventually with proxy on board);
- $G$ - gateway on the ground

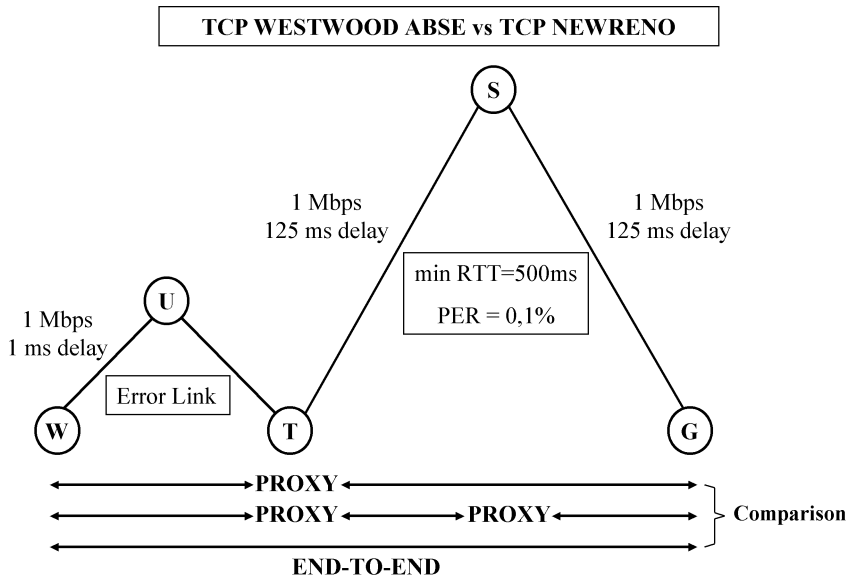


Figure 4.2. Configuration of the simulated scenario

The edges between the various nodes, represent wireless connections. The propagation delay between $W$ and $T$, through $U$, is very short because of the flying altitude of the HAPS/UAV, while a typical GEO satellite delay (125ms) is considered between $T$ and $S$, and between $S$ and $G$. Since errors are a well known characteristic of wireless links, every edge in the scenario requires the presence of an adequate percentage of packet losses. However, since technologies available for satellite links permit to have a quite acceptable reliability on this kind of channel [86][87], a constant PER (Packet Error Rate) of 0.1 is set between T and G. Instead, the links between W and T can be charcterized by a wide variety in the possible

urban condition. For this reason, more than one single PER value has been considered, specifically looking for higher loss levels than that considered in the satellite link. The bandwidth available on each link is 1 Mbit/s and the packet size is 1500 bytes, thus having a pipe capacity of about 42 packets on the round trip connection. The buffer available between each couple of adjacent nodes is 50 packets, while a cache of 200 packets have been utilized by each proxy, if not differently specified, when splitting mechanism is enabled. In the performed analysis, each simulation run utilizing different combinations of transport protocols, various PER on the ground-UAV-ground wireless link, presence or absence of a proxy on the ground satellite station and on the satellite, diverse dimensioning of the cache in the proxy, and alternate direction of the data flow. Summarizing:

- Transport protocol
  - TCPNR, TCPW;
- PER on the link between $W$ and $U$
  - 0.1%, 0.5%, 1.0%;
- proxy on board
  - on the ground satellite station - split enabled, split disabled
  - on satellite - split enabled, split disabled;
- cache size on the proxy
  - 10, 20, 30, ..., 240, 250 packets;
- traffic direction
  - from $W$ to $G$, from $G$ to $W$.

Simulations have been arranged to calculate, for each different configuration, the average throughput during a period of 230 seconds and the time needed to transmit a 5 Mbyte file. Every run has been replicated twenty times changing the seed value of the random generator, thus producing averaged final outcomes.

### 4.1.3 Results

A first set of results concerns performance benefits achievable combining the use of TCPW and a splitting proxy. The analysis is performed at different error loss rate on the link between the wireless device and the terrestrial satellite station $T$, through the flying HAPS/UAV. Figure 4.3 shows the time required to transmit a 5 Mbyte file from $W$ to $G$. A significant advantage is obtained over lossy links implementing TCPW instead of the common TCPNR. In particular, when employed in the end-to-end configuration, TCPW requires only from 59.33% (with 0.1% PER) to 34.99% (with 1% PER) of the average time needed by the TCPNR. Moreover, when intermediate proxy is implemented, the gap between TCPNR and TCPW performance is strongly reduced. If compared to TCPNR, in fact, TCPW only needs from 72.52% (with 0.1% PER) to 64.70% (with 0.5% PER) of the time. Even individually applied, TCPW and splitting scheme lead to a sensible performance improvement.

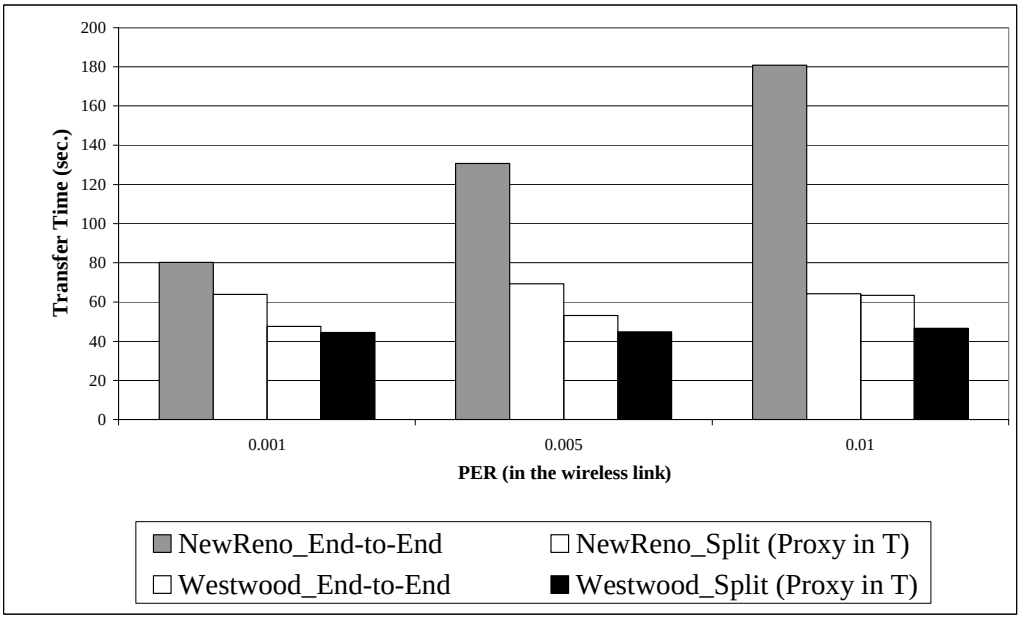


Figure 4.3. Time to transmit a 5 MByte file from $W$ to $G$

Figure 4.4 compares the average throughput achieved for different combinations of transport protocol, eventual utilization of a splitting proxy on node $T$, and various PER in links connecting $W$ and $G$. For each configuration, 20 simulations have been

run and the number of bytes sent in 230 s has been averaged. Again, TCPW coupled with a proxy achieve the best results, with an average throughput that reaches 923.53 kbit/s (with a PER of 1.0%) to 961.18 kbit/s (with a PER of 0.1%). The ability of the splitting mechanism to hide the frequent errors on the shortest wireless link from the rest of the connection is confirmed: once chosen a transport protocol, the average throughput achieved remains almost constant, independently on the PER.

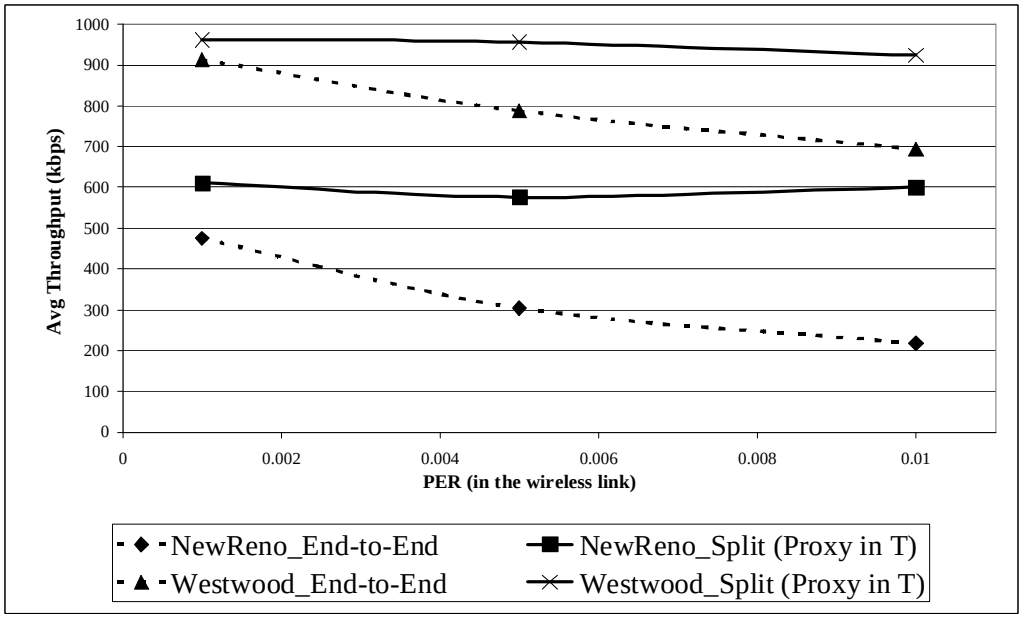


Figure 4.4. Average throughput over a 230 sec transmission from $W$ to $G$

For sake of completeness, a set of simulations which considers the reverse data flow, from $G$ to $W$, has also been performed. Also in this case TCPW and splitting improve performance. The average times required to transmit a 5 MBytes file from $G$ to $W$ are shown in Figure 4.5, while figure 4.6 illustrates the average throughput achieved on 230 s of simulation along the same path.

The advantage in using the splitting technique, exploiting one or more proxies along the path, derives from the capability of coping with wireless characteristics. Moreover, since proxies along the path generate ACKs faster than a $\sim$ 500ms (of RTT) far receiver, the *cwnd* at the sender can increase faster and thus speeding up the transmission rate. Figure 4.7 shows the sending window of the last TCP connection along the path, considering diverse configurations. Specifically, the graphs

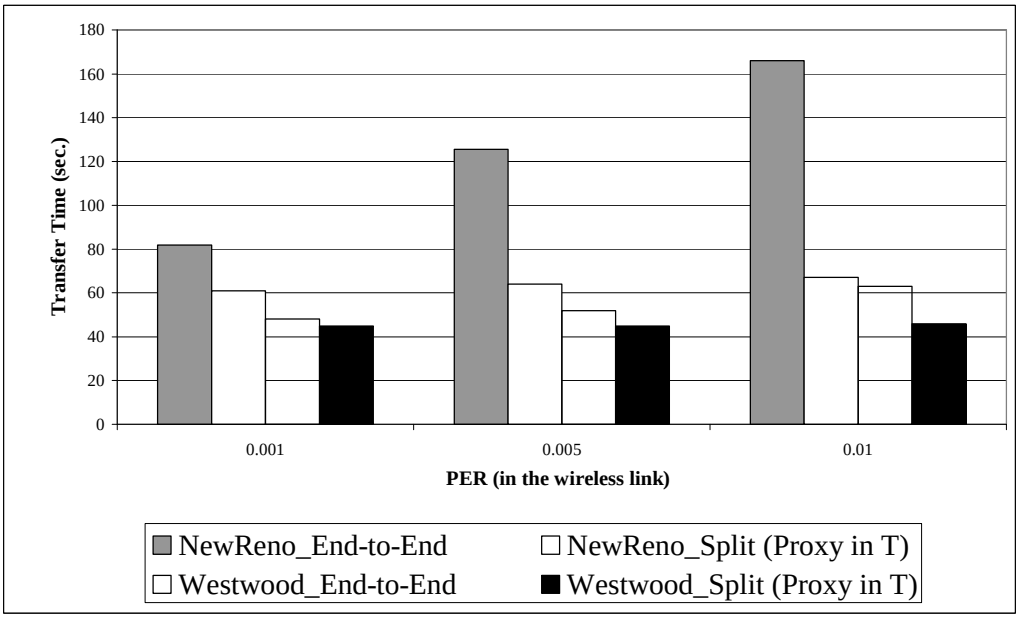


Figure 4.5. Time to transmit a 5 MByte file from $G$ to $W$

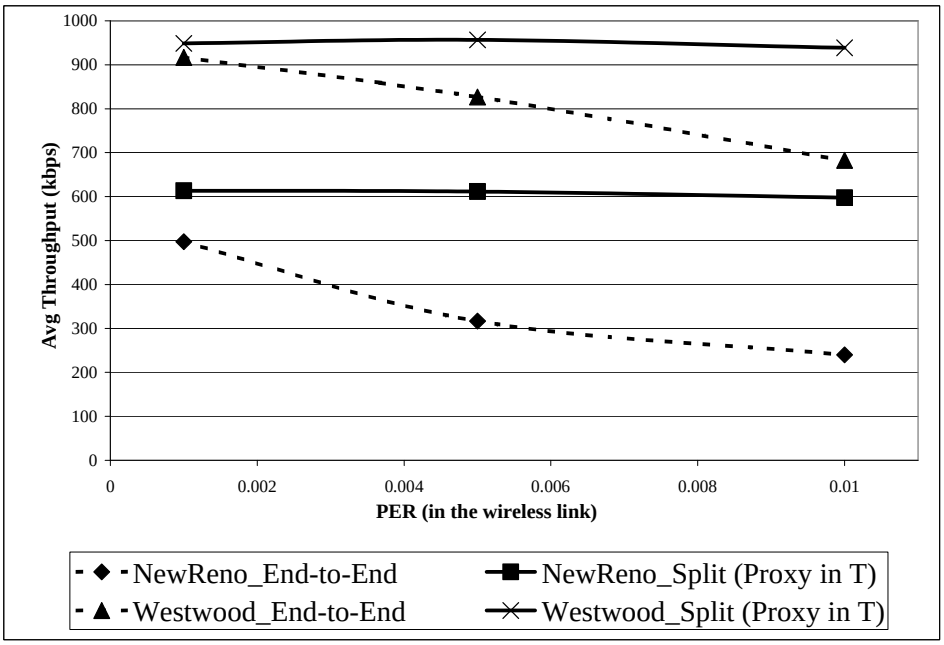


Figure 4.6. Average throughput over a 230 s transmission from $G$ to $W$

refers to a TCPNR connection with a 1.0% PER on the links between $W$ and $G$.

Then, the pure end-to-end case has been compared with those involving the utilization of one or two proxies located as depicted in Section 4.1.2. In every case, to observe the perceived instantaneous throughput at the receiver, the sending window of that TCP connection which has one of the two end nodes in $G$ has been computed. With pure end-to-end connection the TCP between $W$ and $G$ has been considered, while including one proxy the connection between $T$ and $G$ has been taken into account and, finally, with two splits the sending values between $S$ and $G$ has been measured. In case of a pure and performance as a consequence. The sending window is allowed to increase only for a small period before a wireless loss halves its value. This behavior, continuously repeated, causes a conspicuous under-utilization of the available bandwidth. If the connection is splinted into two parts, instead, the high error rate between $W$ and $T$ is hidden to the rest of the links, thus resulting in higher sending rate for longer periods. Wireless losses, in fact, are rapidly recovered in the very short wireless link between $W$ and $T$, and $T$ can rapidly store enough packets to efficiently utilize the long, but still quite reliable, links between $T$ and $G$. In case of double splitting of the connection, the presence of the proxy even on the satellite S abbreviates the length of the packet-ACK cycle between $T$ and $G$ subdividing it into two cycles: the first one between $T$ and $S$, and the second one connecting $S$ to $G$.

The utilization of shorter connections accelerates the sending window increase thus creating a pipelining phenomena that increases the perceived final throughput. The final result, compared with TCP Westwood performance, is presented in Figure 4.8, where it is reported the time required to download from $W$ to $G$ a file of 5 MBytes.

TCPNR takes great benefit from the capability of proxies present in the path in dealing with wireless losses. Deleterious shrinkage of the sending window is avoided when not necessary, thus reaching a very high utilization of the channel (about 91% of the capacity in case of double split) and equaling TCPW performance.

Assuming to utilize the proxy technology to increase performance, it becomes fundamental to optimize cache dimensioning. In Figure 4.9 and Figure 4.10, a configuration with a PER of 1% between $W$ and $T$ is considered and the average throughput for TCPW and TCPNR both in cases with pure end-to-end connection and utilizing a proxy on $T$ is compared.

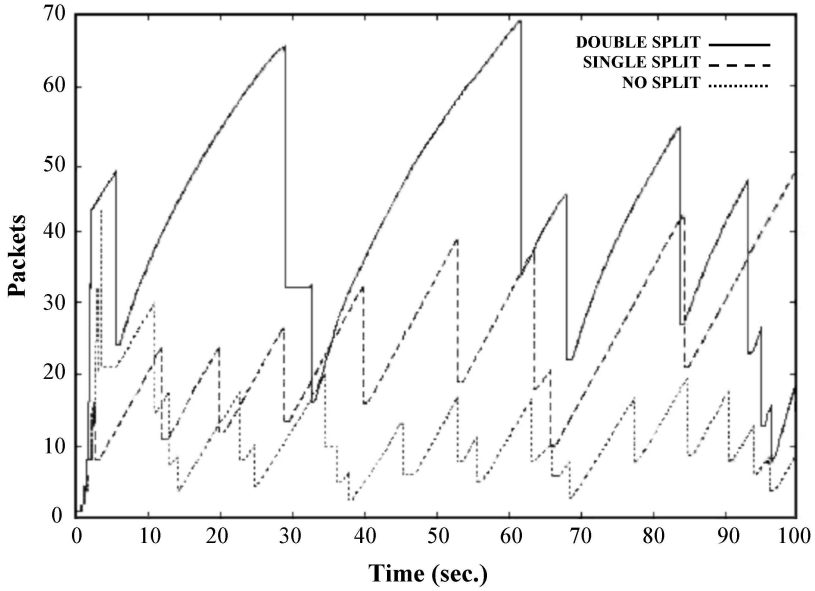


Figure 4.7. Sending window at the last sending/forwarding hop with no split, single split or double split

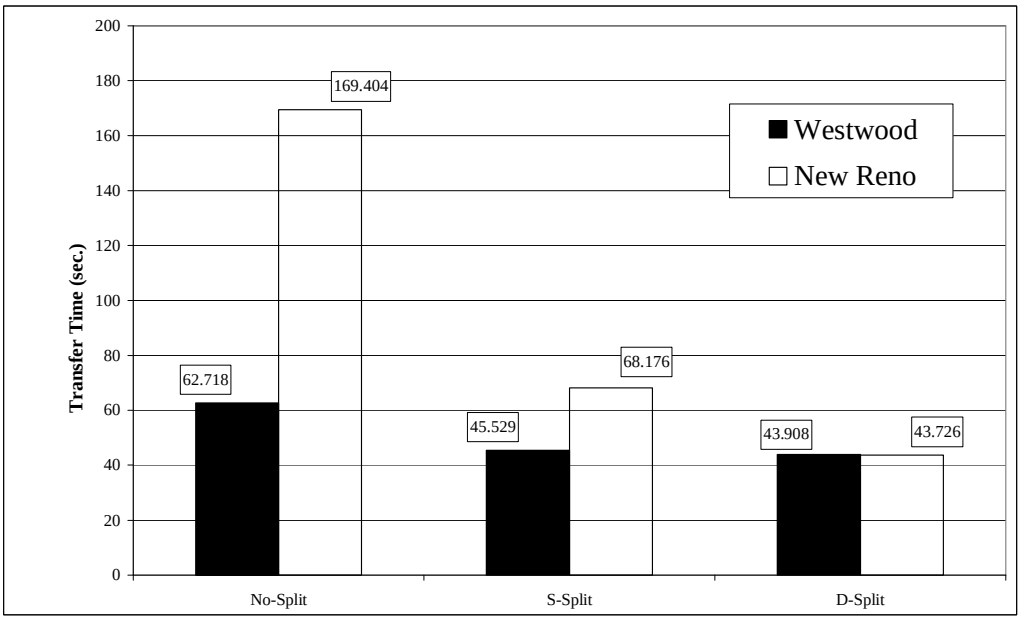


Figure 4.8. Average time to transmit a 5 MBytes file from $W$ to $G$ with no split, single split or double split

An improper amount of packets storable in the proxy causes a sensible reduction of the average throughput. All the configurations analyzed in our simulations

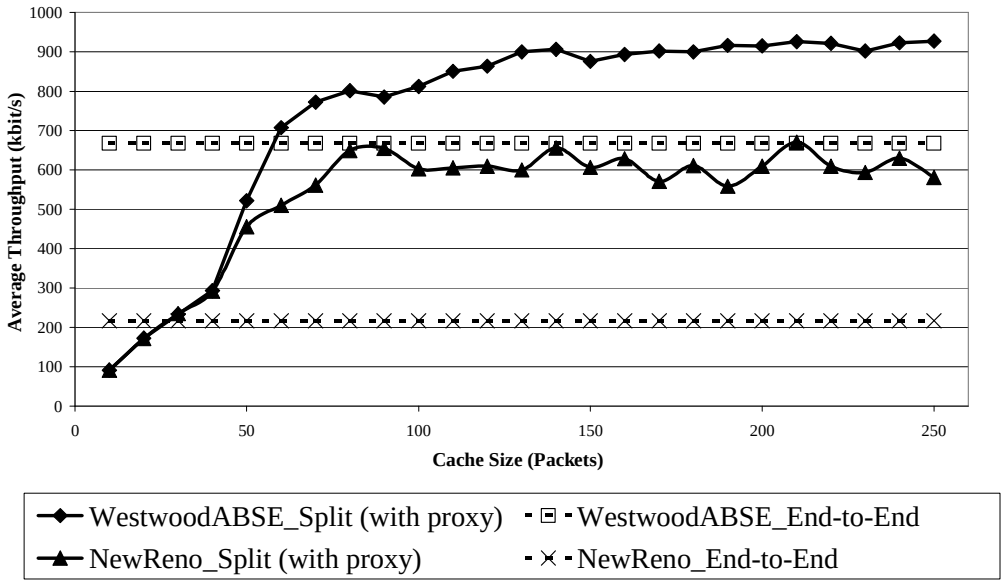


Figure 4.9. Performance achieved per proxy cache size. Transmissions from $W$ to $G$, with a single proxy on $T$

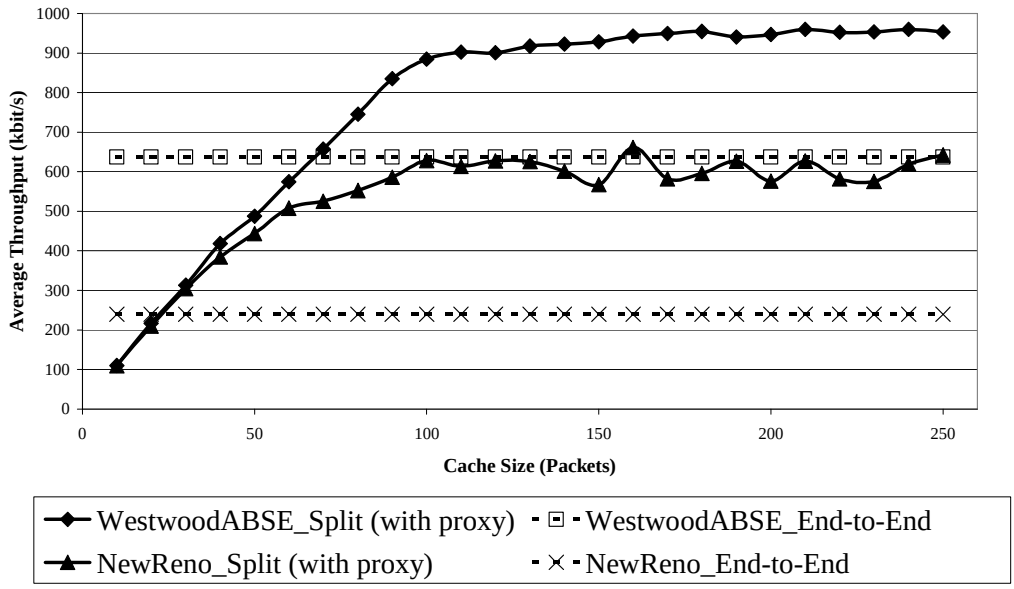


Figure 4.10. Performance achieved per proxy cache size. Transmissions from $G$ to $W$, with a single proxy on $T$

experience a steep performance degradation if the cache size is set to a value lower than the double of the channel capacity (84 packets). Under this threshold, in fact, since the advertised window sent back by the proxy corresponds to half of the free

memory in cache, TCP wastes a considerable amount of available bandwidth. Indeed, with an undersized cache capacity, the proxy may not have stored any data to transmit. The *cwnd* could be large enough to permit further transmissions but still remain unexploited since the undersized cache has not made possible an adequate pre-storing and the proxy has exhausted the few packets stored. Actually, twice of the pipe size should be considered just a lower bound for setting the dimensions of the cache while larger values clearly helps in having even better utilization of the available bandwidth. Utilizing a large memory for pre-storing packets at a proxy helps reliable links in being unaffected by transient burst error situations that temporarily slow down packets forwarding on adjacent error-prone edges. For completeness, the graphs contain also the average throughput achieved by TCPW and TCPNR in case of end-to-end connection without proxy utilization; these values are obviously independent from the proxy cache size and thus remain constant.

