Optimising Internet Access over Satellite Backhaul

Paul Davern, Noor Nashid Islam, Cormac J Sreenan
Mobile and Internet Systems Laboratory,
Department of Computer Science
University College Cork, Cork, Ireland.
p.davern@cs.ucc.ie

March 14, 2012

Abstract

Application level protocols such as the Hypertext Transfer protocol (HTTP) and transport layer protocols such as Transmission Control Protocol (TCP) have performance issues on wired networks. These issues are greatly magnified by the particular conditions of satellite systems. For example, a user experiences long delays e.g. during web browsing, due to the impact that the latency of the satellite link has on HTTP and TCP. In this paper we present the system architecture of the BAChaul OPTimization (BACOPT) project. The main contribution of the BACOPT research is a new split-proxy architecture for data optimisation over resource-constrained backhaul links. The BACOPT architecture is designed to optimise application level protocols such as HTTP and to manage the transport layer while realising target QoS requirements.

Keywords: Satellite, Optimisation, HTTP, TCP, Back-haul.

1 Introduction

The focus of the BAChaul OPTimization (BACOPT) project is to research and develop technologies designed specifically to reduce the required back-haul capacity in current and emerging mobile networks. Back-haul capacity represents a significant operational cost in developed markets with the exponential increase in the bandwidth requirement by the end user and content rich applications [Sadre, 2008]. Further, a significant percentage of web users in emerging markets are accessing the Internet through resource limited satellite connections.

Figure 1 shows a general architecture for such a satellite Internet access system including

- The remote Satellite Gateway (SG), which acts as an aggregation point for remote community traffic and the ground Satellite Gateway (SG), which provides an aggregation point for services hosted in the core network.
- Satellite, whose main role is to forward the traffic across attached satellite terminals, i.e. gateways.

In this architecture, SGs usually represents a bottleneck where functionality such as traffic monitoring, traffic classification, policing, packet scheduling, and resource management needs to be implemented to realise target quality of service (QoS) requirements while minimizing the operational costs. The operation of the satellite Internet access system is challenged by the intrinsic characteristics of a satellite such as large propagation delay, bursty traffic, wireless transmission errors, significant signal attenuation, heterogeneous application QoS requirements, and other similar factors. The impact of these characteristics on the system performance can be reduced by proper design of the protocol stack layer.

In this paper, we present a summary of the BACOPT project's goals and achievements for additional details see [Davern et al., 2011b] [Davern et al., 2011a][Davern et al., 2011c]. The main contribution of the BACOPT research is a new split-proxy architecture for data optimisation over resource-constrained

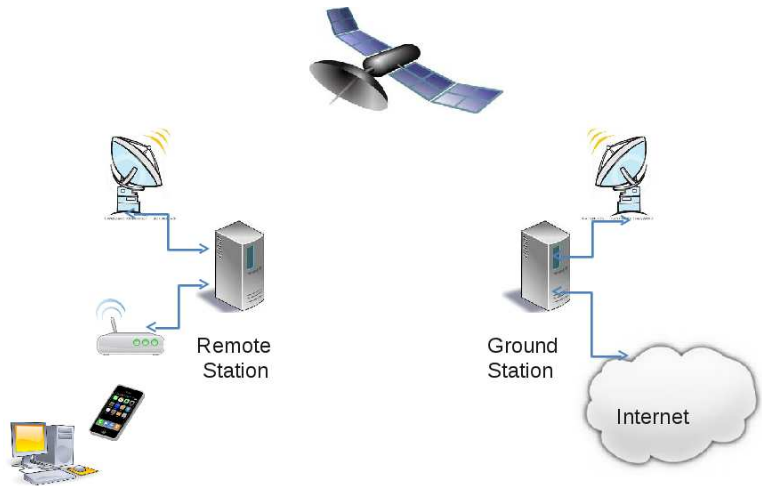

Figure 1: Satellite Internet access System

backhaul links. The BACOPT architecture is designed to optimise application level protocols such as HTTP and to manage the transport layer while realising target QoS requirements.

The rest of the paper is organized as follows. Section 2 is dedicated to background and related work. In Section 3, we present the BACOPT framework. In Section 4 we present the Link Manager for bandwidth management. In Section 5 we present our HTTP acceleration mechanism HTTPEP. In Section 6 we present an evaluation of HTTPEP. Finally, conclusions are presented in Section 7.

2 Background and related work

In this section we provide background as to the impact that satellite systems have on application layer protocols and on the transport layer.

2.1 Impact of Satellite Systems on application layer protocols

The Hypertext Transfer protocol (HTTP) is a stateless, transactional protocol that governs content exchange between web clients and servers. HTTP features a sequential operation that delays the retrieval time of embedded web *resourc*es. For example, when a client on the end-user's machine issues an HTTP GET request for a particular web page, the web server replies with the *base* HTML page containing references for other *nested* resources required by the client to display the page to the end-user. These resources are requested through additional HTTP GET requests over possibly new TCP connections opened by the client to the web server.

The satellite system poses severe challenges for application level protocols such as HTTP. For example, the inherent latency caused by the sequential operation of HTTP on wired networks is greatly magnified on satellite systems. In order to expedite web browsing in today's Internet, especially for users who access via resource constrained links, extensive research [Caini et al., 2009] has been conducted and various approaches[Chakravorty et al., 2005] and products [Riverbed, 2011] have been proposed.