## 4.2 Performance evaluation of ML-IPsec over satellite networks

To achieve good performance is satellite links, TCP Performance Enhancing Proxy mechanisms are often used. In principle, a TCP PEP mechanism accelerates TCP transfers requiring access to TCP headers in intermediate nodes. As a drawback, this conflicts with IPsec [109], which requires end to end semantic be respected. ML-IPsec [110] has been identified as a suitable trade-off solution which can mitigate such a conflict.

### 4.2.1 Conflict between IPsec and TCP PEP

When TCP PEP actions involve only the two end hosts, or involve the intermediate routers but do not require access to TCP data encapsulated in the IP packet, no particular incompatibility arises. In contrast, if splitting or some PEP [71] or network address translator (NAT) options are implemented, as in most of the commercial systems for wideband services, the end-to-end semantic is violated. In particular, TCP PEP performing splitting are installed at the two end of a satellite link, and

operate (at least) on two state information stored in the TCP headers:

- TCP flow identification;
- TCP sequence number.

The former consists of two pairs of values, IP source/destination address and source/destination ports, that are used to know TCP session for each TCP packet, while TCP sequence number allows to match acknowledgements with the corresponding TCP segment. As an example, a TCP PEP agent installed at the satellite gateways could inspect all incoming TCP segments in order to generate premature acknowledgements for TCP sender that will perceive a shorten latency. These techniques conflict with end-to-end security scheme defined by IPsec in both ESP [112] and AH [111] formats. In fact:

- *Encapsulating Security Payload (ESP)* encrypts each IP datagram, including TCP header and thus encryption prevents TCP PEP from seeing and modifying any field of TCP headers;
- *Authentication Header (AH)* does not encrypt IP payload, and then leaves TCP header visible. Nevertheless, the strong authentication process reject segments in which TCP PEP modifies header fields.

### 4.2.2 Possible solutions

To overcome these problems, a number of approaches have been proposed, such as replacing IPsec with a transport-layer security mechanism, tunneling one security protocol within another, using transport-friendly ESP format, and splitting IPsec into as many segments as the whole path is splinted, but each has its own limitations. In [110] all these approaches are described and their limitations are identified. Moreover, an approach based on a multi-layer security protection scheme for IPsec is proposed (ML-IPsec).

#### 4.2.2.1 ML-IPsec

A smarter solution to perform network security over links, including TCP PEP agents installed in intermediate nodes, is based on a IPsec extension called Multi-Layer IPsec (ML-IPsec) [110]. The MLIPsec principle is to divide IP datagram into different zones and apply different protection schemes at each zone. A protection scheme is then composed of a set of security associations (SAs), a set of private keys and access control rules. Therefore, one or more authorized/trusted intermediate nodes can decrypt, modify and re-encrypt a certain part of the incoming segments according to what established in the security associations. An ML-IPsec security protection scheme well suited to satellite networks involving TCP PEP agents installed on both satellite terminal and gateway to accelerate TCP over satellite channel can be structured as follows (Figure 4.11).

- IP datagram payload is divided into two zones:
  - TCP header ($21^{st}$ to $40^{th}$ octet) constitutes the zone 1;
  - TCP data portion ($41^{st}$ to above octet) constitutes the zone 2.
- An end-to-end protection scheme is used for TCP data portion indicated as zone 2: encryption key shares only between end-systems (source and destination).
- A separate protection scheme is used for TCP header indicated as zone 1: encryption key shared among source, destination and one or more trusted intermediate nodes (i.e., satellite gateway and satellite terminal).

### 4.2.3 Performance evaluation

The simulation scenario, implemented through ns-2 [30], reproduces an heterogeneous link compliant to that one represented in Figure 4.11, envisaging networks with both different physical characteristics and different level of trust. In particular, the following three sub-links can be identified:

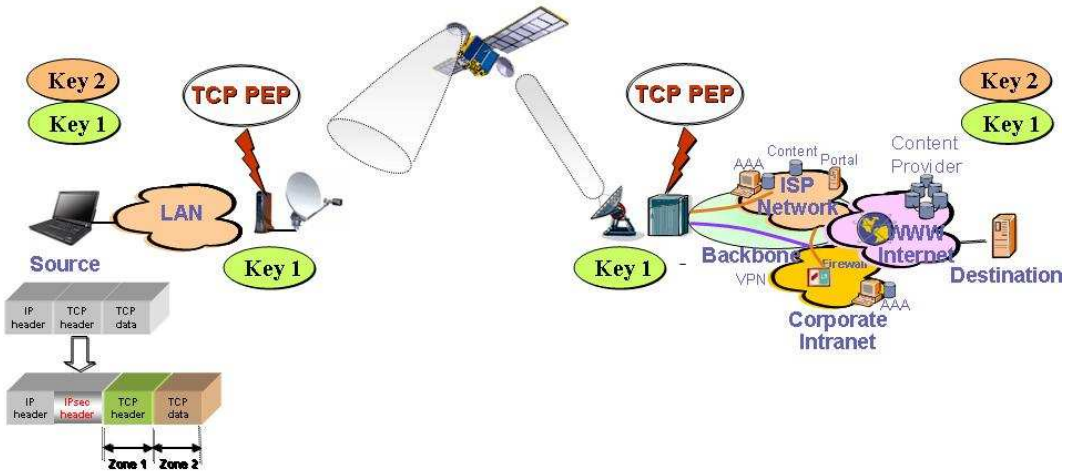


Figure 4.11. ML-IPsec in a TCP PEP satellite scenario

- *Sub-link 1.* It connects TCP sender to the satellite gateway and reproduces the local access network of the source system. Its physical parameters are: bandwidth = 10 Mbit/s, delay = 1 ms.

- *Sub-link 2.* It is a link connecting two satellite nodes: a satellite terminal (ST) and a satellite gateway (GW). For the sake of simplicity, hereafter we will refer to both as satellite gateway (SAT GW). Its physical parameters are: bandwidth = 2 Mbit/s, delay = 260 ms. It is the bottleneck of the overall link and PER is chosen in a range of values (from $10^{-4}$ to $10^{-2}$).

- *Sub-link 3.* It represents the insecure public network (Internet) that connects the SAT GW to a remote TCP receiver. Its physical parameters are: bandwidth = 10 Mbit/s, delay = 1 ms.

At the SAT GWs, TCP PEP modules can be arbitrary enabled or disabled. Optionally, also on board the satellite a TCP PEP module can be enabled to further improve TCP performance as demonstrated in [113]. In particular, the Satellite TCP Performance Enhancing Proxy (SaTPEP) is considered. Mainly, SaTPEP performs connection splitting [114]. Then, in case TCP PEP is on, the end-to-end TCP connection is splinted into three (or four if TCP PEP on satellite is on) sub-connections. Furthermore, in the satellite link an enhanced TCP, adapting

sender/receiver TCP buffer and the initial TCP window to the bandwidth-delay product, is used. The presented analysis evaluates end-to-end performance in terms of both goodput and file transfer time, enabling TCP PEP agents for the following security schemes:

- *IPsec protection provided "hop by hop'.* Every sub-link establishes an own security association. As a drawback, this scheme requires "trusted" SAT GWs (the security level depends on competence and fairness of the network administrator).
- *An end-to-end protection scheme based on ML-IPsec.* SAT GWs access only TCP header fields. Data payload is protected until reaching the receiver.

Then, the question on the basis of the following analysis is: "which is the best trade-off between performance and security?".

#### 4.2.3.1 IPsec performance over TCP PEP

Once that the improvement deriving from the use of TCP PEPs over satellite links is demonstrated, the most suited security scheme must be defined. In the following analysis, confidentiality for IP datagram is assumed as a basic security requirement. Then, all the considered schemes are based on the ESP protocol. In particular, a "hop-by-hop" protection scheme, in which ESP protection is applied at each sub-link separately, is compared to ML-IPsec security architecture, where SAT GWs have the rights to decrypt, manipulate and re-encrypt TCP headers in order to perform TCP PEP operations. In both cases, transport and tunnel modes have been taken into account [109]. ML-IPsec allows to protect end-to-end IP datagram payload, while some extra overhead may need to support the management of two cryptographic keys. On the other hand, by performing IPsec separately in each sub-link, SAT GWs must be assumed absolutely trusted. If a hacker is able to get behind the network administrator control, the communication confidentiality is completely lost. Then, considering the values of packet length overhead reported in [110], Figure 4.12 shows performance achieved in the different security schemes in terms of goodput while Figure 4.13 shows the time needed to transfer a 5 Mbytes file. In Figure 4.12, goodput measurements basically reflects the percentage of overhead associated to

each security scheme (IP payload = 1460 bytes): ML-IPsec in tunnel model (ml esp tu) presents the higher overhead by leading to a goodput of 1834 kbit/s, while the higher goodput occurs when no protection is performed (ip). In general, tunnel mode introduces more overhead then transport mode as well as ML-IPsec requires more overhead than IPsec.

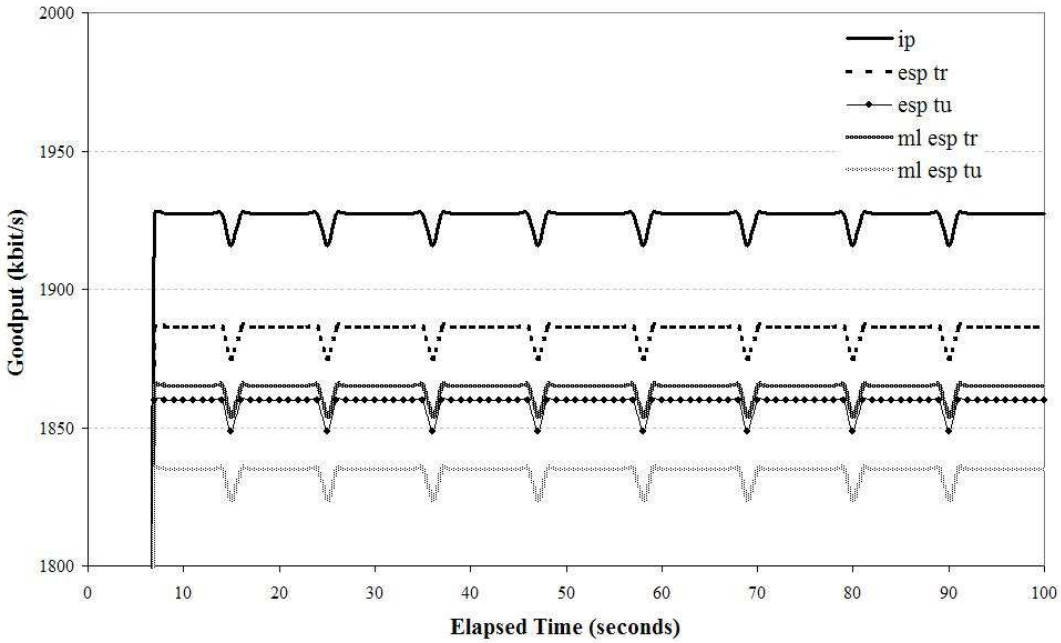


Figure 4.12. Goodput measurements under different IPsec schemes (Packet size 1500 bytes)

Figure 4.13 focuses on measuring time needed to transfer 5 Mbytes of data produced by an application (i.e. FTP) by considering two different packet sizes: 1500 and 284 bytes. If the transfer time is not particular affected in case of large packets (low overhead percentage), it results significantly different from a security scheme to another in case of short packets (high overhead percentage).

An important aspect analyzed in the simulation campaign has been the measurement of the transfer time increase, with respect the case in which protection is not applied (IP), for growing values of PER in the satellite segment. In particular, Figure 4.14 is concerned to a transfer of a 10 Mbytes file by FTP protocol when TCP modules are enabled in the SAT GWs and on satellite and PER over satellite link ranges from $10^{-4}$ to $10^{-2}$. Outcomes show as the growth of the transfer time

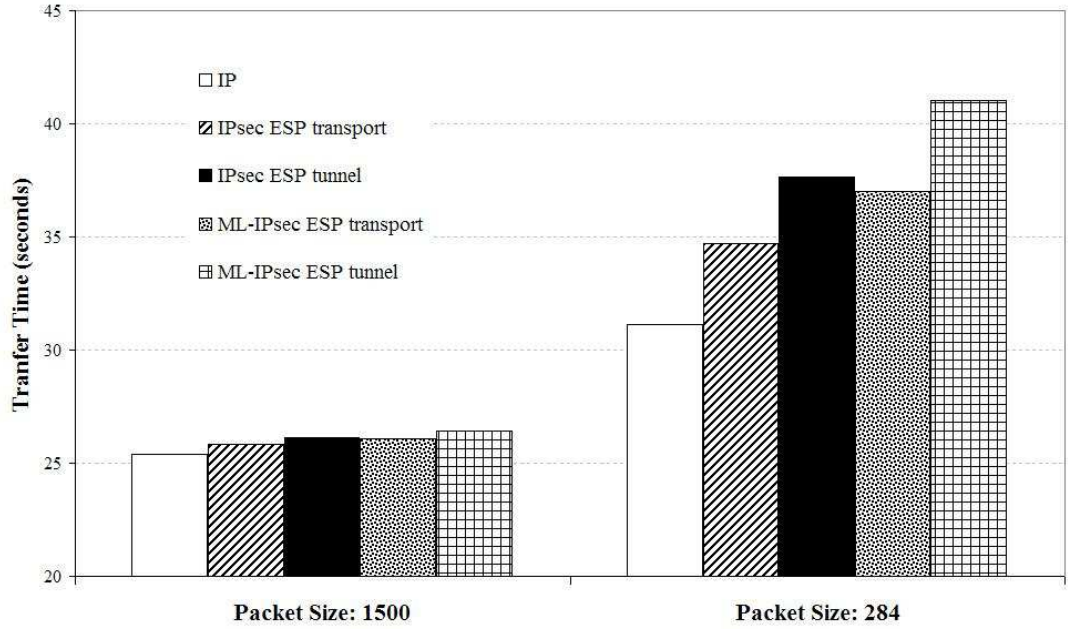


Figure 4.13. Transfer Time of a 5 Mbytes file under different IPsec schemes

increase is not linear and, in addition, the performance gap between different protection schemes is not constant, varying PER value. Definitively, the influence of the overhead on end-to-end performance is more relevant for higher PER values.

## 4.3 A TCP-driven CAC scheme for a HAPS-Satellite Integrated Architecture

In a system architecture including a HAP segment interconnected to a satellite segment, a TCP driven CAC scheme based on a cross-layer interaction has been designed and its performance evaluated. Assuming that radio resources are allocated according to TCP window evolution, the idea is to use TCP statistics as a further input for the CAC algorithm, in order to consider also the effect of the TCP reactive congestion control mechanism in the CAC preventive decisions. The main goal is to maximize the overall network utilization while preserving the target QoS.

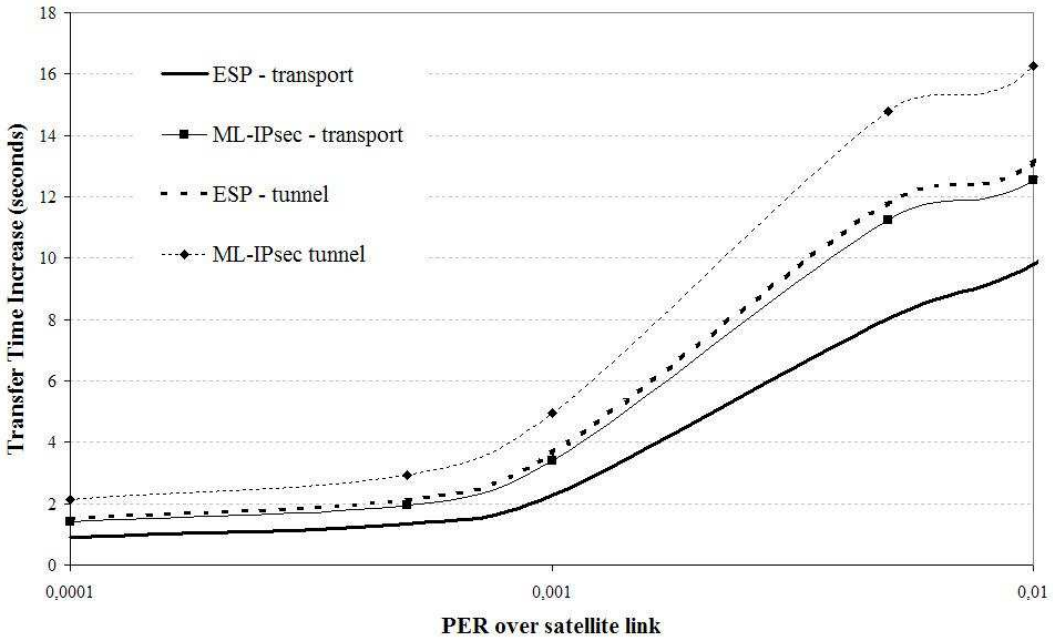


Figure 4.14. Transfer Time Increase vs. PER over satellite links

### 4.3.1 System architecture

The reference scenario concerns the forward channel of an integrated HAP-Satellite architecture to provide wireless access to users. The HAP actually operates as the network access point; the data streams originating in the core network and addressed to the users under the HAP footprint are transmitted through a satellite gateway towards a GEO Satellite which is connected with the HAP which will deliver it to the user terminals.

To evaluate performance of the proposed TCP-driven CAC algorithm under a wide range of network conditions, the users of the integrated HAP-satellite system are classified according to three criteria:

1. *Bandwidth requirements* - the users are divided into three QoS-classes (traffic-groups) with nominal data rate of 128, 256 and 512 kbit/s respectively.

2. *Transport layer protocol type* - the users are distinguished in TCP and non-TCP users.

3. *Mobility* - Users are classified as fixed, nomadic and mobile. Fixed users are equipped with VSATs properly mounted in order to provide line-of-sight towards the HAP. Nomadic users are equipped with portable terminals easy to move but working in stationary conditions. Mobile users, equipped with small terminals, are further classified into mobile-urban (mob_urb), mobile-suburban (mob_sub) and mobile-highway (mob_high) as a function of the specific kind of environment they are moving through.

The HAP footprint is assumed to be single-cell, since in the considered scenario resources are primarily limited by the satellite segment and thus the introduction of HAP multibeam coverage to allow multi-cellular architecture is meaningless [88].

#### 4.3.1.1 "Basic" CAC scheme

The algorithm proposed in [89] supports the provision of telecommunication services to different traffic classes. In particular, network administrator can define the desirable (target) ratio between the blocking probabilities of each pair of supported QoS-classes and the CAC aims at meeting these preferred values by making a corresponding sharing of the network bandwidth among the QoS-classes. In brief, the algorithm divides the aggregate bandwidth of the network into a number of segments equal to the number of the supported QoS classes, i.e. each QoS-class is assigned a certain percentage of the aggregate bandwidth. A new connection is admitted only if the bandwidth assigned to the QoS-class of the connection is sufficient, regardless the overall availability of resources in the network. In terms of CAC theory, the identified CAC algorithm fits exactly to considered scenario. It regards the channel aggregate data rate as the network decisive resource, while it identifies the user QoS requirements with their class nominal data rate.

### 4.3.2 CAC-TCP interaction

Since TCP suffers from several impairments in case of leaky links with a high delay the actual transmission rate is reduced (for example when losses are detected) and in addition the TCP "reaction time" is slowed down by the long perceived latency. In the considered scenario the satellite segment introduces about 560 milliseconds in

the overall RTT, while the three links may be affected by transmission errors depending on propagation conditions [82]. This behavior can entail a severe performance limitation in terms of actual average bit rate. In this context, if radio resources are allocated taking into account TCP window evolution, in order to optimize the utilization of shared radio resources without compromising QoS requirements, the implementation of a CAC schemes that takes decisions on the basis of the constrains risen at the transport layer is recommended. In fact, if CAC decision is based just on static parameters as, for example, the "nominal rate" associated to each terminal according to its QoS class, two situations can occur:

- Accepted TCP connections never reach their nominal rate;
- Connections are refused even if capacity is available.

To avoid this possible inefficiency, a cross-layer interaction between transport layer (where TCP runs) and data link layer (where CAC runs) is proposed. The basic idea is to monitor all the active connections through a TCP proxy in order to have continuously information about TCP dynamics and assign capacity to each link accordingly. Then, samples of the current data rate are computed for each connection at regular intervals and passed as input to the CAC algorithm. CAC, in turn, uses such information for two scopes:

- Estimates the unused capacity;
- Monitors the maximum achievable rate for each terminal class and then uses this parameter (to replace the "nominal rate") in the decision process to accept or not a new connection.

In order to both minimize the signaling overhead and to make the cross-layer inter-operability viable, the TCP proxy is supposed to be located on the HAP, in parallel to the CAC.

#### 4.3.2.1 CAC-TCP simulation model

The efficiency of the proposed algorithm, based on the utilization of information concerning TCP window evolution as input to CAC, is evaluated in comparison with

performance of the basic CAC scheme in terms of Blocking Probability and Average Throughput. Both the TCP driven and the basic CAC scheme are simulated through the off-line combination of two different simulation tools that run sequentially. At first, in order to acquire the TCP feedback, the Network Simulator ns-2 (release 2.27) [30] is executed by configuring sender and receiver nodes, running TCP NewReno [55] as transport protocol and FTP as application protocol. The ns-2 simulations provide the TCP statistics of a user characterized by the channel model of the mobility-group which he belongs to and by the nominal data rate of its QoS-class. The output file contains samples, taken at regular time intervals (10 seconds), of the actually required data rate of the user.

Additionally, a C++ simulation tool capable of emulating the general network environment and utilizing the TCP measurements has been developed. The C++ simulator presents threefold functionality:

**a.** It implements the QoS-based CAC algorithm presented in section 4.3.1.1 to take the admittance/rejection decision upon every new applying connection. In the case of the basic CAC scheme this decision is based uniquely on the data rate value that has been declared by each connection initially as its minimum bandwidth requirement. The algorithm checks the availability of resources in the network after calculating the occupancy of the network capacity based on the nominal data rates of all the active connections. In the case of the TCP-driven CAC, the aggregate data rate is calculated using the actually required data rate of each connection on the basis of congestion window status, which is provided as feedback from the transport layer;

**b.** It calculates the instantaneous and the average throughput of the network;

**c.** It computes the call blocking probability of each QoS-class separately as well as the aggregate call blocking probability of the network.

Functionalities (b) and (c) are common for both the basic and the TCP-driven CAC scheme. In the latter case, the decision of the CAC algorithm relies upon the instantaneous value of the TCP data rate of the active connections at the moment of the new user connection request. However, this could lead to a highly unstable

system, since the TCP congestion window can change abruptly during its evolution . In fact, if a request for admittance coincides with a temporary reduction of the needed data rate of some of the already active users, the new connection is accepted. Nevertheless, when the cause of degradation is removed and the quality of communication is gradually restored, the increase of the throughput in the presence of the extra traffic due to the new user shall result in congestion. The reverse phenomenon can occur due to the temporary improvement of the channel conditions; in particular temporary high data rates would lead to the deceptive conclusion of unavailability of resources and thus to the unjustified exclusion of any candidate user at that period. In order to avoid such inconvenience, a longer term study of the TCP throughput of each connection would be required so that the statistical processing of the TCP output would provide more solid and reliable info to the CAC algorithm. Specifically, the input to the TCP-driven CAC algorithm is a sequence of average data rate samples that asymptotically converge to the average data rate of the connection.

However, although the prediction is much more solid by averaging the sampling sequence, the chance that the network capacity be exceeded due to variations of the TCP congestion window value around the ns-2 sample cannot be excluded. Therefore, the proposed TCP-driven CAC scheme adopts a more pessimistic approach, considering the connection data rate equal to the TCP sample plus a safety margin. This margin is evaluated as a percentage of the difference between the nominal data rate of the connection and the current TCP sample provided by the ns-2:

$$margin = p \cdot (nominal\ rate - sample\ rate) \tag{4.1}$$

Assuming that the data rate of each connection is higher than the feedback from transport layer, the available resources of the network are underestimated; practically some resources are reserved for the case of excessive increase of the traffic load beyond the measured TCP performance. The value for the parameter $p$ is a tradeoff between avoiding congestion and maximizing the throughput of the network. As a matter of fact, an unstable communication environment requires a high $p$ value, while constant reception conditions guarantee the accuracy of predictions and thus low $p$ value can be used aiming to utilize the resources as efficiently as possible. Definitively, the choice of $p$ may not be static; therefore the CAC algorithm can

monitor the level of data in the output buffers and change the $p$ value accordingly, i.e., high size of data waiting to be forwarded means inability to follow the abrupt changes of the wireless channel and thus lower $p$ value should be used. Of course if $p = 1.0$, then the new algorithm falls to the Basic CAC scheme. Finally, light bottleneck phenomena can be handled properly utilizing adequate buffering and scheduling methods.

### 4.3.3 Channel model implementation

To evaluate the performance of the proposed CAC scheme, packet error distributions (derived at TCP level) suitable for HAPS-terminal communications have been provided as simulation input to ns-2. As a first realistic assumption, the Satellite-HAPS and the Gateway-Satellite links are considered "Quasi Error Free" (QEF) since both links are in LOS. Definitively, in terms of error rate the end-to-end communication path can be modeled as user terminal-HAPS link behavior; the adopted channel models are referred uniquely to the HAP-User link. In general, it is widely accepted that some of the channel models developed for satellite environment [84] apply to scenarios in which HAP performs communication with ground stations. Specifically, two approaches have been followed, depending on the mobility of the users and on the corresponding characteristics of the terminals:

1. In the case of fixed terminals (including portable), LOS conditions are assumed. This scenario is well modeled by means of a Rice distribution taking into account the effects of the multipath [84];

2. As far as mobile terminals are concerned, according to the ITU-R recommendation [90], three different scenarios (urban, suburban and highway) and a two-state channel model [91] have been identified to characterize the alternating line-of-sight and shadowing condition. Finally, taking as reference the probabilities of the durations of each state reported in [90] for an elevation angle of 21° and considering probabilities of state transitions based on a approximation of the Land Mobile Satellite (LMS) channel, a suitable two-state error model has been implemented for simulations.

In the ns-2 simulations statistical packet loss distributions have been adopted considering that a loss corresponds to a TCP segment received corrupted and thus to an error perceived at the transport layer. For fixed and portable terminals uniform loss distributions are used, with mean values equal to $10^{-4}$ and $10^{-3}$ respectively, while for mobile terminals a two-state Markov model has been adopted. The "bad" state has an average duration consistent with values provided in [90] and corresponds to all TCP packets dropped (PER = 1). On the other hand, during the "good" state, a uniform loss distribution with a low average PER is considered (PER $\approx$ 0); as a result, the overall average PER can be approximated with the average duration of the "bad" conditions. Moreover, although not strictly necessary as simulation parameters, the average PER is calculated for each mobility group to acquire a metric of the channel quality. These values, along with the average PER of the fixed and the nomadic users, are summarized in Table 2.

| **Terminal characteristics** | **Average PER** |
|---|---|
| Fixed | $10^{-4}$ |
| Portable | $10^{-3}$ |
| Mobile (urban environment) | $4 \cdot 10^{-2}$ |
| Mobile (suburban environment) | $2 \cdot 10^{-2}$ |
| Mobile (highway environment) | $10^{-2}$ |

Table 4.1. User terminal classes in terms of environment and average PER

Finally, it is worth mentioning that to guarantee the randomness of the channel error distributions among different runs of the ns-2, the initializing variable "seed" of the ns-2 random number generator is changed before every new run. In this way, two different users always present different dynamics even if they belong to the same QoS-class and mobility-group.

### 4.3.4 Results

Both the arrival of new admittance requests and the termination of the active connections are considered to follow Poisson distribution [89]. Thus, the time between two successive arrivals of users (inter-arrival time, denoted as $\tau$) and the duration of each admitted connection (denoted as $d$) follow exponential distribution, with mean

value $\frac{1}{\lambda}$ and $\frac{1}{\mu}$ correspondingly:

$$\begin{cases} pdf(\tau) = \lambda \cdot e^{-\lambda \cdot \tau}, & E[\tau] = \frac{1}{\lambda} \\ pdf(d) = \mu \cdot e^{-\mu \cdot d}, & E[d] = \frac{1}{\mu} \end{cases} \tag{4.2}$$

The parameters E[$\tau$] and E[d] along with the aggregate number of users in the network, denoted as $S$, determine the traffic load of the network. $L$ is defined as the average traffic load that could be forwarded if the capacity of the integrated HAP-Satellite system were infinite. Having assumed uniform distribution of the users among the three different QoS-classes, $L$ (in kbit/s) is given by the equation:

$$L = \left(\frac{1}{3} \cdot S\right) \cdot \frac{E[d]}{E[\tau]} \cdot 128 + \left(\frac{1}{3} \cdot S\right) \cdot \frac{E[d]}{E[\tau]} \cdot 256 + \left(\frac{1}{3} \cdot S\right) \cdot \frac{E[d]}{E[\tau]} \cdot 512 \tag{4.3}$$

Considering that E[d] is equal to 500 s (corresponding to 32 MBytes files transferred at a nominal rate of 512 kbit/s), and S is equal to 52, $L$ is calculated as a function of E[$\tau$]d. The values of the above traffic parameters figured out through simulation are summarized in Table 4.2.

| **E[$\tau$] (s)** | **L (kbit/s)** |
|---|---|
| 862.815 | 9000 |
| 834.982 | 9300 |
| 808.889 | 9600 |
| 784.377 | 9900 |
| 761.307 | 10200 |
| 739.556 | 10500 |
| 719.012 | 10800 |
| 699.58 | 11100 |
| 681.17 | 11400 |
| 663.704 | 11700 |
| 647.111 | 12000 |
| 631.328 | 12300 |

Table 4.2. Expected value of the Inter-arrival time and Average traffic load calculation

Performance of the identified metrics (Blocking Probability and utilization of the network resources in terms of average throughput) have been evaluated from four different points of view:

1. traffic load (L);
2. average PER;
3. blocking probability ratio among the three QoS-classes ($BPR_1$ and $BPR_2$)
4. percentage of TCP users ($TCP_{perc}$).

#### 4.3.4.1 Impact of the traffic load

Figures 4.15 and 4.16 present the behavior of the system for a wide range of traffic load. The simulation of the different traffic conditions is achieved as a function of $L$. Uniform distribution of the users among the mobility-groups as well as equal Blocking Probability of the three QoS-classes ($BPR_1 = BPR_2 = 1$) have been considered and all the users are assumed to use TCP as transport protocol ($TCP_{perc}$ = 100). Results presented in Figures 4.15 and 4.16 show that the exploitation of the TCP feedback can lead to a great improvement in the network performance for the whole range of $p$ values. However, as it is expected, the higher is $p$, the lower is the gain in Blocking Probability and in Average Throughput. Nevertheless, even for $p = 0.6$ (which means very pessimistic approach), the knowledge of the TCP performance allows the CAC algorithm to decrease the Blocking Probability of 22% (L = 12300 kbit/s) and to increase the Average Throughput of 10%, in comparison with the basic scheme.

#### 4.3.4.2 Impact of the average PER

Different mobility groups are characterized by different channel model experiencing different PER. Since higher PER implies greater decrease of the TCP average throughput and thus greater divergence from the performance of the basic CAC scheme, the user distribution among the different mobility groups shall play an important role in the system performance. In fact, since

$$\begin{cases} E[PER\{fixed\}] < E[PER\{portable\}] < E[PER\{mob_{highway}\}] < \\ < E[PER\{mob_{suburban}\}] < E[PER\{mob_{urban}\}] \end{cases} \quad (4.4)$$

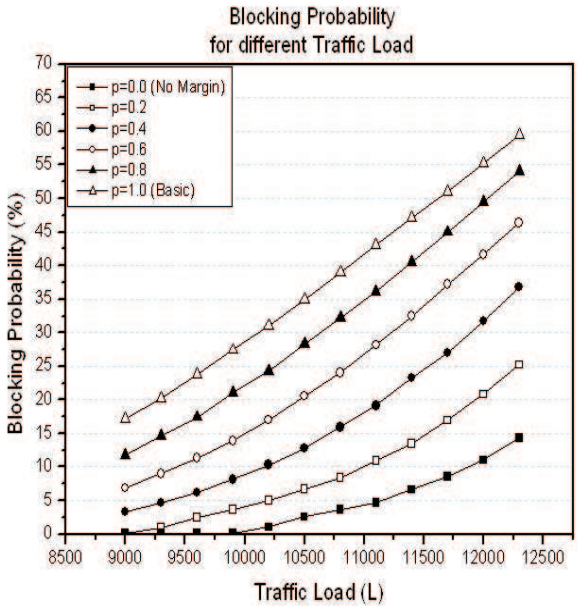


Figure 4.15. Behavior of the system in terms of blocking probability for different traffic conditions ($L$), uniform distribution of the users among the mobility-groups, equal Blocking Probability ($BPR_1 = BPR_2 = 1$), $TCP_{perc} = 100$

higher is the percentage of mobile users (especially in suburban and urban environments), more relevant is the contribution of the TCP feedback to improve system efficiency. Therefore, in order to control the distribution of the users among the five mobility groups, a new parameter, denoted as $mob_{ratio}$, is defined as:

$$\begin{cases} mob_{ratio} = \frac{perc\{mob_{urban}\}}{perc\{mob_{suburban}\}} = \frac{perc\{mob_{suburban}\}}{perc\{mob_{highway}\}} = \\ = \frac{perc\{mob_{highway}\}}{perc\{mob_{portable}\}} = \frac{perc\{mob_{portable}\}}{perc\{mob_{fixed}\}} \end{cases} \tag{4.5}$$

where *perc* denotes the percentage of each mobility-group in $S$. Moreover, $mob_{ratio}$ is related to the average PER ($av\_PER$) of the network:

$$av\_PER = \sum_{k \in M} perc\{k\} \cdot E[PER\{k\}] \tag{4.6}$$

where $M$ is the set of the five mobility-groups. On the basis of equations (4.5) and (4.6), as well as on the values of average PER for each mobility-group (Table 4.1), Table 4.3 includes the $av\_PER$ for a variety of $mob_{ratio}$ values.

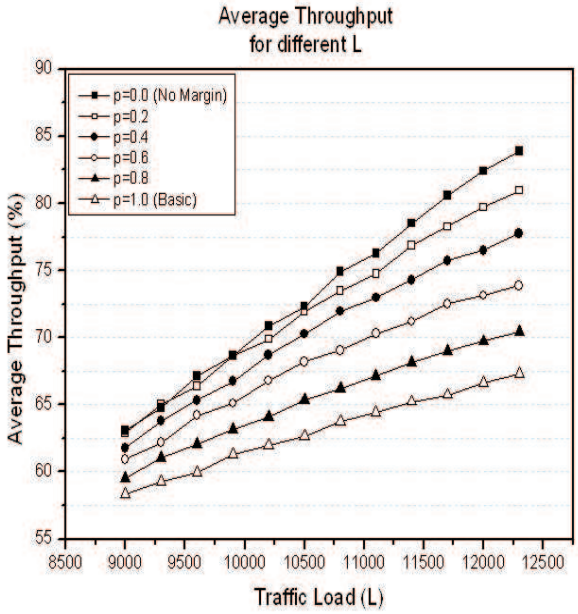


Figure 4.16. Behavior of the system in terms of Average Throughput for different traffic conditions ($L$), uniform distribution of the users among the mobility-groups, equal Blocking Probability ($BPR_1 = BPR_2 = 1$), $TCP_{perc} = 100$

| $mob_{ratio}$ | $Av\_PER$ |
|---|---|
| 0.5 | 0.00418065 |
| 0.7 | 0.0079925 |
| 0.9 | 0.0121912 |
| 1 | 0.01422 |
| 1.1 | 0.0161314 |
| 1.3 | 0.0195159 |
| 1.5 | 0.0223014 |

Table 4.3. Average PER as function of the $mob_{ratio}$ parameter

Figures 4.17 and 4.18 present respectively the Blocking Probability decrease and the Average Throughput increase that are achieved implementing the CAC-TCP interaction, for $0.5 \leq mob_{ratio} \leq 1.5$, $TCP_{perc} = 100$ and $BPR_1 = BPR_2 = 1$, while the traffic intensity $L$ is considered equal to 10800 kbit/s. The results of the simulations show that higher is the average PER (higher $mob_{ratio}$) in the overall network (the harsher are the reception conditions) more important is to take into

account the TCP feedback in the CAC procedure.

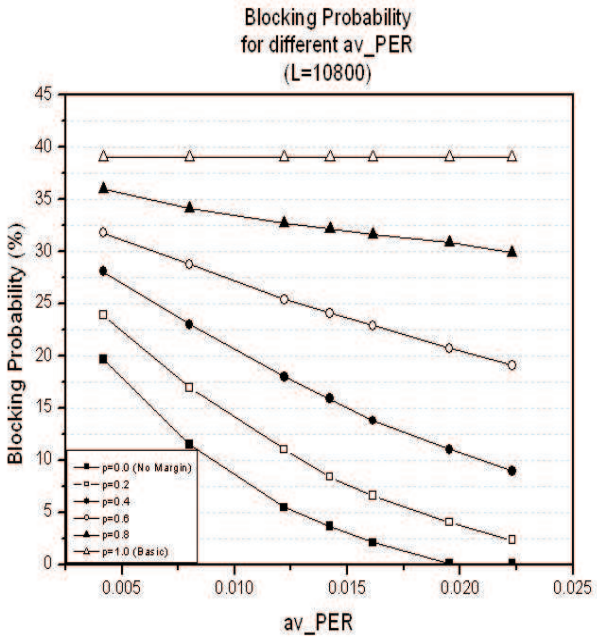


Figure 4.17. Blocking Probability decrease achieved by the CAC-TCP interaction, for $0.5 \leq mob_{ratio} \leq 1.5$. $TCP_{perc} = 100$ and $BPR_1 = BPR_2 = 1$, traffic intensity equal to L = 10800 kbit/s

#### 4.3.4.3 Impact of Blocking Probability Ratio among the QoS-classes

Higher is the nominal data rate of a connection, higher is the degradation, in terms of unused capacity, that the connection experiences by considering a specific PER. Thus, assigning larger portions of the network capacity to the QoS-classes with higher data rate requirements, decreasing their blocking probability, the redundancy of assigned bandwidth increases. Consequently, the implementation of the proposed scheme maximizes the revenue for networks serving high data rate users. The conclusions are evident in Figures 4.19 and 4.20 where the system performance is evaluated for $0.5 \leq BPR_1 = BPR_2 \leq 1.5$, $TCP_{perc} = 100$, $mob_{ratio} = 1$, and $L =$ 10800 kbit/s.

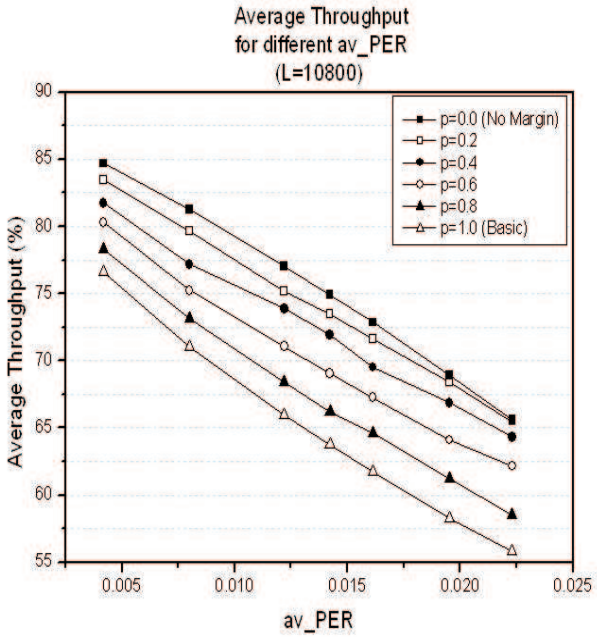


Figure 4.18. Average Throughput increase achieved by the CAC-TCP interaction, for $0.5 \leq mob_{ratio} \leq 1.5$. $TCP_{perc} = 100$ and $BPR_1 = BPR_2 = 1$, traffic intensity equal to L = 10800 kbit/s

#### 4.3.4.4 Impact of the percentage of TCP users

Since the CAC-TCP interaction is applied only under the hypothesis of TCP based communication, the impact of the percentage of the network occupancy by TCP users has been investigated. Figure 4.21 presents the Average Throughput of the system, for different of $TCP_{perc}$ values, $BPR_1 = BPR_2 = 1$, $mob_{ratio} = 1$, $L = 10800$ and $L = 11700$ kbit/s.

## 4.4 Dynamic resource allocation based on a TCP-MAC cross-layer approach

Satellite networks pose special design challenges especially due to the high propagation delays, the impact of adverse weather conditions (rain for frequencies beyond 10 GHz), shadowing in case of mobility, and the management of time-varying traffic loads. Dynamic network management and related control algorithms are therefore

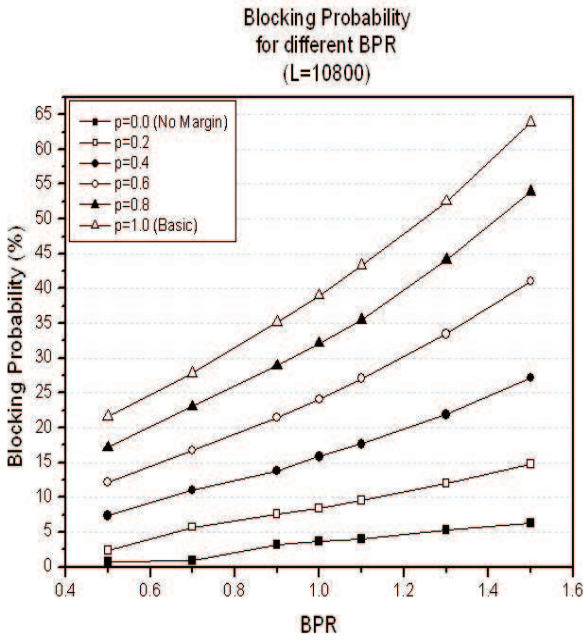


Figure 4.19. System performance in terms of Blocking Probability for $0.5 \leq BPR_1 = BPR_2 \leq 1.5$, $TCP_{perc} = 100$, $mob_{ratio} = 1$, and $L = 10800$ kbit/s

necessary for the support of data transmissions with Quality of Service (QoS) guarantees. In this frame, one of the most important features of this DVB-RCS standard is the implementation of DAMA strategies aiming at a fair and efficient use of the shared resources in the return channel via satellite.

DAMA scheme introduces a new control loop, based on a signaling exchange between RCSTs and the NCC. As a consequence, a new component is added to the overall end-to-end delay: the "access delay" (see for more details Section 1.5.3.2).

TCP behavior is based on the RTT experienced in the network. Therefore, working of TCP over DVB-RCS is governed by two nested control loops both depending on RTT. The two nested control loops can interact in a way that degrades performance. Both theoretical and simulation analysis are provided in Section 3.8.3.

To achieve good TCP performance ensuring optimum capacity utilization, a novel Bandwidth on Demand (BoD) scheme is proposed. It is centrally controlled by the NCC, that performs an optimized resource allocation by means of a cross-layer interaction between Medium Access Control (MAC) and TCP taking into account TCP dynamics. This scheme allows the co-existence with other allocation

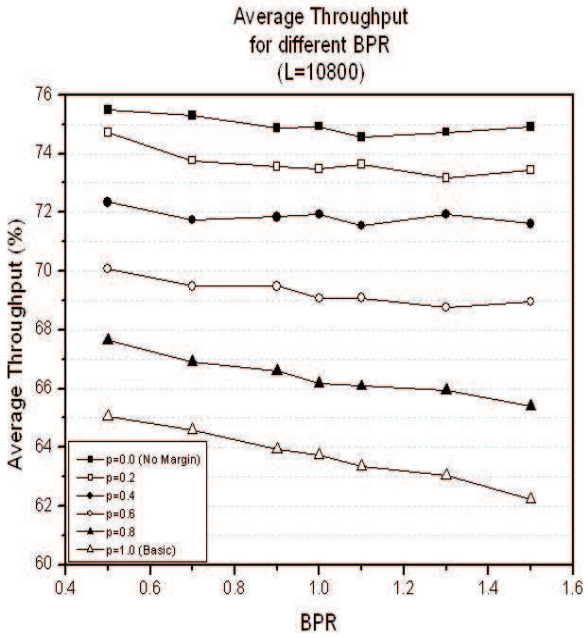


Figure 4.20. System performance in terms of Average Throughput for $0.5 \leq BPR_1 = BPR_2 \leq 1.5$, $TCP_{perc} = 100$, $mob_{ratio} = 1$, and $L = 10800$ kbit/s

schemes (e.g. the proposed cross-layer BoD scheme is employed for all low-priority TCP-based traffic flows, while other policies are used to manage real-time traffic).

### 4.4.1 A novel TCP-driven dynamic resource allocation scheme

The implementation of a dynamic access scheme allows to optimize resource sharing. The DVB-RCS standard defines the following set of capacity request methods: CRA, RBDC, VBDC, AVBDC, and FCA. In particular, VBDC performs a capacity requests as long as new data arrive in the RCST queue maximizing the capacity utilization. The amount of capacity per frame, a generic RCST requests at the $k^{th}$ super-frame, can be expressed by using the formula defined in [31]:

$$r_{VBDC}(k) = \left\lceil \frac{Q(k) - n_s \cdot a(k) - n_s \cdot \sum_{j=1}^{L-1} R(k-L+j) - n_s \cdot w(k)}{n_s} \right\rceil^{+} \qquad (4.7)$$

where:

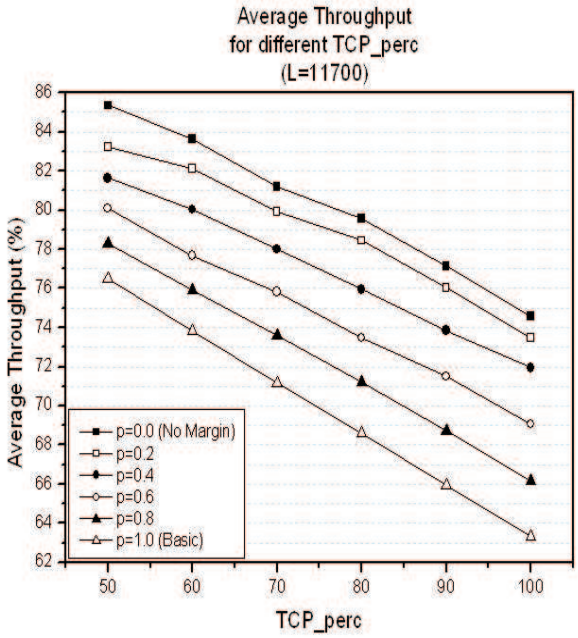


Figure 4.21. Average Throughput vs. Percentage of TCP connections for different of $TCP_{perc}$ values, $BPR_1 = BPR_2 = 1$, $mob_{ratio} = 1$, $L = 11700$ kbit/s

- $Q(k)$ represents the amount of data stored in the ST queue at the beginning of the $k^{th}$ resource allocation period;
- $a(k)$ is the amount of resources per frame assigned to the ST for the current superframe;
- $\sum_{j=1}^{L-1} R(k-L+j)$ is the sum of all the previous requests sent but not yet served;
- $w(k)$ indicates the owed resources by the DAMA controller to the STs, because in one or more of the previous allocation periods, the controller assigned fewer or ever none of the requested resources.
- $n_s$ is the number of frames in a superframe.

Unfortunately, the VBDC allocation method leads to a huge increase in the RTT perceived by the systems. In fact, the above mentioned access delay involves in this case the following contributions (Figure 4.22):

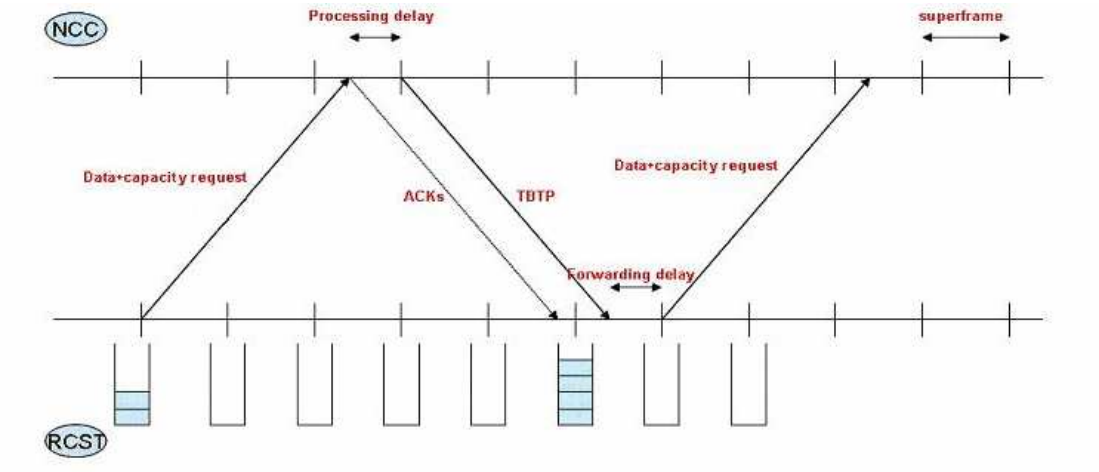


Figure 4.22. Description of access delay contributions (DAMA allocations)

- *Reservation delay* (RD)- time interval between te arrival of data in the MAC buffer and the transmission of the corresponding capacity request (sent in dedicated slots);

- *Propagation Delay* (PD) - sum of the time to propagate the capacity request from the RCST to the NCC and the time to deliver the Terminal Burst Time Plan (TBTP) in the opposite direction;

- *Processing and synchronization delay* (PSD) - time spent by the DAMA controller (in the NCC) to elaborate and transmit the TBTP message with the capacity assignment;

- *Forwarding delay* (FD) - time between the reception of the TBTP by the RCST and the actual transmission of data.

On the basis of the above delay contributions, the RTT values corresponding to the VBDC case can be of the order of 1.6 s [92].

The DVB-RCS standard also supports RBDC capacity request method. In this case, resources are allocated on the basis of the rate at which RCST wishes to transmit (usually based on monitoring the arrival rate at its layer 2 queue). This method reduces the access delay at the expenses of reduced network efficiency (see

Section 3.8.3. Most RCS systems provide a wide range of Bandwidth on Demand (BoD) schemes based on a combination of both methods (VBDC and RBDC).

In this context, the aim of the novel allocation scheme is to reduce the access delay, keeping optimal network efficiency, by using TCP status information to predict the amount of data that will feed the RCST queue in the future. In order to exchange cross-layer signaling between layer 2 and layer 4, dedicated local messages are generated each time that TCP parameters (e.g., *cwnd*) go beyond a certain threshold; this is according to an explicit cross-layer method [93].

Assuming that system response time (as defined in 3.8.3) is greater than the physical RTT [1], the proposed algorithm aims to predict the further data that will be injected in the RCST queue when the resources will be allocated, according to both the amount of data transmitted at the $k^{th}$ superframe and the TCP phase (SS or CA):

$$Q^{'}(k) = \begin{cases} 2 \cdot n_s \cdot a(k) & Slow\ Start \\ n_s \cdot a(k) \cdot \left(1 + \frac{1}{cwnd}\right) & Congestion Avoidance \end{cases} \tag{4.8}$$

Therefore, a new term is added to (4.7) and then, in each super-frame the amount of resources per frame requested at the $k^{th}$ superframe, is:

$$R_{VBDC}(k) = \left\lceil \frac{Q(k) - n_s \cdot a(k) - n_s \cdot \sum_{j=1}^{L-1} R(k-L+j) - n_s \cdot w(k)}{n_s} + \frac{Q^{'}(k)}{n_s} \right\rceil^{+} \tag{4.9}$$

Finally, in addition to R(k), also the TCP phases will be communicated by the RCST to the NCC by setting a flag (*TCP phase flag*) in the capacity request message:

- 1 → Slow Start;
- 0 → Congestion Avoidance;

On the other side, the NCC serves all incoming requests by considering two priority levels: a *High priority level* associated to requests with the TCP phase

[1] Such assumption results largely verified in all current DVB-RCS systems, especially if PEP mechanisms running at the satellite gateway end TCP connections within the satellite segment.

flag set to 1, and a *Low priority level* associated to requests with the TCP phase flag set to 0. In this way, short transfers are privileged as well as the just started connections. In each queue (i.e., the queue for requests in the SS phase and the other for requests in the CA phase), requests are satisfied according to Maximum Legal Increment (MLI) algorithm [94] in order to guarantee a fair allocation among the different competing flows. If the amount of needed resources exceeds those available in a superframe, the NCC creates a "waiting list" to assign the resources in the next superframes and stops the *cwnd* growth of all the connections coming from the RCSTs that have not obtained the requested resources. The proposed allocation scheme at the:

- assures that resources are fairly shared among all the active TCP connections;
- performs a further cross-layer action setting a new variable, named *cwnd**, in order to modify the current *cwnd* value used by the TCP source in the RCST ($cwnd \leftarrow cwnd^*$). The NCC (acting like a PEP) sends back the *cwnd** value using a field for TCP options (layer 4 ACKs) in the headers. The rationale of this modification on the TCP protocols is to avoid congestion at the RCST side and, then, the possibility of buffer overflows.

The main expected effects of the proposed cross-layer-based access scheme are:

- *Reduction of the access delay* - since the request algorithm predicts also the amount of data that will feed the RCST queue due to the TCP congestion control mechanism, the access delay will be reduced of an RTT;
- *Avoiding congestions at the RCSTs* - the cross-layer interaction between RRM and TCP layers permits to prevent layer 2 buffer overflows due to satellite network congestion;
- *Efficient and dynamic resource allocation* - resources are dynamically assigned on a super-frame basis according to explicit requests, thus allowing a better utilization of the available capacity.

### 4.4.2 Analysis of the allocation process

A simulator has been implemented using ns-2 (release 2.27) [30], in order to evaluate the performance of the cross-layer allocation process and the resulting performance. The ns-2 extensions that reproduce a traditional GEO satellite network have been modified to simulate a centralized Multi Frequency - Time Division Multiple Access (MF-TDMA) scheme and the NCC functionalities. The interaction between the TCP *cwnd* trend and the corresponding allocation process has been analyzed by means of the average resources assigned (in slots) as a function of time; such parameter has been monitored for one or more TCP connections sharing the return link of a communication network compliant to DVB-RCS architecture. The main simulation parameters are detailed in Table 4.4.

| **Physical Parameters** | |
|---|---|
| Physical RTT | $\sim$ 515 ms |
| Return link bandwidth | 2048 kbit/s |
| Maximum number of RCSTs | 32 |
| **Frame Parameters** | |
| Superframe duration | 96 ms |
| Number of slot per frame | 32 |
| **Protocols** | |
| Transport Protocol | TCP NewReno |
| Application Protocol | FTP |
| **TCP Parameters** | |
| TCP packet size | 1500 bytes |
| PER | Variable, from 0 to 0.0001 |

Table 4.4. Main simulation parameter values

Considering a file transfer from an RCST to the NCC, Figure 4.23 highlights how the allocated resources (continuous line) are strictly correlated to the cwnd trend (dotted line) with our scheme. Three different phases can be recognized in the allocation process:

1. An initial exponential growth corresponding to the TCP SS phase;
2. A reduction of the allocated resources (approximately one half) when the Fast Recovery mechanism is invoked as reaction to the detection of a loss;

3. A linear growth corresponding to the TCP CA phase.

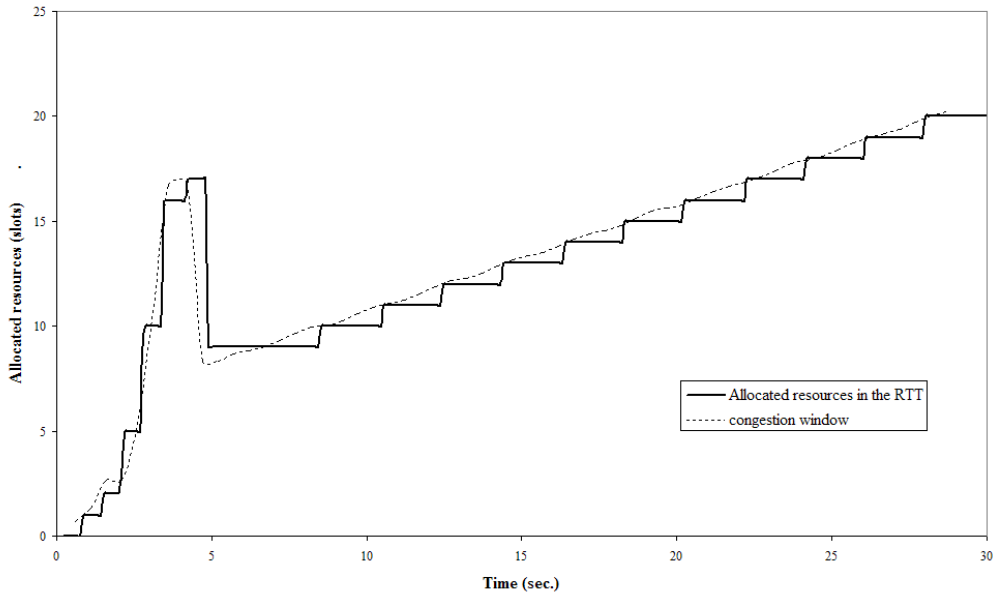


Figure 4.23. Comparison among allocated resources and cwnd trend versus time (1 TCP connection, PER = $10^{-4}$)

Figure 4.24 shows fair resource sharing between two TCP connections, when losses occur. At the beginning, the capacity is saturated (i.e., the NCC stops the *cwnd* growth of both the connections in order to prevent congestion and losses), the capacity is perfectly divided between the two connections. When a connection is affected by a transmission error (loss), with consequent *cwnd* reduction, the NCC re-assigns temporally the unused capacity to the other connection in order to optimize the utilization of resources.

### 4.4.3 Performance evaluation

Since TCP performance strictly depends on the perceived latency at the end-systems, RTT represents the main parameter to evaluate the TCP performance. Hence, the proposed TCP-driven RRM scheme has been compared to the classical CRA and VBDC capacity allocation techniques [11]. The main simulation parameters are those provided in Table 4.4.

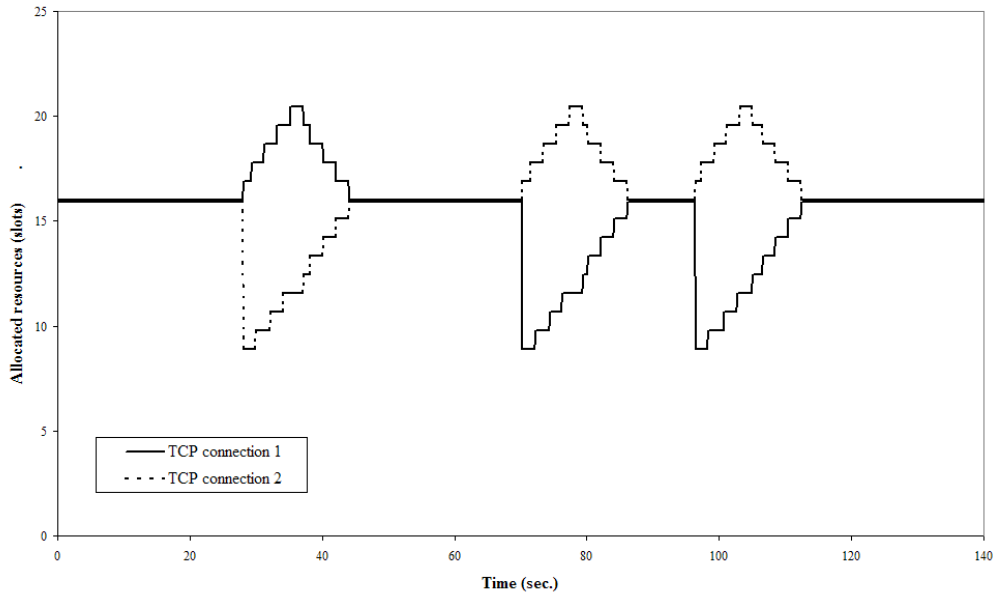


Figure 4.24. Comparison among allocated resources in the RTT vs. time (2 TCP connections, PER = $10^{-4}$)

Figure 4.25 shows the average perceived RTT for the three considered access schemes.

The obtained results allow the following considerations:

- VBDC presents the higher delay equivalent to almost three times the physical RTD [92];
  - 1 RTT for the capacity request and notification exchange,
  - 1 RTT for the TCP segment and ACK exchange,
  - an additional delay due to all the processing tasks at both RCST and NCC sides (comparable to a RTT).
- In the CRA case, RTT is only affected by the propagation delay, since the capacity is not negotiated, but permanently assigned in the set-up phase of a connection;
- The proposed TCP-driven scheme (simply called "cross-layer scheme") reduces the overall VBDC delay by almost 1 physical RTT, trying to predict the amount of data that will feed the RCST queue.

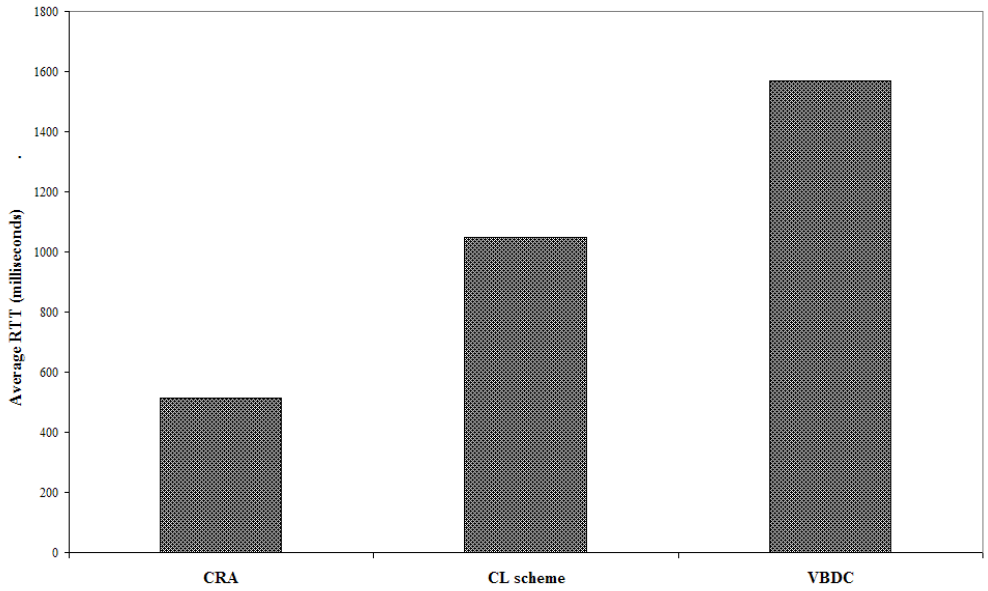


Figure 4.25. Comparison among average RTT values obtained with the following techniques: VBDC, CRA and the cross-layer scheme

Then, evaluating only the end-to-end performance in terms of RTT for a single TCP connection, the proposed cross-layer technique represents a good trade-off solution between VBDC and CRA.

The principle of assigning capacity on the basis of the real needs of the data sources leads to significant improvements in terms of both end-to-end performance and network utilization when multiple TCP connections compete for the overall capacity. The following simulations have been carried out considering: 20 FTP transfers coming from different RCSTs and with start time instants spaced of 5 s; 10 Mbytes files have to be uploaded to a remote system. As a reference, a fixed allocation scheme (i.e., CRA) is considered where the capacity is equally divided at the beginning among the RCSTs in a static manner. The average file transfer time has been measured for different PER values and then compared with the mean transfer time of the proposed cross-layer scheme. The results, shown in Figure 4.26, highlight that the TCP-driven RRM scheme with cross-layer information allows a transfer time reduction ranging from 12.3% (PER = 0) up to 26.5% (PER = 0.01).

Finally, Figure 4.27 highlights the benefits deriving from the use of the proposed cross-layer scheme with respect to CRA in terms of channel utilization. In fact,

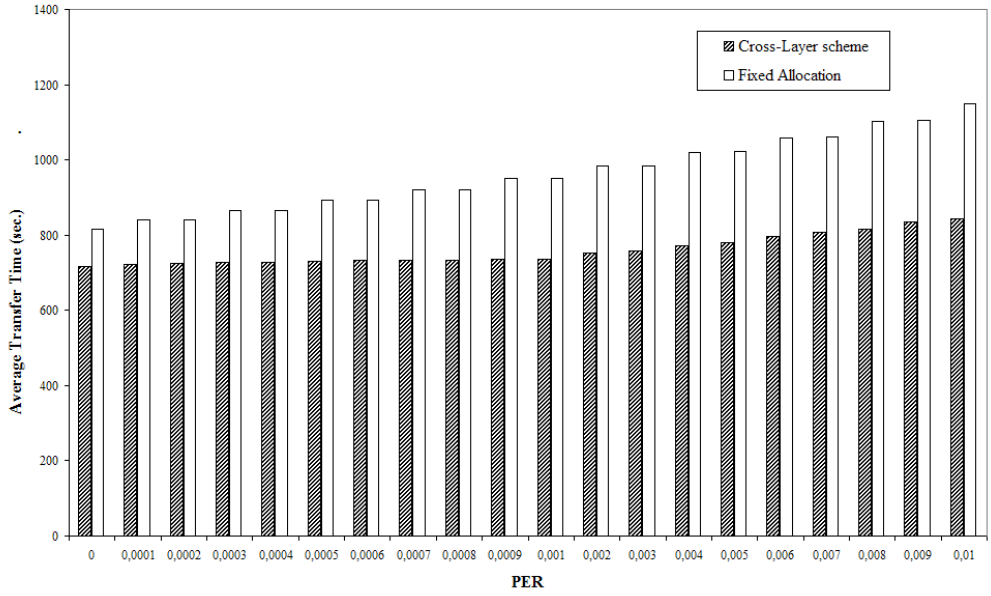


Figure 4.26. Average file transfer time versus PER (20 FTP transfers starting at instants spaced of 5 s)

the continuous line indicates the percentage of the average utilization increase, for the cross-layer scheme with respect to CRA, when 5 FTP transfers (10 Mbytes) are running at instants spaced of 5 s with PER = $10^{-3}$. This figure also shows the curve representing the instantaneous channel utilization when the cross-layer scheme is used, in order to show the optimum values constantly achieved.

## 4.5 A Cross-Layer based handover for TCP applications

Mobile Internet Protocol and its newer realization Hierarchical Mobile IPv6 [95][96][97] has been identified as a valid approach to provide a mobility support at the IP layer, allowing mobile nodes to change network without changing their IP address to keep communication consistency. Unfortunately, TCP/IP protocols were not designed to optimally handle mobility. Handover (HO) between links with different delays and bandwidth-delay product causes a number of problems such as the generation of segment bursts and delivery of out-of-order packets [98]. Even a good management

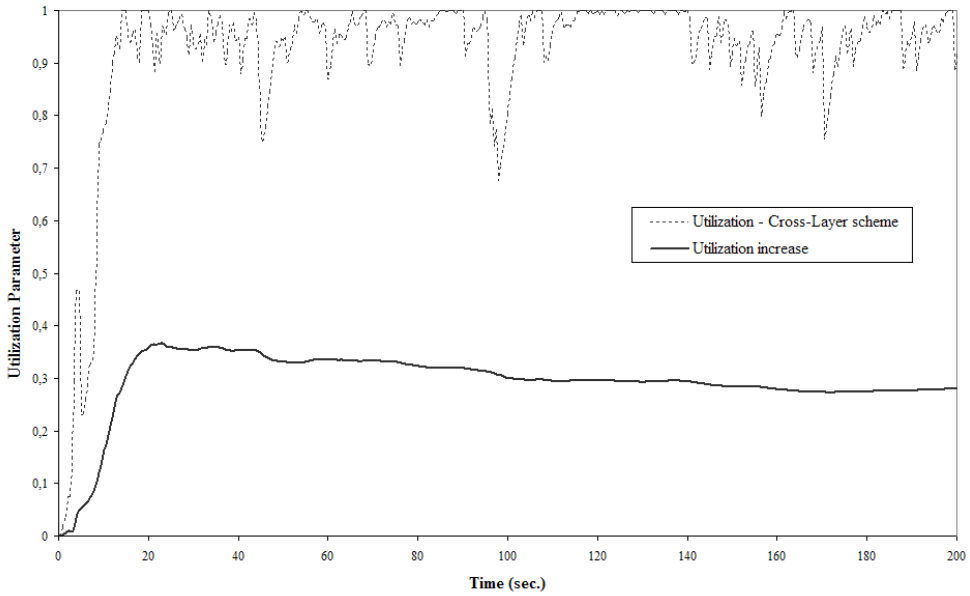


Figure 4.27. Cross-layer access scheme: utilization and percentage of the average utilization increase with respect to the CRA scheme (5 FTP transfers starting at instants spaced of 5 s, PER = $10^{-3}$).

of the HO at IP level does not prevent completely TCP from experiencing unwanted effects. In fact, both transmission errors and the sudden delay change cause harmful consequences to TCP performance:

- Losses are misinterpreted as a congestion and thus TCP congestion window is dropped to a lower value [15];
- TCP computes the retransmission timeout (RTO) on the basis of measured RTT values. Then, an unexpected increase of the RTT could lead to an undesired RTO expiration with a consequent reduction of the congestion window to a minimum value [15];
- After the HO procedure is completed, the new link is established but TCP packet flow resumes with inefficient or un-initialized parameters set.

When moving from a segment to another, even assuming that negotiation procedures to access the new channel are efficient and transparent to upper layers, a certain time $t^*$ is needed to switch channel. In this time period, no data are received

(hypothesis of Hard HO [99]). Moreover, when the HO is performed for necessity (i.e. WLAN link power decrease at the edge of the coverage area) BER increases as well as packet losses perceived by the unaware TCP. To help on this, TCP could benefit of explicit notifications of handovers. To predict a handover event and to allow TCP to perform a certain number optimization tasks, a cross-layer mechanism between transport layer (adopting TCP) and network layer (adopting Mobile IP protocol) has been conceived.

### 4.5.1 Reference scenario

The reference scenario is compliant with HMIPv6 (Hierarchical Mobile IPv6) standard [97] with the key elements shown in the block diagram in Figure 4.28.

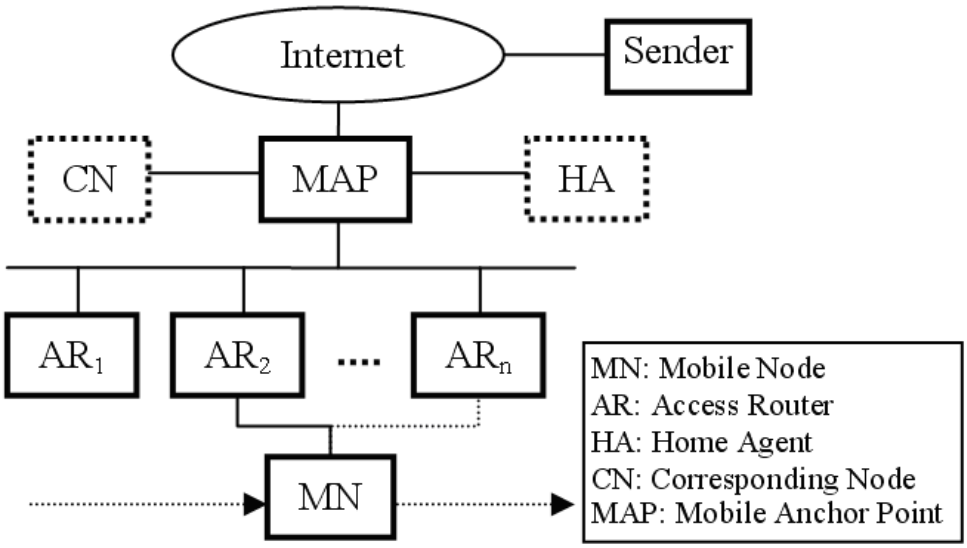


Figure 4.28. Mobile environment

The mobile node, which represents the user terminal, is moving across a hybrid network composed of one Access Router (i.e. $AR_1$) that represents the satellite access and at least one AR (i.e. $AR_2 \ldots AR_n$) which uses WLAN technology. The source of the data received by the Mobile Node (MN) is supposed a remote host connected via Internet to the mobile environment ("Sender"). All the downlink traffic coming from the Sender through different radio networks is routed via a

single entity, the Mobile Anchor Point (MAP), which redirects traffic to the proper radio network (satellite or WiFi AR) and manages handover signaling. Then, MAP is logically connected to both satellite gateway and WiFi access points management, but also to other entities in charge to perform the mobility functions (Home Agent and Corresponding Node) that will not be addressed herein.

The reference scenario is composed of different overlapping segments of an access network. The technologies considered for the connections with the end user will be either:

**a)** two way satellite standard (DVB-RCS) over a GEO link;

**b)** WLAN access via WiFi "Hot Spots" (IEEE802.11b).

In general, the moving user terminal is supposed to be able to perform transparently the hard HO operations between the access technologies ($a$) and ($b$) to obtain a seamless data channel during its movement. Only two different HO occurrences are addressed:

1. DVB-RCS $\longrightarrow$ WiFi;
2. WiFi $\longrightarrow$ DVB-RCS;

The case satellite-satellite is clearly of no practical interest, since the satellite coverage is assumed large enough with respect to user mobility needs. The case $WiFi \longrightarrow WiFi$ is out of the scope of this work. Figure 4.29 pictorially shows the reference scenario, with every possible handover occurrence between segment a) and segment b) highlighted by vertical arrows, with the user terminal (MN) under satellite coverage and performing HO whenever a WLAN connection becomes available. The speed of the MN determines the time spent into the WLAN coverage.

It is assumed that the HO procedure is fully handled by HMIP up to the network layer and that during the HO procedure a certain number of IP packets are lost. An additional no-connection time (out of service or HO time) occurs because of limited buffers in the terminals and for the transient in the setup of the Intermediate Frequency electronics and Phase Locked Loop that produces corrupted packets [100]. The out of service time is mainly due to the limits of hardware dedicated to the hard HO switch and to the HO decision delay [99].

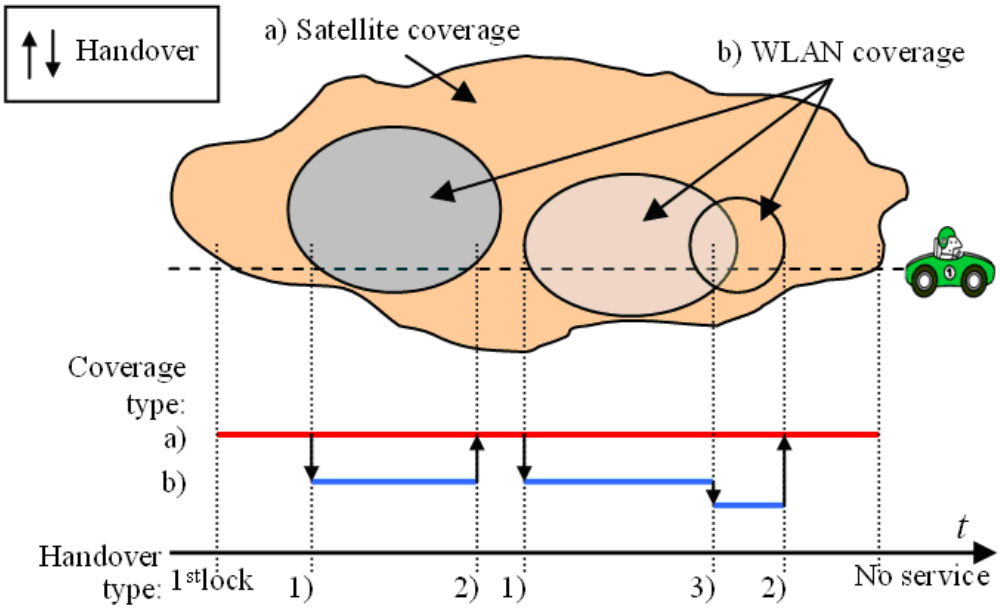


Figure 4.29. Reference scenario

### 4.5.2 TCP behavior during handovers

The main dynamics affecting TCP behavior in the considered scenario are briefly described.

#### 4.5.2.1 Transmission Errors

Handover from WLAN Hot Spot to Satellite system is triggered by a gradual degradation of the WLAN link power that increases the Packet Error Rate (PER) perceived by TCP. TCP misinterprets such losses as a congestion notification, reduces the congestion window and triggers the congestion avoidance algorithm. Then, TCP flow will resume on the satellite link with very small congestion window and increases it slowly due to the long latency. As a consequence, satellite capacity will be wasted for long time.

#### 4.5.2.2 Handover time ($t^*$) implications on RTO

The handover execution time $t^*$ is the time needed to the mobile host to switch between the two different segments. Therefore, when a handover is performed, an

unexpected deep variation of the RTT could lead TCP to unnecessarily trigger its ACK-clocked recovery mechanisms [15]. In particular, Retransmission Time-Out (RTO) is automatically updated over the time on the basis of the RTT measurements. A sudden increase of the RTT due to the handover time may lead to RTO expiration. In fact, when moving from WLAN to satellite system, the large delay difference ($Delay_{satellite} \gg Delay_{WLAN}$) increases the ACK-clock break and then the probability of the RTO expiration. On the contrary, when the handover is exploited in the opposite direction, the RTO results to be largely overestimated causing too long wait time to restart transmission in case of network congestion.

#### 4.5.2.3 Generation of segment bursts

If the delay of the new link is smaller than the old one (it is the case of handovers from satellite system to WLAN), the first ACK sent over the new link could overtake the last $k$ ACKs sent over the old one. Thus, due to TCP cumulative ACK scheme [16], a burst of $k$ packets is sent at once. Intermediate buffers could not be dimensioned to support such a burst length causing multiple losses.

#### 4.5.2.4 Bandwidth-Delay Product variation

TCP congestion window represents a sender's estimate of the bandwidth-delay product (BDP) or pipe size. Different effects follow a BDP change depending on whether BDP increases or decreases after a handover:

- moving from WLAN to the satellite system ($BDP_{new} > BDP_{old}$), satellite link capacity will be underutilized; in fact, the initial congestion window is small and its increment is slow if congestion avoidance is performed (increase of 1 segment per RTT);

- moving from satellite system to WLAN ($BDP_{old} > BDP_{new}$), TCP will inject in the new link more packets than it would actually fit, causing congestion and dropping packets in the bottleneck router; if all the routers through the new path have enough room to accommodate all the packets the perceived RTT will drastically increase.

### 4.5.3 Cross-Layer Signalling Architecture

Among several Cross Layer (CL) signaling architectures referenced in literature, ECLAIR [101] has been considered as baseline for the proposed design because it significantly suites the selected scenarios. In fact, it provides:

- Bi-directional CL communication channel between each pair of layers in the stack;
- Centralized control mechanism, called Optimizing Subsystem (OSS), to collect more CL adaptations, handle a multiple CL interaction, and to avoid conflicts between one or more CL interactions [102];
- Several Cross-layer enhancements via an open interface called Tuning Layer (TL) between each layer and the OSS.

Figure 4.30 shows the general architecture of ECLAIR adopted. Using this architecture TLs are a sort of middleware to mask to the Optimization Subsystem the platform dependent TCP/IP stack.

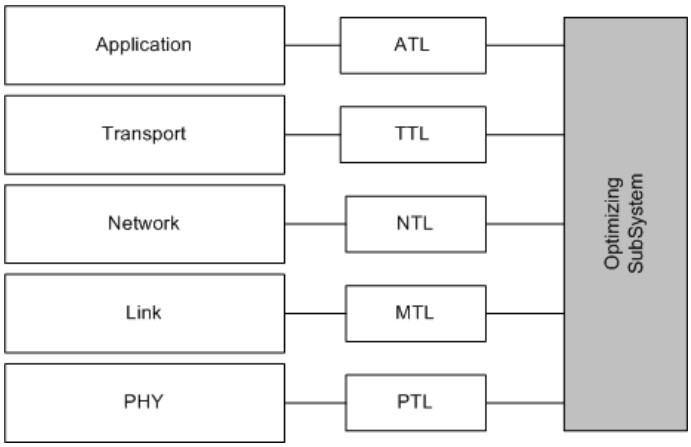


Figure 4.30. Cross-Layer Architecture Adopted (ECLAIR)

The ECLAIR architecture has been already proposed in [101] to better handle HO addressing only theoretical analysis (no simulation results). Moreover, previous works use a different approach in TCP *cwnd* management during HO.

### 4.5.4 Cross-Layer Interaction Design

To avoid the harmful dynamics that arise during handovers, three CL adaptations to the standard TCP implementation have been added: Freezing TCP window, Resetting of the sampled RTT, optimized handling of *cwnd* and *sstresh*.

#### 4.5.4.1 Freezing TCP window

While MN is changing AR, to avoid the burst overflow and an unnecessary overgrowth of the internal queues, it is useful to freeze TCP data flow. Although this enhancement is the most intuitive, it implies a few problems, which are addressed and solved with the next two enhancements.

#### 4.5.4.2 Resetting of the sampled RTT

TCP standard implementations attempt to predict future round-trip times by sampling and averaging the RTT of each packet according to Equation 2.7. To compute the RTO, usually equal to $2 \cdot sRTT$, it is useful to reset the sRTT after each change of interface, avoiding a misleading estimation of sRTT with the above mentioned consequences.

#### 4.5.4.3 Optimized handling of *cwnd* and *sstresh*

Since *ssthresh* represents the threshold above which TCP performs a linear increase of the *cwnd*, it is useful to set *ssthresh* close to the real capacity of the new channel that is related to the BDP. Moreover, after a handover from satellite to WLAN, *cwnd* may be greater than BDP, causing harmful dynamics of TCP and channel congestion. To prevent this it is advisable to set, after a handover, the *cwnd* value to the Maximum Segment Size (MSS).

### 4.5.5 Simulation

#### 4.5.5.1 Description

In order to evaluate performance of the proposed enhancements in comparison with the standard protocol stack, a custom simulator in C++ has been developed because

no other existing simulator provides explicit methods to enable CL communication channels. Through the Object Oriented modeling (OOD) [103] shown in Figure 4.31 adopted for the simulator, it is possible to map classes to elements of ECLAIR, so that the stack layering concept can be deployed in OOD using classes hierarchy. This will help practical feasibility.

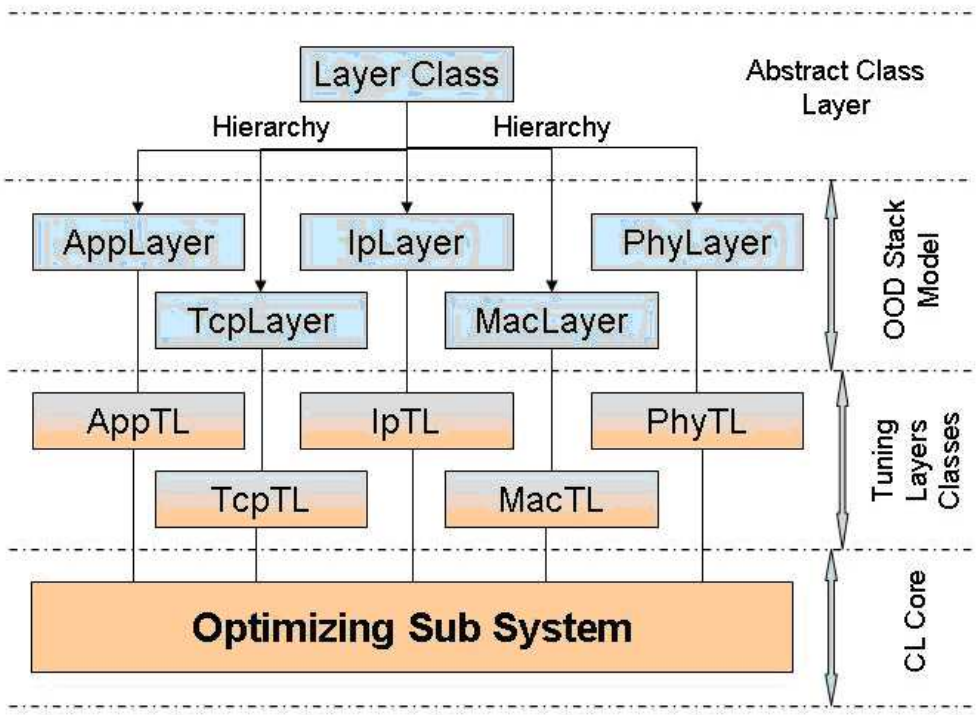


Figure 4.31. Simulator Object Oriented Simplified Model

The CL simulator core has been setup to simulate a data transfer, assuming a SAT channel of 1 Mbit/s with RTT fixed to 550 ms and a WLAN "Hot Spot" with a bandwidth of 11 Mbit/s and a RTT of 25 ms. A PER of $10^{-4}$ is assumed in both SAT and WLAN channel. Data packet dimension is fixed to 1024 bytes. It is measured the Data Transfer Time (DTT) and the instantaneous throughput of a heavy data transfer application, such as a FTP, with data file of 10 MBytes. During the transfer, at $T_0$ it is assumed that the MN, currently served by a satellite system, enters in a Hot Spot area, and thus a handover to the WLAN network occurs. After this time interval, it leaves the Hot Spot and comes back to the SAT network. This simulation

has been repeated varying the WLAN time permanence $T_{WLAN}$, and assuming the handover execution time $t^*$ constant regardless the handover direction.

#### 4.5.5.2 Performance Evaluation

Figure 4.32 shows instantaneous throughput variation as a function of time when a handover occurs during a data transfer with $t^* = 2.5s$, $T_0 = 5s$ and $T_{WLAN} = 8s$. When at $T_0$ the MN performs the HO to the "Hot Spot" using the modified stack, the throughput increases thanks to the effect of the proposed technique, which in particular avoids an overflow of the internal queues. A reset of the sRTT is also forced, causing a faster adaptation of TCP at $(T_0 + t^*)$ to new channel conditions.

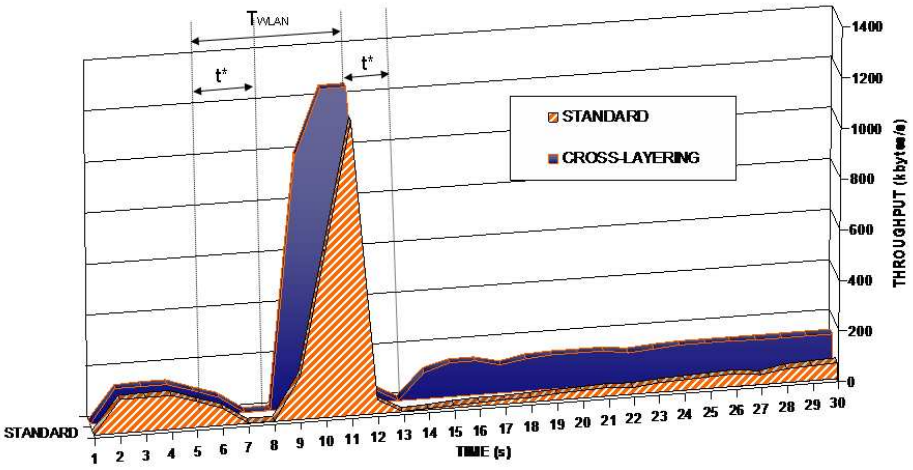


Figure 4.32. Throughput with $t^* = 2.5s$ and $T_{WLAN} = 8s$

At $T_0 + T_{WLAN}$, when the MN moves back to the SAT channel, the standard stack shows a worse response due to the high RTT that causes a slower growth of *cwnd*. With the proposed cross-layer handling of *cwnd* and *ssthresh* values it is possible to force TCP to re-apply the Slow Start algorithm and obtain a faster growth of data rate up to the BDP of the new link. This, combined with a reset of sRTT, prevents unnecessary timeouts and unwanted *ssthresh* decreases. Finally, by freezing TCP data flow, it is also possible to avoid the initial delay caused by the data stored in

the internal queues during the handover that must be sent when the new channel becomes available.

Figure 4.33 shows the variation of the Data Transfer Time (DTT) by varying $t^*$ in the interval [500 ms, 6500 ms] with $T_{WLAN}$ fixed to 7 s, using the standard and the modified stack. The horizontal line represents the transfer time if the MN keeps staying connected through the SAT channel without performing HO. The value of $t^*$, defined as $t^*_{MAX}$, at the intersection between the DTT curves and the SAT constant transfer line represents the trade-off HO execution time. If the HO procedure takes less than $t^*_{MAX}$, in case a WLAN is available, it is advisable to perform a HO to reduce DTT. If the HO procedure takes more than $t^*_{MAX}$ there are no advantages in terms of DTT with respect to the SAT constant transfer rate and thus it is better that the MN does not perform the HO, unless it is strictly necessary.

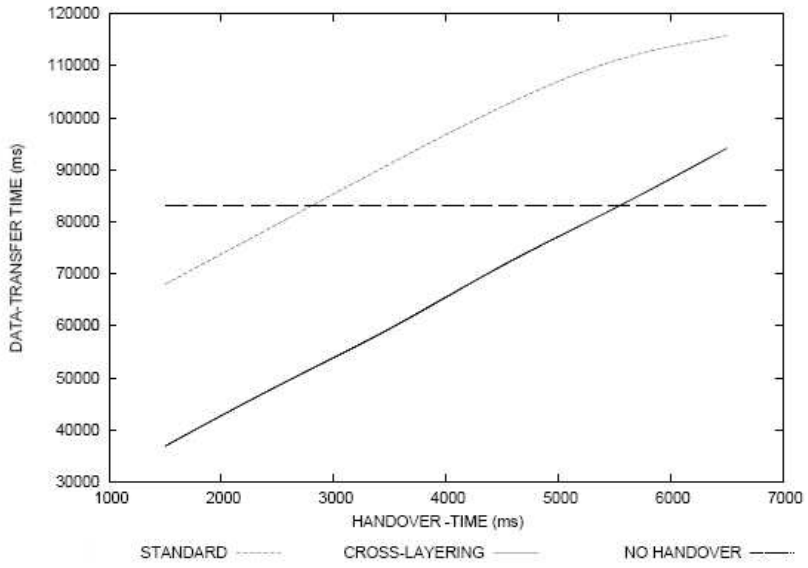


Figure 4.33. DTT with $T_{WLAN}$ constant (7 s)

In Figure 4.33, it is also evident a meaningful improvement in terms of DTT implementing the modified stack with respect to the standard (about 30s). Also $t^*_{MAX}$ value results improved: while $t^*_{MAX}$ is about 2.5 s for the standard stack, it is 5.5 s for the modified stack. As a consequence, less severe requirements in terms of HO execution time are allowed.

Figure 4.34 shows how DTT varies as a function of $T_{WLAN}$, considering a $t^*$ of

2.5 s. Values for $T_{WLAN}$ ranges between 2 s and 15 s.

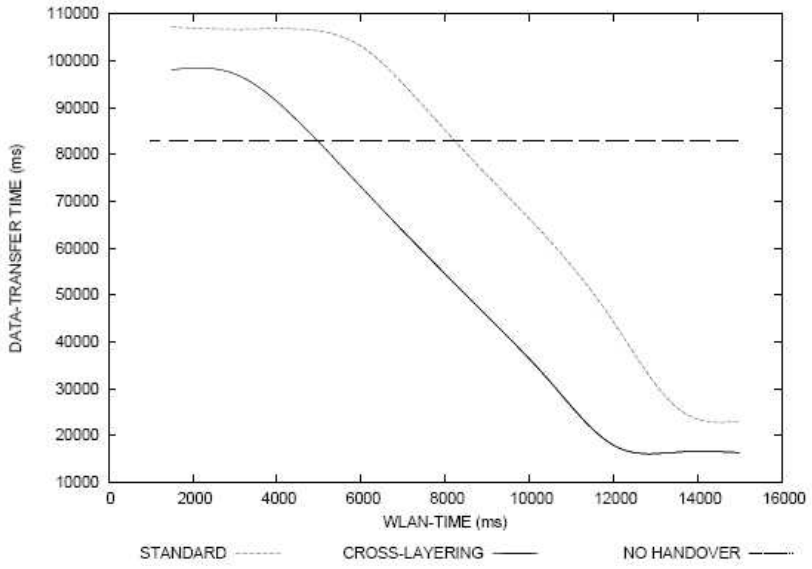


Figure 4.34. DTT with $t^*$ constant (2.5 s)

In the whole range of $T_{WLAN}$, Figure 4.34 shows that the proposed mechanisms always help to greatly alleviate the TCP problems which increase the DTT when HO is performed. When $T_{WLAN}$ is small (less than 3 s), the improvement introduced by the modified stack is relatively low because $T_{WLAN} \simeq t^*$ and of course it is not possible to full benefit of the Hot Spot coverage. Increasing the $T_{WLAN}$ beyond 3 s, the DTT of the modified stack rapidly decreases, while DTT of the standard stack keeps constant, due to the slow growth of throughput when the session switches back to the SAT channel at $T_0 + T_{WLAN}$. Then, the gap between the curves remains almost the same (about 30 s) until $T_{WLAN}$ is big enough to complete the 10 Mbytes file transfer before the communication comes back to the SAT channel. In these circumstances, the gain is limited since there is no transfer when the MN switches back to the SAT channel. $T_{WLAN-MIN}$ is the minimum $T_{WLAN}$ value that leads to a benefit, in terms of DTT decrease, while performing a HO: it can be identified as the value of $T_{WLAN}$ at the intersection of DTT curves with the SAT constant transfer rate in Figure 4.34. In practice this means that if the MN is under the Hot Spot coverage less than $T_{WLAN-MIN}$ there is no improvement in terms of DTT and then it is better not performing the HO, unless it is strictly necessary. It is possible to

note that $T_{WLAN-MIN}$ is lower for the modified stack implying that we can accept shorter permanence time in Hot Spot coverage areas.

## 4.6 Interworking between MANET and Satellite Systems for Emergency Applications

SAVION is a dynamic mesh network solution for voice and data designed to provide a reliable communication capability in emergency situations (i.e., disaster area, rescue operations, public safety or military missions). SAVION [2] project aimed to allow the integration between SAVION communication units with all the existing communication devices used by different operative teams converging in a mission (i.e. rescue team, VVFF, policy, etc.). In fact, in each SAVION network an unit, working as a gateway and interfaced to a VHF/UHF device, satellite modem/phone, cellular phone or other existing communication devices, allows at any other unit to be connected to the external networks. The SAVION solution can be considered as an efficient "gap filler" aiming to provide communication services in either areas characterized by the total absence of infrastructures or in situations requiring interaction among teams using different communication technologies.

The proposed solution to achieve the total integration of the SAVION network with any other radio units or network (i.e. IP, PSTN, ISDN) has been based on the "translation" of both voice and data signals into the IP format performed by an IP multimedia gateway, and vice versa. In addition, even if SAVION has been designed to be interfaced to any IP access network, satellite networks have been identified as an optimal solution in order to guarantee an ubiquitous, broadband, uninterrupted and wireless access. A generic scenario, highlighting the SAVION characteristics, is shown in Figure 4.35.

The author contribution in the SAVION project has specifically concerned the interface design and the field trials execution.

[2]Work developed in the frame of Savion project supported by Foreign Affair Ministers of Italy and Israel (bilateral cooperation).

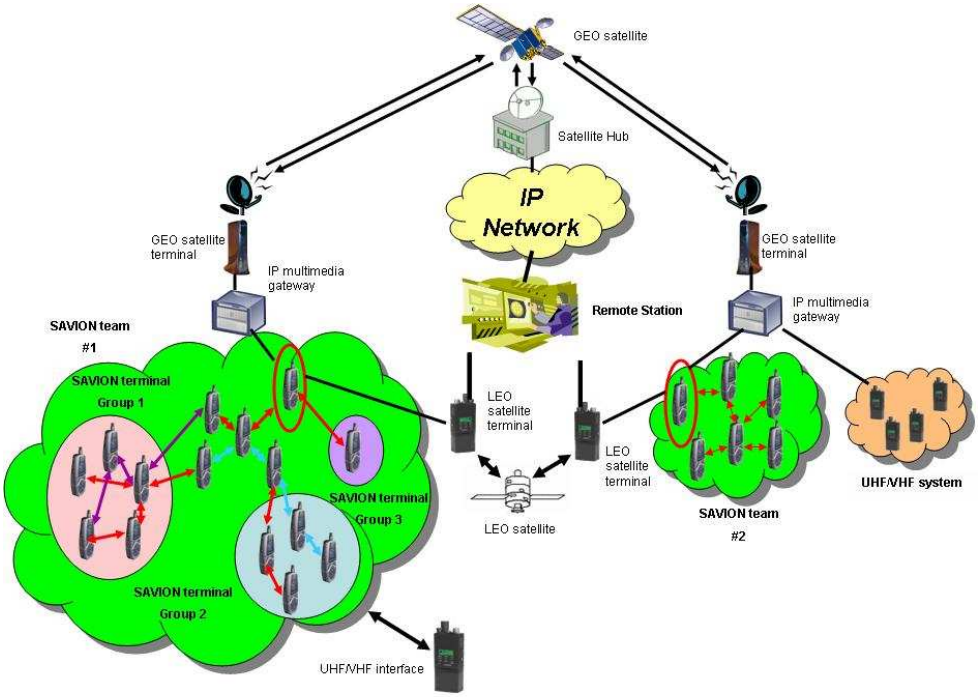


Figure 4.35. SAVION scenario

### 4.6.1 MANET-Satellite interconnection

Three different communication segments are involved in the SAVION system:

1. A mobile ad-hoc mesh network (MANET);
2. A satellite segment either narrowband and broadband;
3. A MANET-Satellite interface.

The system can be configured, time by time, depending on both the operative needs and the requested services. In any specific case, the most important task is to build up interconnection between the ad-hoc terminal, set as gateway, and the satellite terminal. Two different operational satellite solutions have been considered to cover the needs of different operative scenarios: Globalstar (Telit SAT 600 terminal) [105] and a Ku band satellite of the Eutelsat fleet (e-bird 33° east) with the ground segment from Hughes Network Systems (DW7000 terminal) [106]. In details:

- *Telit SAT 600.* This a new generation Globalstar mobile phone working in both GSM and satellite mode. It presents a serial interface to be connected to external devices for both voice and data transmissions.

- *DW7000.* This Hughes broadband terminal is based on the IPoS (IP over Satellite) global standard, optimized for high-speed IP. It acts as an IP router, and then is able to be interconnected to external devices/PCs through an Ethernet interface.

#### 4.6.1.1 Interface MANET-narrowband satellite system

The TELIT SAT 600 mobile phone can be directly connected to the SAVION gateway unit through the data transmission module DT 600 included in the Telit data kit pack [104]. Such a module allows the SAVION gateway unit to use TELIT SAT 600 as an external modem by installing an appropriate driver. The DT 600 module is composed of three units (Figure 4.36):

- a cable connecting TELIT SAT 600;

- a serial cable to connect the SAVION gateway;

- a connector for external power supply.

Figure 4.36. MANET-Globalstar terminal interconnection

#### 4.6.1.2 Interface MANET-broadband satellite system

The conversion of both the analogue voice signal and digital data, coming from the SAVION gateway unit, into IP datagrams to forward to the IP broadband satellite terminal is performed through an IP multimedia gateway able to interface radio systems to IP networks (Figure 4.37).

- IP interface. The interconnection between the IP multimedia gateway and the satellite terminal is simply set up through a 10/100 Mpbs Ethernet cable (RJ-45 connectors, automatic cross-over adaptation). Then, by using Windows OS to configure TCP/IP settings of the IP multimedia gateway, a local network connection can established by choosing, as default gateway, the IP address of the satellite terminal.
- Radio interface. The radio interface is based on a MINIMATE 18-pin connector able to simultaneously 2 different radios.

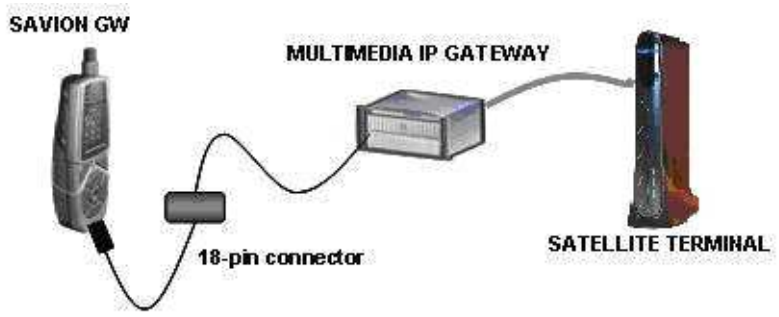


Figure 4.37. MANET-VSAT interconnection

### 4.6.2 Achievements

In SAVION project, volunteer fire brigades of Trento have been involved to define requirements and test the system in the different phases of its development. In particular, an extensive trial campaign has been carried out in Riva del Garda (Trento) in order to test SAVION system in operative scenarios characterizing the fire fighter's activity. Furthermore, tests have been oriented to both highlight the

properties of the ad-hoc mesh network and demonstrate the interconnection to the satellite segment, both through a broadband modem and through a narrowband terminal. The last set of trials involving data transfer through the Globalstar terminal were performed in Rome at Telespazio premises [107].

#### 4.6.2.1 Requirements

To evaluate coverage requirements, it is considered the path loss over the channel given by the following model [108]:

$$L(d) = L_{one-slope}(d) + M_w = l_0 + 10 \cdot \gamma \cdot log(d) + M_w \tag{4.10}$$

where $l_0$ is the path loss at 1 m distance, $\gamma$ is the power decay index or the path loss exponent, $d$ is the distance between the transmitter and the receiver, $M_w$ takes into account the features of the propagation environment (i.e. presence of walls, doors, trees, etc.).

Based on the fire fighter's needs, and assuming that the users are uniformly distributed in a bi-dimensional plane, three different classes of distance with respect to the features of the operational area have been considered:

- *Class 1* (outdoor areas without obstacles). Maximum single-hop distance ∼500 m;
- *Class 2* (outdoor areas with few obstacles). Maximum distance ∼300 m;
- *Class 3* (indoor areas with many obstacles). Maximum distance ∼50 m.

As far as the network virtual infrastructure is concerned, it must be self-consistent and will not rely on static or designated infrastructure of any kind. The network should dynamically update itself according to rapid changes in the network topology. Regarding radio communications in the satellite segment from 1 to 8 channels for narrow band and more than 8 wide band channels are required. This is justified by the need to both support simultaneous communications for a given operative team and transfer images concerning the event situation to the station (no real-time transfer of fixed images through narrow band satellite segment or real-time transfer of moving images through the wide band channel). To access databases

to research data, dangerous substances characteristics, phone numbers or other information, communication capability must be compliant with service requirements, which means to allow data rates from a few kbit/s for narrowband satellite segment up to some Mbit/s for broadband satellite segment.

#### 4.6.2.2 Ad-Hoc Mesh Network Tests

A first set of tests has involved a little group of 6 SAVION terminals (Figure 4.38) deployed in different environments to provide the following services:

- private calls;
- simultaneously private calls;
- group calls;
- broadcast calls;
- data transfers.

The metric used to evaluate test results includes:

- type of supported applications;
- Quality of Service (QoS);
- achieved distance in a single hop;
- overall coverage.

**T.1** *Indoor environment (fire fighter's base).* Deployment of a fire fighters team (6 units) in the numerous rooms of their operative base (2 floors). All the benefits due to the features of the ad-hoc mesh network have been successfully tested (i.e., dynamic self configuration, multi-hop and multi-session) simulating a real intervention to put out the fire inside a building. Data transfers have been successfully performed (even simultaneously at vocal calls). In the tests a clang, and some time difficult to understand, has been experienced. The order of magnitude of the distance between two consecutive units (indoor) is about 10 meters (presence of two or three walls in between).

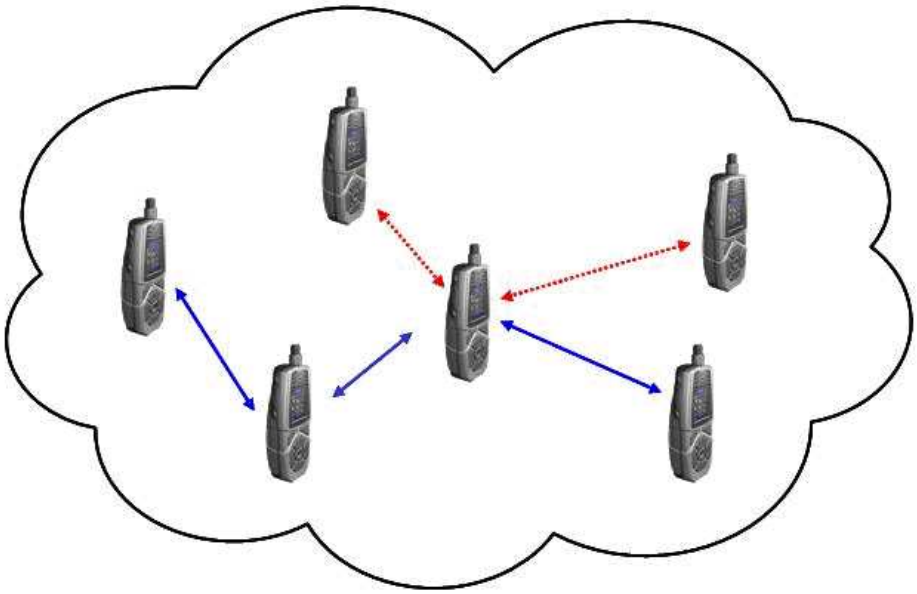

Figure 4.38. Ad-hoc mesh network

**T.2** *Indoor environment (fair sheds).* Deployment of a fire fighters team (6 units) in an indoor environment with large open spaces and few obstacles (fair sheds of Riva del Garda). Tests of communications (multiple private calls, group calls, broadcast calls) among distant units have been performed, simulating usually coordination tasks during mass events. The maximum covered distance between two consecutive units has been of about 200 meters. A good coverage of the overall area has been experienced in the case the grid of units are uniformly spaced of not more than 100 meters.

**T.3** *Tunnels.* Tests aiming to carry the radio signal out of the tunnel through a limited number of hops for both data and voice communications (even simultaneously). The connection to the bottom of the tunnel has been performed through a single hop (200 meters). Furthermore, a double-hop communication consisting on a transmitter at the bottom of the tunnel, a relay at the entry of the tunnel, and the receiver about 100 meters far from the tunnel entry, has been successfully set up (simultaneous calls from the bottom of the tunnel and both chat and picture transfer along the tunnel and among units sited in and out the tunnel).

**T.4** *Outdoor environment (lake shore).* Tests over the lake shore have been performed to reproduce an approximated "open space" scenario. The maximum covered distance between two units has been of about 320 meters.

Table 4.5 summarizes such results. In particular, it is possible to observe a good match with the requirements presented in Section 4.6.2.1 as far as class 1 and class 2 are concerned while class 3 is characterized by sub-optimal performance that will be improved in the following steps of the SAVION project.

| **Test id** | **Applications** | **QoS** | **Single-hop dist.** | **Overall coverage** |
|---|---|---|---|---|
| **T.1** | Call | discrete | $\sim 10m$ | $\sim 200m^2$ |
| | Data transfer | good | | |
| **T.2** | Call | discrete | $\sim 200m$ | $\sim 8000m^2$ |
| | Data transfer | good | | |
| **T.3** | Call | good | $\sim 200m$ | – |
| | Data transfer | good | | |
| **T.4** | Call | good | $\sim 320m$ | – |
| | Data transfer | good | | |

Table 4.5. Trial achievements

#### 4.6.2.3 Voice service through broadband satellite terminal

With this set of tests the interoperability of the SAVION network with either other VHF/UHF radio systems or IP terminals (IP phones and PCs) via a broadband satellite access segment has been demonstrated. The architecture of the test bed is shown in the Figure 4.39, where the interface elements are represented by functional blocks.

In this context, each SAVION terminal belonging to a group has been successfully connected (for both voice and data communication) via a satellite IP network to the following remote systems/terminals:

- Fire fighter's radio (VVFF radio);
- a generic private mobile radio (PMR radio);

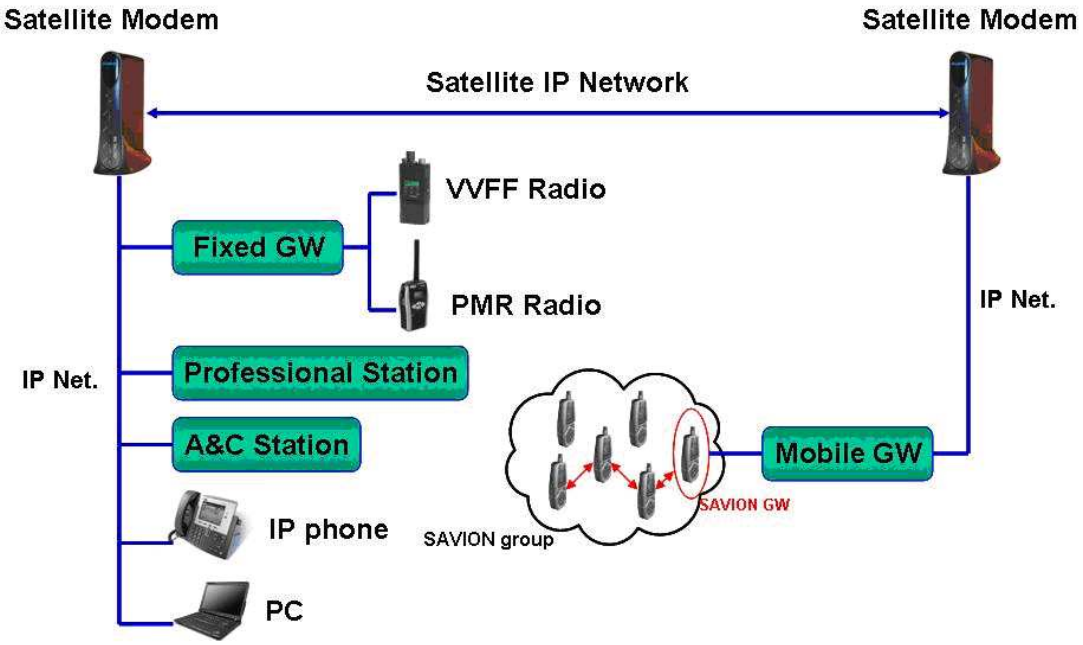


Figure 4.39. Test-bed configuration

- an IP phone;
- a PC.

As far as the interface elements is concerned:

- the *Fixed GW* is a fixed station used to interconnect the VHF radio of the fire fighters (or the generic PMR radio) to the IP satellite modem;
- the *Mobile GW* is a mobile station used to interconnect the SAVION GW to the IP satellite modem. It is configured and managed by a PC through an USB interface;
- the *Professional Station* manages the communications. It is based on a software package that allows to manage up to 10 simultaneous voice calls;
- the *A&C Station* is the control center that manages and controls all the tasks performed in the fixed/mobile GW.

Figure 4.40 shows the picture of the test bed deployed in Trento.

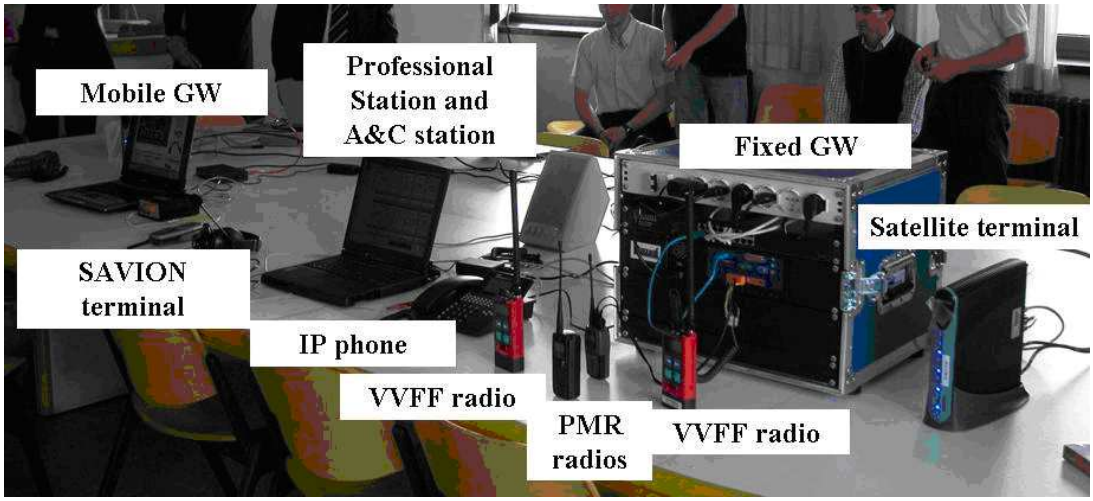


Figure 4.40. Test-bed in Trento

#### 4.6.2.4 Data transfers through Globalstar terminal

The capabilities of the Globalstar terminals (Telit DT600), provided by Elsacom [104], to allow data communications have been proved. The test bed included two Telit DT600 terminals connected to two PCs, running Windows XP operating system, through an RS232-USB adapter. Both a data channel (non-TCP/IP) and a TCP/IP connection have been successfully setup and used for chatting and file transfers.

1. *Data channel.* In the PCs, Windows HyperTerminal has been launched to create a new test connection through a COM port which the satellite phone is connected to. Then, the procedure to establish the data connection and exchange data can be summarized as follows.

   (a) The caller PC must dial the data number (different from the voice call number) associated to the satellite terminal, connected to the called PC. The corresponding command is "ATD¡data_number¿".

   (b) Consequently, on the called PC a "`RING`" signal appeared every 2 seconds. After the first one, the call can be picked up by typing the "`ATA`" command.

(c) After these steps the connection is established after maximum 30 s. Usually, after about 15 s the "`CONNECT`" message appeared on both PCs to announce the successful data connection.

(d) Finally, it was possible to start a chat session as well as a file transfer by using one of the serial protocol offered by HyperTerminal (i.e., Zmodem).

2. *TCP/IP connection*. Windows XP allows to configure a Remote Connection Server answering to the incoming calls. In this way, satellite phone acting as a modem and connected to the called PC can receive connection establishment requests from the other satellite phone connected to the caller PC. Furthermore, a SLIP/PPP tunnel will be established between the two PCs in order to allow two-way IP traffic. Using the created dial-up connection, the most popular application protocols (i.e., HTTP, FTP, etc.) has been used to transfer data between the two PCs associated to local IP addresses. Usually, 30 s were needed to set up the connection and run applications.

# Chapter 5

# Design and Development of a Satellite Network Emulation Platform (SNEP)

A classical approach to test and debug new communication protocols is based on the experimentation through real technology. This test methodology generally leads to several issues:

- high costs mainly due tothe involved technology (i.e. the target operational network includes a satellite system.);
- lack of control on the network conditions that makes very difficult to achieve reliable measurements and tests;
- Reduced (or sometime completely absent) knowledge of lower layer protocol or software implemented in the intermediate entities involved in the experimentation. This further reduces the value of the resulted measurements and tests.

On the contrary, simulation and particularly event-driven simulation is a powerful tool to achieve cheap and fast experimentation. In fact, simulation approach aims at isolating the most critical issues of the target network in order to model them. In this way, simulation allows at performing simplified and const-effective

tests. On the other hand, this tendency to move from the general to the particular can cause the loss of effects generated by the mutual interaction of different software or hardware "entities" involved in the real networks. For example, the analysis of the TCP dynamics over a simulation platform, that models the target link just as a "delay and bandwidth", neglects the potential effects of:

- the queue disciplines adopted in the edge routers;
- protocols and transparent mechanisms implemented in the intermediate nodes (PEP);
- limitations characterizing the user end-system.

Furthermore, it is also difficult to define a general model for characterizing the Internet traffic characteristics, owing to its extreme heterogeneity and dynamism.

To meet such requirements, emulators have been conceived and developed. In particular, network emulators reproduce in real-time the services and quality of service of the target network, while artificial network impairments (i.e., end-to-end delay) are forced by software modules. In this way, test results can be achieved by using both real protocols and distributed applications and by creating a controlled communication environment. This characteristic makes the emulation approach a trade off between simulation and real testing. Existing emulation tools can be usually classified in:

- Static emulation tools, consisting in applying static constraints on data flows;
- Dynamic emulation tools, allowing the variation of several parameters (i.e. according to simulated model).

In such a frame, a new emulator has been developed extending the notion of dynamic emulation in order to drive the emulation using as input the emulated traffic. This novel approach can be defined "active" and "self-controlled" to highlight its capability to decide how to impair the flows depending on packet or flow characteristics.

The name chosen for the proposed emulation platform is "Satellite Network Emulation Platform" (SNEP), where *Satellite* indicates that the target environment

include a GEO satellite (but open to be extended to wireless terrestrial systems) and *Network* is for emphasize the capability to reproduce a whole satellite system including multiple terminals, a Network Control Center (NCC), interfaces to other systems, etc. In particular, the target architecture is DVB-RCS like, focusing on a DAMA access scheme.

## 5.1 Survey on emulation tools

The simplest way for emulating a target network is based on a static emulation approach. It consists in applying fixed constraints to the data flows and as a consequence, once defined, emulation parameters do not evolve during the experiment. A number of tools enable a static network emulation. Mainly, these tools are traffic shapers used as a "black box" between real end users. Dummynet [115] is implemented into the Unix FreeBSD kernel and works by simply intercepting the traffic and simulating the effects of both routers with bounded queue size with a given queuing discipline and communication links (pipes) with fixed bandwidth and delay. A similar approach is developed for Linux Kernel (NISTnet) [116], providing better control on the emulation impairments (i.e., packet duplication, different distributions for the delay, etc.). NetEm [118] is a recent enhancement of the traffic control facilities of Linux, documented in [130], importing most of the NISTNet functions. ONE [117] is based on the same idea and provides the same functions of the previous tools, but working on a single Solaris-based workstation. ENDE [120] emulates end-to-end delays between two hosts running in the local host and without requiring access to the remote one. In detail, ENDE generates accurate traces of one-way delays between two hosts in delay-observing mode, while in the delay-impacting mode uses such traces to simulate the operation of a protocol or an application as if it were running on the network. DelayLine [121] is a user level emulator library that provides a fully configurable mechanism for emulating wide area network communications on a local area network.

The main advantages of these static emulation tools are their simplicity of use, accuracy and good performance (processing dynamics do not alter the target behavior). On the other hand, they do not allow the implementation of complex

algorithms referring to real network characteristics dynamically changing due to external parameters such as traffic load, congestion, weather conditions and mobility.

Various types of dynamic emulation approaches have been proposed in the literature. A first approach, presented in [122], is based on a emulation tool called "NETShaper". NETShaper is implemented as a Linux kernel module and provides emulation on link layer that is completely transparent to applications and also network layer protocols. It uses a bandwidth throttling approach that exactly mimics the actual behavior of shared network devices. Patterns of the actual impairments that drive the emulation can be, for instance, derived from traces of real network experiments. Emulators built on a trace-based approach have the advantage of faithfully reproducing the behavior of a real network, while capturing traces can be difficult and expansive in time as well as simple snapshots of specific network conditions at some times.

Another approach to drive dynamically the emulations consists in combining the emulation with real-time simulation. Ns-2 [30] operates with a modified scheduler and is able to process real packets. Two modes are proposed: an opaque mode where real packets are not modified, and a protocol mode where real packets can interact with the simulated model. The drawback is that the real-time scheduler makes emulation mode of ns-2 very resource consuming, leading to very low emulation capacity (in term of processing load).

A last approach envisages a two-state solution, where an off-line simulation stage is used to compute the dynamics of the emulation parameters and then an emulation stage reproduces the target scenario by computing the simulation outcomes. In this frame, SWINE [123] is a two-stage tool for wireless emulation based on the real-time control of Dummynet.

### 5.1.1 Examples of emulation platforms

This section provides details of two emulation platforms dealing with satellite environments and used as references for collecting the requirements for the proposed solution. In particular, their most relevant characteristics are summarized and evaluated in the frame of the target goal: design and implementation of a dynamic and "self-controlled" emulator platform for satellite networks (DVB-RCS like).

#### 5.1.1.1 A Testbed for Upper Layers Performance Evaluation

University of Bologna (UoB) and National Inter-University Consortium for Telecommunications (CNIT) have conceived and set-up the integrated testbed aiming to evaluate performance of TCP transport protocols over both satellite and terrestrial networks. The testbed consists of several PCs running either the Linux operating system or FreeBSD (running a TCP Westwood released by UCLA Computer Science department). The overall configuration is shown in Figure 5.1.

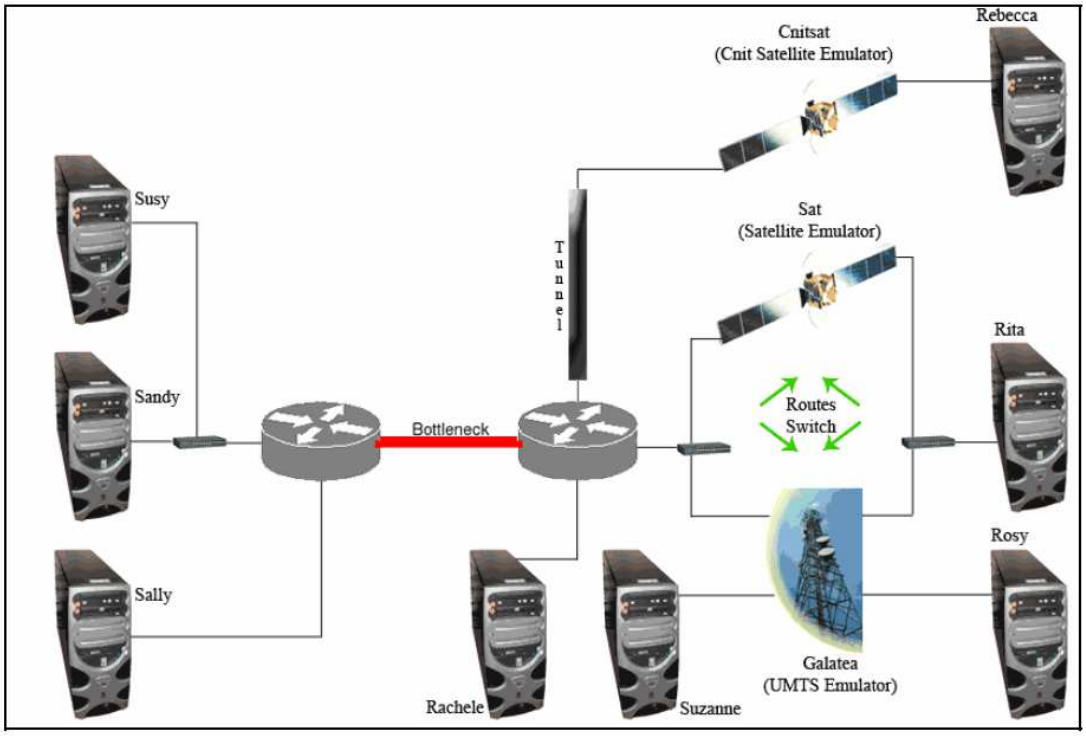


Figure 5.1. UoB-CNIT testbed topology

PCs running applications (end-systems) have been patched with the Multi-TCP package [124] allowing to both switch among a large set of TCP variants (i.e. TCP Hybla, TCP Westwood, TCP Vegas, TCP NewReno, etc.) and enable/disable TCP options or architectural agents (i.e. SACK, spacing, PEP agents etc.). All the connections share the same bottleneck link characterized by a fixed bandwidth limitation and bounded by routers where congestion events are managed through either a DT (Drop Tail) or a RED (Random Early Detection) queue policy. Both TCP

and physical parameters can be fully configurable from a web interface. The wireless connections are realized by means of three different emulators:

- a commercial emulator reproducing the UMTS propagation environment named Galatea [125];
- a satellite emulator based on NISTNet [116];
- the CNIT packet-switching satellite emulator [126] allowing the tuning of both transmission medium and MAC parameters.

The CNIT emulator is not installed at the UoB premises. It is connected to the rest of the testbed through the creation of an IP tunnel by exploiting the VPN paradigm.

This platform appears very efficient and flexible in managing transport layer and the test configuration, especially thanks to an optimized web interface. The presence of a large number of nodes and links allows emulation of a large set of communication scenarios involving wired, terrestrial wireless and satellite segments. Furthermore, accurate probes implemented in the Linux kernel makes the testbed suited to evaluate performance of applications and new transport protocols.

On the other hand, as far as the satellite segment is concerned, lower layers are managed by fixed constraints not able to reproduce either dynamic variations and a star-based network topology . As a matter of fact, the satellite segment is a simple point-to-point link accessed in an Aloha-based scheme. As described in [126], CNIT emulator implements on-board satellite packet switching supporting different types of data link layers and different encapsulation formats at the data link layer, while the management of DAMA access scheme is not addressed. Finally, flow control at IP level is not supported so far.

#### 5.1.1.2 Network Engineering Platform (NEP)

The Network Engineering Platform (NEP), described in Section 3.7.2, is an emulation platform designed to representatively emulate a DVB-S/DVB-RCS satellite telecommunication system. The implemented network topology is shown in Figure 5.2

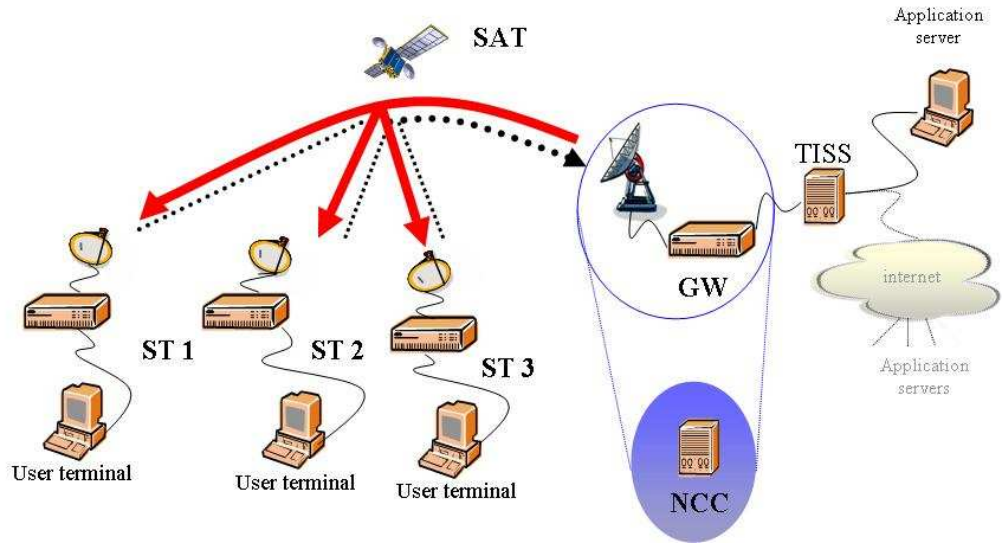


Figure 5.2. NEP emulator topology

With respect to other emulation platforms, NEP originality is concerned functions it implements at both access and resource management sub-layer. In particular, the access layer provides both IP interfaces managing data exchanges between user terminals and satellite terminals and ATM/MPEG-2 hybrid interfaces managing a transport stream in the satellite link. In detail, the emulated satellite terminal manages traffic as follows:

- In transmission mode, the ATM interface function receives data sent by the application terminal (IP packet in native IP mode, Ethernet frames) and calls the ATM transport function that performs the AAL5 encapsulation and cell segmentation. The resulting cells will then be processed by the MAC sub-layer.
- In reception mode, the MPEG transport function receives data arriving from the satellite channel (gateway performs a MPEG-2 encapsulation), re-assembles and de-encapsulates it and retransmits the resulting IP packet/Ethernet frame towards the application terminal.

Resource management functions are distributed in both satellite terminals and gateway (taking in charge the NCC functionalities) and are responsible for all RCS signalling (capacity request and allocation messages). All the access scheme presented in Section 1.5.3.2 are supported.

NEP limitations can be summarized as follows:

- the forward link (gateway → satellite terminal) is less modeled than return link. In particular, neither flow control nor ACM functionalities are supported;
- MAC parameters are not tunable. Then, there is no way for testing enhancements at lower layers;
- Superframe duration of 1 s makes VBDC and RBDC disciplines behave in a very similar way (especially in case of short transfers);
- No probes are available for monitoring low-level behavior (i.e., DAMA, ATM, MPEG)
- There is not support for selecting transport protocol.
- Currently, NEP is an isolated platform. It is not configured for any external access.

Definitely, NEP appears an efficient tool for testing application on a likely DVB-RCS environment but does not allow to easily configure and test new network protocols or to modify lower layer setting.

## 5.2 The Satellite Network Emulator Platform

The idea behind the development of the SNEP is to create an emulation platform, reproducing a DVB-RCS network architecture, extremely flexible and easily usable. Flexibility is meant as the capability to manage every protocol layer (i.e., select target protocols, set parameters, configure probes, etc.). Simultaneously, test sessions can be entirely configured and performed via a graphical web interface. Its novel contribution with respect to existing emulator platform can be summarized into two items:

- enhancement of the dynamic emulation notion to drive the emulation with the emulated traffic. The variations of parameters is internally managed by SNEP, while users can easily configure or add algorithms/parameters in the set up phase.

- Access to both parameters and statistics concerning all the protocol layers.

SNEP is designed to emulate a DVB-RCS system based on a single Hub/gateway (GW) station including NCC functionalities. In particular, GW provides a twofold interface: an IP interface to the Internet core networks via a Local Area Network that allows remote users to access to the platform, and an interface towards one or more emulated satellite terminals (STs). The return link (from STs to GW) is compliant to DVB-RCS standard, while transmission from GW to STs is based on DVB-S (DVB-S2). A satellite channel (SAT), characterized by GEO physical parameters (i.e., one-way delay of $\sim$ 250 ms), is implemented between GW and STs. Finally, user terminals (UTs) are connected to STs and run applications.

SNEP implementation has been scheduled in three stages:

**Stage 1** *Platform set up.*

- Configuration and interconnection of PCs involved in the emulator platform;
- Transparent integration of SNEP with the LAN of DIE (Department of Electronics Engineering) of "Tor Vergata" university;
- Installation of a vast gamut of TCP protocols easily selectable;
- Configuration of the physical parameters of the emulated satellite link;
- Configuration of traffic sources.

**Stage 2** *DVB-RCS emulation.*

- Management of Differentiated Services (DiffServ) at IP layer;
- Implementation of DAMA schemes compliant to DVB-RCS standard;
- Implementation of probes to collect statistics concerning different levels of the implemented protocol stack;
- Implementation of routines for the processing of the collected data.

**Stage 3** *Development of advanced functionalities.*

- Implementation of interfaces towards other communication systems (i.e. WiFi);

- Implementation of traffic loaders;
- Implementation of TCP PEP agents;
- Management of security issues (IPsec support).

Currently, stage 1 was completed and stage 2 is in progress.

### 5.2.1 Operational Environment

The SNEP is designed to operate as a stand-alone platform, providing interfaces to be interconnected to external terminals/systems as well. It is composed of 5 PCs that are interconnected either directly or through network equipments. The reference architecture is shown in the Figure 5.3, which shows also how the different parts of the target real system (DVB-RCS like) are mapped onto the different components of the emulator.

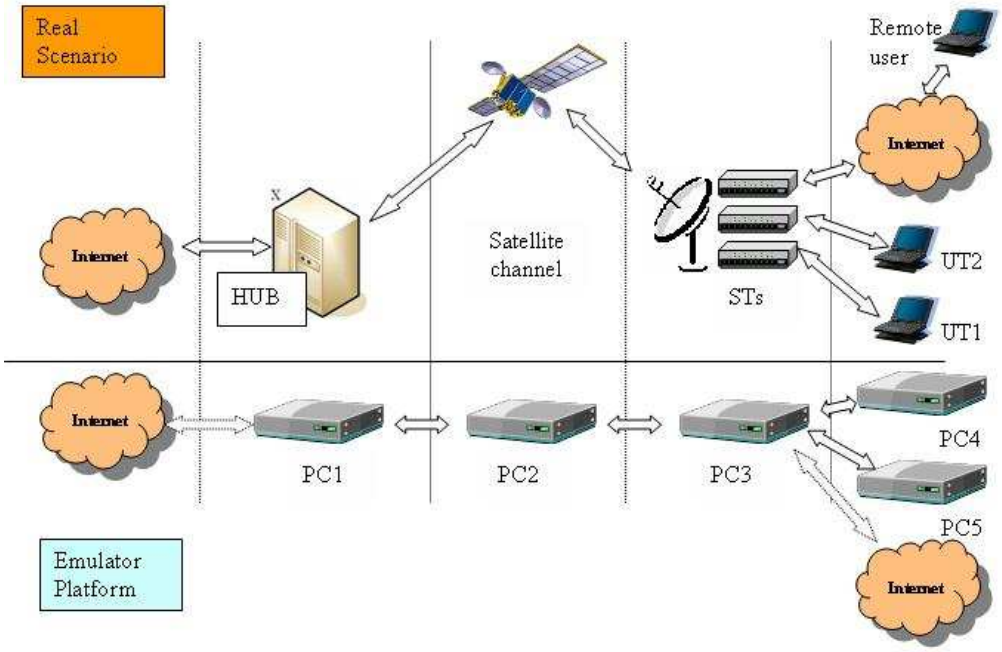


Figure 5.3. SNEP vs. Real System

In more detail, the platform includes:

- 1 PC to emulate the Hub station including NCC functionalities (PC1). It also hosts the web interface allowing the configuration of all the PCs.
- 1 PC to emulate the satellite channel (PC2).
- 1 PC to emulate the satellite terminals (PC3); it represents a satellite access system emulating one or more satellite terminals.
- 2 PCs acting as user terminals (PC4, PC5), in which the user applications or traffic loaders run.

#### 5.2.1.1 Hardware Configuration

The hardware equipments involved in the SNEP platform and the corresponding costs are reported in Table 5.1.

| Name | Description | Quantity | Cost (EUR) |
|---|---|---|---|
| Case barebone | EX761 BAREBONE-VGA+ PCI-E SOCKET 754 | 5 | 926,10 |
| CPU | SEMPRON 2800/754 - BOXED | 5 | 281,75 |
| RAM | 1GB (512x2) DDR PC3200 DUAL CHANNEL | 5 | 374,85 |
| Hard Disk | 80GB Plus9 | 5 | 176,40 |
| DVD reader | DVD 16X48X IDE BLACK bulk | 5 | 75,95 |
| NIC | 10/100 REALTEK PCI | 3 | 17,64 |
| Switch keyboard/ monitor/mouse | KVM 4 PORTS (Cables Inclusive) | 1 | 12,5 |
| Monitor | M17RKA 17" LCD TFT AUDIO | 1 | 130,34 |
| Network cables | Cable PATCH 1mt | 5 | 9.80 |
| Network cables | Cable CROSS 1mt | 2 | 5,88 |
| Keyboard | Keyboard ITA ps2 | 1 | 5,88 |
| Mouse | WHEELMOUSE OPTICAL BLACK oem | 1 | 9,80 |
| Switch 10/100 | 16-PORTS SWITCH 10/100 | 1 | 51,94 |
| | | | **Total: 2525,46** |

Table 5.1. SNEP hardware

A picture is shown in Figure 5.4.


Figure 5.4. SNEP configuration

#### 5.2.1.2 Operating systems & software packages

SNEP modules are developed over Linux OS in order to take advantage from the Open Source development strategy [127] that makes it continuously up to date, thanks to the constant activity of a vast community of programmers. Linux source code can be entirely explored and handled giving the opportunity to customize the system according to the user needs. In particular, the used kernel is the version 2.6.20.1, which offers a large set of functions for classifying and scheduling network traffic [128][129][130], IPsec support [109], and a large set of transport protocols including UDP-Lite [132] and new variants of TCP protocols (i.e., TCP Hybla, TCP Westwood+, TCP Veno, TCP Low-Priority). Finally, all of the most used applications are provided: i.e., Apache as web server, Mozilla as web client, proftpd as FTP

server, gftp as FTP client, Gnomemeeting for Videoconferencing, VideoLanCLient for Videostreaming.

User terminals (PC4 and PC5) are also able to operate (partitioning) with Windows OS and FreeBSD in order to allow respectively the testing of commercial applications (Microsoft@) and the installation of the UCLA release of TCP Westwood (implemented for FreeBSD).

### 5.2.2 Technical description

The target satellite networks require complex models and mechanisms to be emulated due to the highly dynamic behavior of the target protocols. Protocols react depending on both internal factors and external factors, such as traffic crossing the network. In particular, DAMA access scheme proposes five different types of traffic assignment techniques (VBDC, AVBDC, RBDC, CRA, FCA) that can be jointly implemented. VBDC, AVBDC and RBDC depend on the traffic characteristics while the others not. Therefore, the behavior of the access scheme is not predictable in advance, but depends on the emulated scenario. The only way to have a realistic emulation behavior is to react in real time at the emulated traffic conditions. In this sense, the emulation needs to be "active" and "self-controlled".

To achieve such goals, the proposed emulation architecture is divided into two sets of modules:

- *High level modules* managing emulation models, monitoring traffic and reacting on the basis of the data transferred. They envisage a large number of programmes, written in C programming language, which allow the dynamic configuration and the management of the emulation on the basis of the received configuration parameters. The configuration parameters are provided by users through a graphical interface.

- *Low level modules* capturing and sending data and applying low level rules according to the high level emulation models. They are based on the Linux traffic control functions included in the kernel code. In particular the traffic control code consists of the following components:

  - *queuing disciplines* controlling how packets enqueued on that device are processed;
  - *classes* defining the way packets must be processed;
  - *filters* allowing to distinguish among different classes;
  - *policing* allowing to switch among classes according to traffic variations.

The communication between the two levels is performed through "tc" [128], which is a user-space program used to manipulate individual traffic control elements. Figure 5.5 shows this emulation architecture.

The platform configuration, the setting of parameters, the execution of simulation sessions and the access of the results/statistics can be performed through a user interface based on a graphical front-end. An user can:

- Configure the network architecture (i.e. enable PEP, define the number of the active STs, etc);
- Set physical parameters (i.e. satellite delay, loss distribution, bandwidth limitations);
- Set the QoS policy;
- Select TCP protocol;
- Configure and schedule traffic loaders;
- Execute simulation sessions;
- Select probes to store and display statistics.

The configuration front end also include an "expert user" mode to build up the platform or add/change modules.

With reference to Figure 5.3, the various SNEP functionalities can be classified on the basis of the network element they refer to.

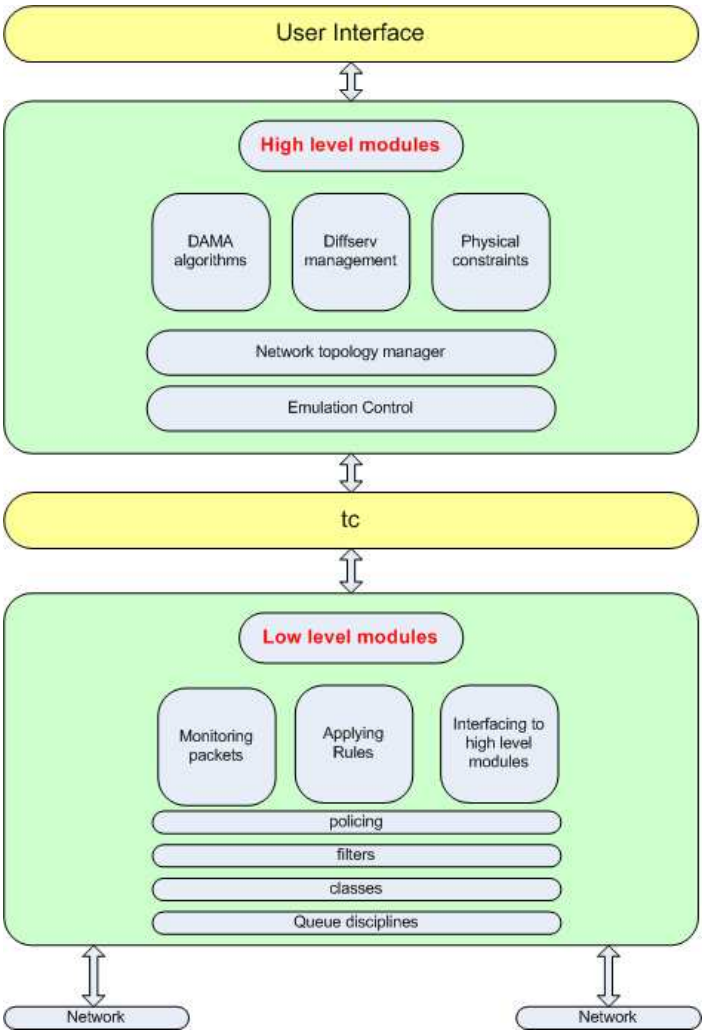


Figure 5.5. SNEP general architecture

#### 5.2.2.1 User Terminals

The user terminals (UTs) generate one or more IP flows in a controlled manner, including also specific traffic generators acting as traffic loaders. Application and transport protocols can be chosen within a large set of variants. Finally, it is possible to schedule the flows within of the simulation session and classify traffic by encoding the 6-bit value called "Differentiated Services Code Point" (DSCP) into the 8-bit Differentiated Services (DS) field of the IP packet header [133]. The following commonly-defined Per-Hop Behavior (PHB) are supported:

- *Best Effort* (BE), which does not meet any QoS requirement;
- *Expedited Forwarding* (EF), which requires low delay, low loss and low jitter (suitable for voice, video and other real time services);
- *Assured Forwarding* (AF), which provides assurance of delivery as long as the traffic does not exceed some subscribed rate (traffic exceeding the subscription rate faces a higher probability of being dropped if congestion occurs).

#### 5.2.2.2 Satellite Terminals

The emulated satellite terminal (ST) can host the functionalities of multiple real STs (Figure 5.3). It includes functions of two layers:

- network (IP router);
- resource management (DAMA) in both data and control plane.

The overall architecture of an emulated satellite terminal is depicted in Figure 5.6. At IP level, a traffic monitoring agent processes each packet coming from the user terminals (UTs), according to the DSCP field in the IP header [133]. Then, traffic is shaped and scheduled depending to DiffServ policy. Furthermore, enqueued data are continuously monitored in order to compute, every superframe, how many bytes are stored (*Q(kT)* where $T$ is the superframe duration and $k$ is an integer).

The amount of enqueued bytes is used every T seconds as an input for the *DAMA requester* agent, which is composed of the following modules:

- a *CRA filter* manages the capacity (bytes per superframe) statically assigned to the ST in each superframe (CRA capacity). Such a capacity is assigned at the setup of the emulation session by a CRA manager. Then, the CRA filter returns the amount of enqueued bytes that exceed the CRA capacity ($Q'(kT) \leq Q(kT)$).
- an *RBDC module* receives as input the amount of bytes exceeding the CRA capacity (Q'(kT)), and computes a capacity request on the basis of both the implemented RBDC algorithm (currently equation 3.12 is considered) and the

maximum number of slots dedicated for RBDC requests (set in the configuration phase).

- a *VBDC module* receives as input the amount of bytes exceeding the sum of the CRA capacity and the maximum amount of capacity that can be requested through RBDC policy ($Q''(kT) \leq Q'(kT)$).

- a *Request computation module* collects both RBDC and VBDC requests and send them to the NCC.

Then, packet generation is managed through a *Token Bucked Filter* (TBF), which schedules outgoing packets, accommodated in a FIFO (First-In-First-Out) queue, based on the expenditure of tokens. Tokens correspond to capacity allotted by the NCC and notified through the Terminal Burst Time Plane (TBTP).

#### 5.2.2.3 Gateway/Hub

The Gateway/Hub is the responsible of transmission of data packet towards the satellite terminals over the forward link. Furthermore, it implements functionalities of Network Control Center (NCC), managing the access to the return link, ensuring a common reference for the MF-TDMA transmissions, Network Clock Reference (NCR), dynamically assigning and redistributing bandwidth among satellite terminals. Before sending any data, STs must join the network by communicating (logon) to the NCC both their identity and their DAMA profile. The logon message is sent using out of band control messages. Furthermore, if needed, the NCC can force ST to log off.

Once STs are logged, the gateway/Hub performs the following functions:

- IP router functions to interface the Internet;

- either application "sink" or application client/server;

- *DAMA Controller* functions.

With reference to Figure 5.6, the DAMA controller is composed of four modules:

- a *CRA manager* which keeps track of capacity permanently allocated at each ST. It also performs a Call Admission Control (CAC) so that the capacity statically allotted (CRA capacity) does not exceed the overall capacity. On the other hand, the difference between the overall capacity and CRA capacity is used for dynamic allocation.
- a *DAMA manager* which collects all the DAMA requests coming from STs and assigns capacity available (not statically assigned) on a superframe basis.
- a *FCA (Free Capacity Assignation) module* which arbitrarily decides if and to which ST to assign spare capacity.
- a *Capacity allocator* which creates a "grid of allocation" and computes capacity allotments in terms of tokens to send to STs.

#### 5.2.2.4 Satellite channel

Satellite channel emulation concerns each considered link:

- the forward link (from GW to STs);
- the return link (from STs to GW);
- the link for DVB-RCS signalling.

For each link the following parameters are emulated:

- The propagation delay;
- the bandwidth constraint;
- Propagation conditions in term of packet losses.

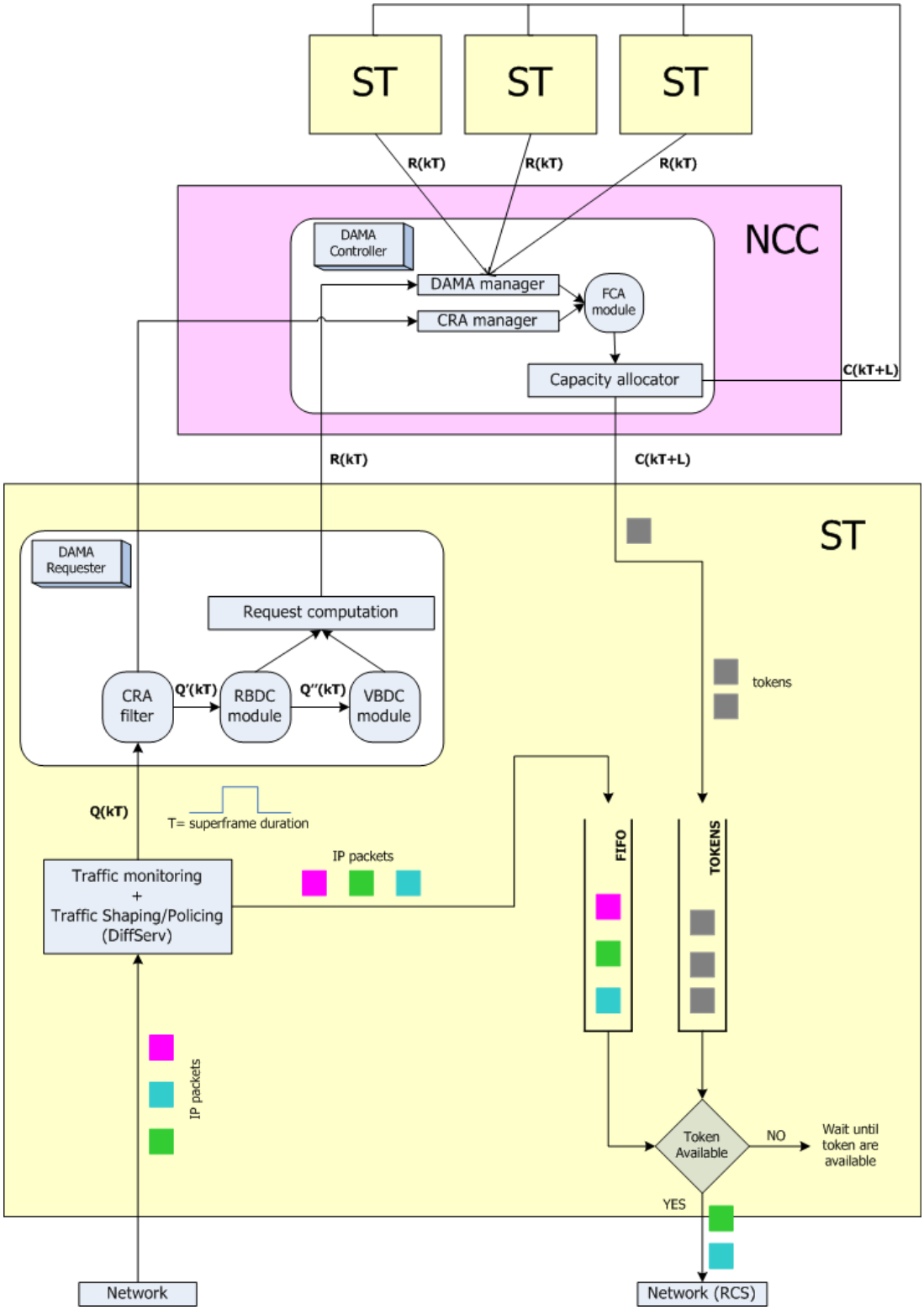


Figure 5.6. Satellite Terminal architecture

# Chapter 6

# Conclusions

This work highlights the increasingly important role of satellite in the current and the future information-based society. The main benefits of satellite communications include high bandwidth, global coverage, and untethered connectivity. Furthermore, the integration of satellites with other systems makes possible the rapid and efficient provision of communication services in challenging scenarios where traditional networks appears inadequate or completely absent. On the other hand, the study and the analysis of transport layer in environments including satellite are demanding tasks, since they require the definition and the implementation of innovative protocols and architecture solutions.

Focusing on point-to-point services, this work addresses the analysis of TCP dynamics over a large set of network architectures based on either a stand-alone satellite segment or an satellite-based integrated network. The main segments, which satellite is integrated to, are HAPS platform, wireless network or MANET. Furthermore, considered scenarios have envisaged combinations of different application programs (i.e., HTTP, FTP, etc.) and lower layer protocols (i.e., IPsec, DAMA, etc.). In each case, the primary goal pursued has been to optimize end-to-end performance by enhancing transport layer protocols. Such enhancements involve both the use of TCP variants (i.e. TCP Westwood) to replace the standard TCP and architectural modifications (i.e. installation of TCP PEP between the end systems). Results show relevant improvements in terms of throughput and channel utilization, with respect to the case standard TCP is used in an end-to-end connection.

Particular attention was dedicated to DVB-RCS-like satellite architectures, where DAMA loops drastically impact on TCP performance. Theoretical analysis, simulations and experiments through emulated DVB-RCS environment (NEP) are allowed to identify characteristics of the TCP-DAMA interaction, and as a consequence to select the best DAMA scheme for the target performance.

Furthermore, some Cross-Layer (CL) interactions have been designed in order to adapt TCP behavior at mechanisms running at other layers, and vice versa. The advantages coming from these inter-layer communication have been successfully checked in the following scenarios: TCP-DAMA in DVB-RCS networks to synchronize DAMA capacity assignment to the TCP window trend , CAC-TCP in HAPS-satellite integrated networks to guarantee an efficient channel utilization, and Mobile IP-TCP in a Wireless-satellite hybrid network to efficient counteract to handover events.

Finally, based on the experience achieved, the design and the implementation of an advanced emulation platform, named SNEP, has been developed aiming to provide a real-time environment to assess and benchmark the performance of both popular internet applications and satellite dedicated applications. Specifically, SNEP reproduces a DVB-S/DVB-RCS satellite access system as far as network, access and resource management layers are concerned. Its innovative contribution, with respect to existing emulator platforms/tools, in bases on both the enhancement of the dynamic emulation approach in order to mutually correlate the emulated DAMA algorithms and the injected traffic, and the possibility to access parameters/statistics of all the protocol layers in a flexible way.

# Bibliography


[1] Y. Hu, and V. Li, *Satellite-based Internet: a Tutorial*, IEEE Communication Magazine, vol. 39, n. 3, 2001, pp. 154-162.

[2] Y. Zhang, D. De Lucia, B. Ryu, and S. Dao, *Satellite Communications in the Global Internet: Issues, Pitfalls, and Potential*, in Proceedings of INET '97, 1997.

[3] E. Lutz, M. Werner, A. Jahn, *Satellite systems for personal and broadband communications*, Springer, Berlin, 2000.

[4] G. Maral, M. Bousquet, *Satellite communications systems: Systems techniques and technology*, ($4^{th}$ ed. Chichester), Wiley, 2003.

[5] EBU/ETSI,*Digital Video Broadcasting (DVB): DVB specification for data broadcasting*, EN 301 192, Jun. 1999.

[6] EBU/ETSI, *Digital Broadcasting Systems for Television, Sound and Data Services; Framing Structure, Channel Coding and Modulation for 11/12 GHz Satellite Services*, ETS 300 421, Aug. 2002.

[7] EBU/ETSI, *Digital Video Broadcasting (DVB); Implementation guidelines for the use of MPEG-2 Systems, Video and Audio in satellite, cable and terrestrial broadcasting applications*, ETR 154, Sep. 1997.

[8] EBU/ETSI, *Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications*, EN 302 307, Jul.2004.

[9] MPEG, *Information Technology - Generic Coding of Moving Pictures and Associated Audio Information. Part 1: Systems MPEG*, ISO/IEC 13818-1, Nov. 1994.

[10] H.D. Clausen, H. Linder and B. Collini-Nocker,*Internet over Direct Broadcast Satellites*, IEEE Communications Magazine, pp. 146-151, Jun. 1999.

[11] ETSI, *Digital Video Broadcasting (DVB); Interaction Channel for Satellite Distribution Systems, DVB-RCS standard*, EN 301 790, Mar. 2003.

[12] ETSI, *Digital Video Broadcasting (DVB); Interaction Channel for Satellite Distribution Systems, Guidelines for the use of EN 301 790*, TR 101 790, V. 1.2.1, Jua. 2003.

[13] M. Luglio, A. Saitto, *Security of Satellite Networks*, chapter in H. Bidgoli (Ed), "The Handbook of Information Security", John Wiley & Sons, Inc., 2006, Hoboken, N.J., Vol. 1, pp. 754-771.

[14] M. P. Howarth, S. Iyengar, Z. Sun and H. Cruickshank, *Dynamics of key management in secure satellite multicast*, IEEE Journal on Selected Areas in Communications, Vol. 22, n. 2, pp. 308-318.

[15] W. Stevens, *TCP/IP Illustrated. Vol. 1*, Ed. Addison Wesley, Reading, UK, 1994.

[16] J. Postel, *Transmission control Protocol*. Internet RFC 793, 1981.

[17] J. Postel, *User Datagram Protocol*, Internet RFC 768, 1980.

[18] M. Luglio, C. Roseti, and M. Gerla, *The Impact of Efficient Flow Control and OS Features on TCP Performance over Satellite Links*, ASSI Satellite Communication Letter (Sat-Comm Letter), 9th edition, special issue on Multimedia Satellite Communication, vol. III, n. 1, 2004, pp. 1-9.

[19] W. Stevens, *TCP Slow Start, Congestion Avoidance, Fast retransmit, and Fast recovery Algorithms*, Internet RFC 2001, 1997.

[20] V. Jacobson, *Congestion Avoidance and Control*, in Proceedings of ACM SIGCOMM '88 Conference, 1988, pp. 314-329.

[21] V. Paxson, M. Allman, *Computing TCP's Retransmission Timer*, RFC 2988, Nov. 2000.

[22] M. Allman, W. Stevens, *TCP congestion control*. IETF RFC 2581, Apr. 1999.

[23] R. Braden, *Requirements for Internet Hosts – Communication Layers*, STD 3, RFC 1122, October 1989.

[24] I. Jacobs, R. Binder, E. Hoversten, *General Purpose Packet Satellite Networks*, in Proceedings of the IEEE, Vol. 66, n. 11, pp. 1448-1467, 1978.

[25] C. Partridge, T. Shepard, *TCP Performance over Satellite Links* IEEE Network, vol. 11, n. 5, pp. 44-49, 1997.

[26] M. Allman, C. Hayes, H. Kruse, S. Osterman, *TCP Performance over Satellite Links*, $5^{th}$ Int. Conference on Telecommunication Systems Modeling and Design, pp. 1-13, 1997.

[27] T. Lakshman, and U. Madhow, *The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss*, IEEE/ACM Transactions on Networking, vol. 5, n. 3, 1997, pp. 336-350.

[28] G. Xylomenos, G. C. Polyzos, P. Mahonen, and M. Saaranen, *TCP Performance Issues over Wireless Links*, IEEE Communication Magazine, vol. 39, n. 4, 2001 pp. 52-58.

[29] C. Charalambous, V. Frost, and J. Evans, *Performance of TCP Extensions on Noisy High BDP Networks*, IEEE Letters on Communications, vol. 3, n. 10, 1999, pp. 294-299.

[30] URL: http://www.isi.edu/nsnam/ns/.

[31] M. Karaliopoulos, R. Tafazzoli, B. G. Evans, *Providing Differentiated Service to TCP Flows Over Bandwidth on Demand Geostationary Satellite Networks*, IEEE Journal on Selected Areas in Communications, Vol. 22, No. 2, Feb. 2004.

[32] R. Jain, W. Hawe, D. Chiu, *A Quantitative measure of fairness and discrimination for resource allocation in Shared Computer Systems*, DEC-TR-301, Sep. 1984.

[33] H. Balakrishnan, V. Padmanabhan, R. Katz, *The effects of Asymmetry on TCP Performance*, in Proceedings of ACM SIGCOMM Conference, 1997, pp. 175-87.

[34] E. Lutz, C. Cygan, M. Dippold, F. Dolainsky, W. Papke, *The Land Mobile Satellite Channel - Recording, Statistics and Channel Model*, IEEE Transactions on Vehicular Technology, Vol. 40, n. 2, 1991, pp. 375-386.

[35] E. Lutz, A Markoff Model for Correlated Land Mobile Satellite Channels, International Journal of Satellite Communications, vol. 13, 1996, pp. 333-339.

[36] P. Loreti, M. Luglio, *Satellite Diversity: a Technique to Improve Link Performance and Availability for Multicoverage Constellations*, chapter in P. Stavroulakis (Ed) "3rd Generation Mobile Communication Systems, UMTS and IMT-2000", Springer Verlag, 2001, Berlin.

[37] V. Jacobson, R. Braden, D. Borman, *TCP Extensions for High Performance*, Internet RFC 1323, 1992.

[38] T. Berners-Lee, R. Fielding, H. Nielsen, *Hypertext Transfer Protocol - HTTP/1.0*, RFC 1945, May 1996.

[39] R. Fielding, J.Gettys, J. C. Mogul, H. Frystyk, T. Berners-Lee, *Hyper- text Transfer Protocol - HTTP/1.1*, RFC 2068, Jan. 1997.

[40] J. Heidemann, *Performance Interactions Between P-HTTP and TCP Implementations* Computer Communication Review, Vol. 27, N. 2, Apr. 1997, pp. 65-73.

[41] URL: http://dast.nlanr.net/Projects/Iperf/

[42] J. Mogul, S. Deering, *Path MTU discovery*, Internet RFC 1191, 1990.

[43] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, J. Crowcroft, *Forward Error Correction (FEC) Building Block*, IETF RFC3452, 2002.

[44] B. Sklar, *Digital Communications*, $2^{nd}$ edition, Prentice Hall, 2001.

[45] J. Stadler, *A Link Layer protocol for Efficient Transmission of TCP/IP via satellite*, in Proceedings of IEEE MILCOM '97, vol. 2, pp. 723-727, 1997.

[46] K. Ramakrishnan, S. Floyd, D. Black, *The Addition of Explicit Congestion*

*Notification (ECN) to IP*, RFC 3168, Sep. 2001.

[47] R. Fox, *TCP big window and NAK options*, RFC 1106, Jun. 1989.

[48] V. Jacobson, *Compressing TCP/IP headers for low-speed serial links*, RFC 1144, Feb. 1990.

[49] R. Braden, *Extending TCP for Transactions – Concepts*, RFC 1379, Nov. 1992.

[50] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, *TCP Selective Acknowledgment Options*, RFC 2018, Oct. 1996

[51] M. Allman, S. Floyd, C. Partridge, *Increasing TCP's Initial Window*, RFC 2414, Sep. 1998.

[52] M. Allman, D. Glover, L. Sanchez, *Enhancing TCP Over Satellite Channels using Standard Mechanisms*, RFC 2488, Jan. 1999.

[53] M. Allman, *TCP Congestion Control with Appropriate Byte Counting (ABC)*, RFC 3465, Feb. 2003.

[54] J. Nagle, *Congestion Control in IP/TCP Internetworks*, RFC 896, FACC, Jan. 1984.

[55] S. Floyd, T. Henderson, *The NewReno Modification to TCP's Fast Recovery algorithm*, Internet RFC 2582, 1999.

[56] K. Fall, S. Floyd, *Simulation-based Comparisons of Tahoe, Reno, and SACK TCP*, ACM Computer Communications Review, vol. 26, n. 3, pp. 5-21, 1996.

[57] R. Braden, *T/TCP – TCP Extensions for Transactions Functional Specification*, RFC 1644, Jul. 1994.

[58] S. Floyd, *HighSpeed TCP for Large Congestion Windows*, RFC 3649, Experimental, Dec. 2003.

[59] E. Souza, D.A. Agarwal, *A HighSpeed TCP Study: Characteristics and Deployment Issues*, LBNL Technical Report LBNL-53215.

[60] L. S. Brakmo, L. L. Perterson, *TCP Vegas: End-to-End Congestion Avoidance on a Global Internet*, IEEE Journal on Selected Areas in Communications, vol. 13, n. 8, pp. 1465-1480, 1995.

[61] J. Cheng, D. X. Wei, and H. Steven, *TCP FAST: motivation, architecture, algorithms, performance*, In Proceedings of IEEE Infocom, Mar. 2004.

[62] I. F. Akyildiz, G. Morabito, S. Palazzo, TCP-Peach: a new congestion control scheme for satellite IP networks, IEEE/ACM Transactions on Networking, vol. 9, n. 3, pp. 307-321, 2001.

[63] C. Caini, R. Firrincieli, *A new transport protocol proposal for Internet via satellite: the TCP Hybla*, In Proceedings of ESA ASMS conference 2003, Jul. 2003.

[64] C. Caini and R. Firrincieli, "TCP Hybla: a TCP Enhancement for Heterogeneous Networks", International Journal of Satellite Communications and Networking, vol. 22, n. 5, pp. 547-566, Sep. 2004.

[65] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, R. Wang *TCP Westwood: End-to-end Congestion Control for Wired/Wireless Networks*, Wireless Networks Journal, n. 8, pp. 467-479, 2002.

[66] R. Wang, M. Valla, M. Y. Sanadidi, M. Gerla, *Adaptive Bandwidth Share Estimation in TCP Westwood*, In Proc. GLOBECOM'02, pp.2604-2608.

[67] M. Gerla, B. K. F. Ng, M. Y. Sanadidi, M. Valla, R. Wang, *TCP Westwood with adaptive bandwidth estimation to improve efficiency/friendliness tradeoffs*, Journal of computer communications, Vol. 27, N. 1, Jan. 2004.

[68] R. Wang, G Pau, K. Yamada, M. Y. Sanadidi, M. Gerla, *TCP Startup Performance in Large Bandwidth Delay Networks*, INFOCOM 2004, Hong Kong, March 2004, pp.796-805.

[69] V. Tsaoussidis, C. Zhang, *TCP-Real: receiver-oriented congestion control*, International Journal of Computer and Telecommunications Networking, Vol. 40, N. 4, pp. 477-497, Nov. 2002.

[70] V. Tsaoussidis, H. Badr, R. Verma, *Wave and Wait Protocol: an energy-saving transport protocol for mobile IP-devices*, in proceedings of the $7^{th}$ IEEE International Conference on Networks Protocols, Toronto, Canada, 1999.

[71] J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby, *Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations*, IETF RFC 3135, Jun. 2001.

[72] T. R. Henderson, R. H. Katz, *Transport protocols for Internet-compatible satellite Networks*, IEEE Journal on Selected Areas Communication, vol.17, No.2, pp.326-34, Feb.1999.

[73] M. Marchese, *TCP/IP-Based Protocols Over Satellite Systems: A Telecommunication Issue*, in Reliability, Survivability and Quality of Large Scale Telecommunication Systems, P. Stavroulakis, Ed. Chichester: Wiley, pp. 167198, 2003.

[74] S. Fu, M Atiquzzaman, *SCTP: State of the Art in Research, Products, and Technical Challenger*, IEEE Communication Magazine, vol. 43, N. 4, pp 64 -76, Apr. 2004.

[75] URL: http://www.ccsds.org/

[76] *Space Communications Protocol Specification (SCPS)-Transport Protocol (SCPS-TP)*, Recommendation for Space Data System Standards, CCSDS 714.0-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, May 1999.

[77] *Space Communications Protocol Specification (SCPS)-Transport Protocol (SCPS-TP)*. Recommendation for Space Data System Standards, CCSDS 714.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, Oct. 2006.

[78] K. Ramakrishnan , S. Floyd, D. Black, *The Addition of Explicit Congestion Notification (ECN) to IP*, RFC 3168, Sep. 2001.

[79] R.Sanders, A.Weaver, *The Xpress transfer protocol (XTP): A tutorial*, SIGCOMM Comput. Commun. Rev. 20, 5, pp.67-80, Oct. 1990.

[80] M. Mathis, J. Semke, J. Mahdavi, T. Ott, *The macroscopic behavior of the TCP congestion avoidance algorithm*, ACM Computer Communication Review, vol. 27, no. 3, Jul. 1997.

[81] T. Lakshman, U. Madhow, *The performance of TCP/IP for networks with high*

*bandwidth-delay products and random loss*, IEEE-ACM Transaction on Networking, vol. 5, no. 3, pp.336-350, Jun. 1997.

[82] T. C. Tozer and D. Grace, *High-altitude platforms for wireless communications*, Electronics & Communication Engineering Journal, vol.13, no.3, pp.127-137, Jun 2001.

[83] D. Avagnina, F. Dovis, A. Ghiglione, P. Mulassano, *Wireless networks based on high-altitude platforms for the provision of integrated navigation/communication services*, IEEE Communications Magazine, vol.40, no.2, pp.119-125, Feb 2002.

[84] E. Lutz, D. Cygan, M. Dippold, F. Dolainsky, W. Papke, *The Land Mobile Satellite Channel - Recording, Statistics and Channel Model*, IEEE Transaction on Vehicular Technology, vol. 40, May 1991.

[85] E. Corazza, F. Vatalaro, *A Statistical Model for Land Mobile Satellite Channels and its Application to Nongeostationary Orbit Systems*, IEEE Transaction on Vehicular Technology, vol. 43, pp.738-741, Aug. 1994.

[86] L. Baldantoni, H. Lundqvist, G. Karlsson, *Adaptive End-to-End FEC for Improving TCP Performance over Wireless Links*, ICC 2004, Jun 2004.

[87] H. Lundqvist, G. Karlsson, *TCP with End-to-End Forward Error Correction*, International Zurich Seminar on Communications, IZS 2004, Feb 2004.

[88] S. Karapantazis, N. F. Pavlidou, *Broadband communications via high-altitude platforms: a survey*, IEEE Communications Surveys & Tutorials, Vol. 7, no. 1, pp. 2-31, 2005.

[89] B.M. Epstein, M. Schwartz, *Predictive QoS-based admission control for multiclass traffic in cellular wireless networks*, IEEE Journal on Selected Areas in Communications, Volume 18, Issue 3, March 2000 pp. 523 - 534.

[90] *Recommendation ITU-R P.681-6.*

[91] J.L. Cuevas-Ruiz, and J.A. Delgado-Penin, *Channel model based on semi-Markovian processes. An approach for HAPS systems*, $14^{th}$ International Conference on Electronics, Communications and Computers, pp. 52-56, Feb. 2004.

[92] M. Sooriyabandara, G. Fairhurst, *Dynamics of TCP over BoD satellite Networks*, International Journal of Satellite Communications and Networking, Vol. 21, No. 4-5, pp. 427-449, Jul. 2005.

[93] Qi Wang, M.A. Abu-Rgheff, *Cross-Layer Signalling for Next- Generation Wireless Systems*, Wireless Communications and Networking, Vol. 2, pp. 1084-1089, Mar. 2003.

[94] G. Acar, C. Rosenberg, *Algorithms to Compute for Bandwidth on Demand Requests in a Satellite Access Unit*, in Proceedings of $5^{th}$ Ka-band Utilization Conference, pp. 353-360, 1999.

[95] C. Perkins, *IP Mobility Support*, RFC 2002 , Oct. 1996.

[96] C. Perkins, *IP Mobility Support for IPv4*, RFC 3344, Aug. 2002.

[97] H. Soliman et al., *Hierarchical Mobile IPv6 Mobility Management (HMIPv6)*, RFC 4140, Nov. 2005.

[98] W. Hansmann, M. Frank, *On Things to Happen During a TCP Handover*, $28^{th}$ Annual IEEE International Conference on Local Computer Networks (LCN'03), Oct. 2003.

[99] M. Zonoozi, P. Dassanayake, M. Faulkner, *Optimum Hysteresis, Signal Averaging Time and Handover Delay*, IEEE Vehicular Technology Conference, Phoenix, AZ, USA , vol. 1, pp 310-313, Mar. 1997.

[100] J-O. Vatn *An experimental study of IEEE 802.11b handover performance and its effect on voice traffic*, technical paper, Jul. 2003. (citeseer.ist.psu.edu/vatn03experimental.html)

[101] V. T. Raisinghani and S. Iyer, *ECLAIR: An Efficient Cross Layer Architecture for wireless protocol stacks*, in Proceedings of World Wireless Congress (WWC04), San Francisco, USAMay 2004.

[102] V. Kawadia, P. R. Kumar, *A cautionary perspective on cross layer design*, IEEE Wireless Communications, Vol. 12, N. 1, pp. 3-11, Feb. 2005.

[103] P. Karn and C. Partridge, *Improving Round-Trip Time Estimates in Reliable*

*Transport Protocols*, ACM Transactions on Computer Systems, Vol. 9, N. 4, pp.364-373, Nov. 1991.

[104] Elsacom web site, URL: http://www.elsacom.it.

[105] Globalstat web site, URL: http://www.globalstar.com.

[106] Hughes web site, URL: http://www.hughes.com.

[107] Telespazio web site, URL: http://www.telespazio.it.

[108] A. Borrelli, F. Mazzenga, C. Monti, M. Vari, *Channel Models for IEEE 802.11b Indoor System Design*, IEEE International Conference on Communications, Vol. 6, pp. 3701 - 3705, Jun. 200.

[109] R. Atkinson, *Security architecture for the Internet Protocol*, RFC 2401, Nov. 1998.

[110] Y. Zhang, *A multilayer IP security protocol for TCP performance enhancement in wireless networks*, IEEE Journal on Selected Areas in Communications, Vol. 22, n. 4, pp. 767-776, May 2004.

[111] R. Atkinson, *IP Authentication Header*, RFC 2402, Nov. 1998.

[112] R. Atkinson, *IP Encapsulating Security Payload (ESP)*, RFC 2406, Nov. 1998.

[113] J. Stepanek, A. Razdan, A. Nandan, M. Gerla, M. Luglio, *The Use of a Proxy on Board the Satellite to Improve TCP Performance*, IEEE Global Telecommunications Conference, GLOBECOM '02, Vol. 3, pp. 2950-2954, 2002.

[114] D. Velenis, D. Kalogeras, and B. Maglaris, *SaTPEP: a TCP Performance Enhancing Proxy for Satellite Links*, $2^{nd}$ International IFIPTC6 Networking Conference, May 2002.

[115] L. Rizzo, *Dummynet, A simple approach to the evaluation of protocols*, in ACM Computer Communication Review, Vol. 27, n. 1, pp. 31-41, 1997.

[116] M. Carson, D. Santay, *NIST Net: A Linux-Based Network Emulation Tool*, in ACM Computer Communication Review, Vol. 33, n. 3, Jul. 2003.

[117] URL: http://irg.cs.ohiou.edu/one/.

[118] S. Hemminger, *Network Emulation with NetEm*, In Proceedings of LCA, Apr.

2005.

[119] B. Hubert, at al., *Linux Advanced Routing & Traffic Control*, http://lartc.org/.

[120] I. Yeom, N. Reddy, *ENDE: An End-to-end Network Delay Emulator Tool for Multimedia Protocol Development*, Journal on Multimedia Tools and Applications, Vol. 14, pp. 269-296, 2001.

[121] D.B. Ingham, G.D. Parrington, *DelayLine: a wide-area network emulation tool*, USENIX, Computing Systems, Vol. 7, n. 3, pp. 131-332, 1994.

[122] D. Herrscher, K. Rothermel, *A Dynamic Network Scenario Emulation Tool*, in proceedings of the $11^{th}$ International Conference on Computer Communications and Networks ICCCN'02, Miami, pp. 262-267, 2002.

[123] T. Perennou, E. Conchon, L. Dairaine, M. Diaz, *Two-Stage Wireless Network Emulator*, Workshop on Challenges of Mobility (WCM 2004), Aug. 2004.

[124] C. Caini, R. Firrincieli, D. Lacamera, *A Linux Based Multi-TCP Implementation for Experimental Evaaluation of TCP Enhancement*, 2005 International Symposium on Performance Evaluation of Computer and Telecommunication Sytems (SPECTS'05), Philadelphia, USA, pp. 875-883, Jul. 2005.

[125] URL: http://www.wirelessfuture.it.

[126] M. Marchese, M. Perrando, *A packet-switching satellite emulator: A proposal about architecture and implementation*, IEEE ICC 2002, vol. 25, n. 1, pp. 3033-3037, Apr. 2002.

[127] Free Software Foundation, Inc., *GNU General Public License*, v. 2, Boston, MA, USA, Jun. 1991. URL://www.gnu.org/copyleft/gpl.html.

[128] W. Almesberger, *Linux Traffic Control-Implementation Overview*, ftp://lrcftp.epfl.ch/pub/people/almesber/pub/tcio-current.ps.gz, Technical Report SSC/1998/037, EPFL, Nov. 1998.

[129] *Differentiated Services on Linux*, http://diffserv.sourceforge.net.

[130] B. Hubert, *Linux Advanced Routing & Traffic Control HOWTO*, http://ds9a.nl/2.4Networking/.

[131] URL:http://www.ipsec-howto.org/.

[132] L. A. Larzon, M. Degermark, S. Pink, L. E. Jonsson, G. Fairhurst, *The Lightweight User Datagram Protocol (UDP-Lite)*, IETF RFC 3828, 2004.

[133] K. Nichols, S. Blake, F. Baker, D. Black, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, RFC 2474, Dec. 1998.