2.2 Impact of Satellite Systems on the Transport Layer

In the transport layer, Transmission Control Protocol (TCP) also introduces several performance issues, which magnify the latency in the application layer [Kota and Marchese, 2003]. Each TCP connection requires a three-way handshake (SYN-SYN.ACK-ACK) for session establishment before sending any data over a TCP connection. This behavior not only delays the data transmission but also introduces a data transmission overhead when opening multiple TCP connections. TCP depends on a reactive congestion control mechanism by which a sender should conservatively increase its transmission window

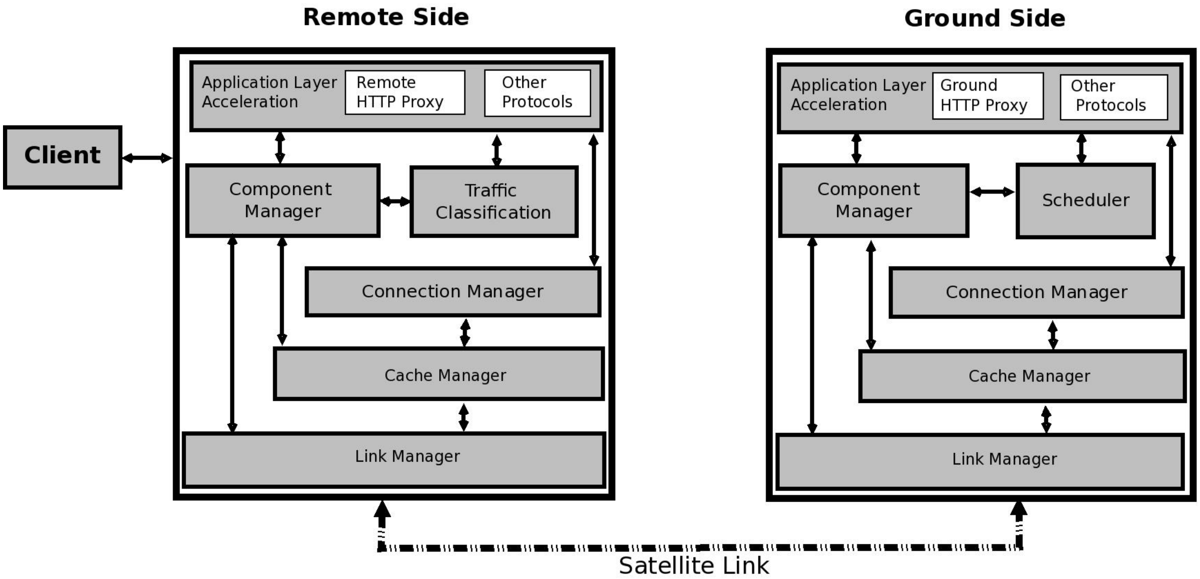

Figure 2: Logical Overview of BACOPT framework

during the slow start phase. This behavior leads to slowing down the TCP session and under-utilization of network resources [Allman et al., 1997].

These issues with TCP on wired networks are greatly accentuated when it comes to high latency links such as broadband satellite systems. The long delay of this type of link reduces the TCP congestion window (CWND) growth rate, which is dependent on the link RTT. Whereas, TCP Hybla [Carlo and Firrincieli, 2004] was specifically designed for high latency links and its congestion algorithm is not based on the RTT of the link, but rather on the RTT of a reference connection. Different mechanisms [Caini et al., 2009] are proposed to improve TCP performance in satellite systems. One particular mechanism is the use of TCP performance enhancing proxies (PEPs) [Border et al., 2001]. TCP PEPs adopt different techniques to enhance TCP performance over high latency links including ACK spacing, local ACK, local transmission, header compression, payload compression and priority-based multiplexing.

3 The BACOPT framework

Figure 3 gives a logical overview of the system components of the BACOPT framework. The framework has a generic architecture split across the two ends of the back-haul link to improve the link utilization, while maintaining the quality of service to the end user.

1. The framework is designed to accommodate a number of components for accelerating application level protocols. Here we have included the generic structure of such a component, which consists of a proxy for the protocol on the remote and ground nodes. In section 5we give a description of our HTTP acceleration mechanism. Further resource-demanding application level services are to be accommodated into BACOPT framework such as media streaming and peer-to-peer (p2p) mandates the need for intelligent backhaul traffic optimization techniques such as caching and function splitting.

2. The Component manager is responsible for ensuring QoS between services and users. For example, under system load the Component manager may decide that a more aggressive policy of compression is required – appropriate messages are transmitted to the compression engine to enable this.

3. The traffic classification component is responsible for identifying the services that are being used by the end users on the system.

4. The Cache manager is responsible for synchronizing the cache between the remote and ground nodes.

5. The Connection Manager maximizes the use of the available bandwidth while providing fairness between the various active TCP streams.

6. The scheduler interfaces to the connection manager to provide fairness to the end-users.

7. The link manager and service differentiation component for fixed bandwidth links is described in section 4. This component ensures efficient handling for different services according to predefined quality of service constraints.

4 Link Manager

The link manager is responsible for bandwidth management and enforcing traffic prioritization by allowing various traffic streams to be handled differently. This goal is achieved by creating different DiffServ queues serving a set of corresponding traffic streams [Ronga et al., 2003]. The following traffic per-hop-behaviour (PHB) are defined

- Voice Traffic is routed to the *expedited forwarding* queue (EF).
- Classified data streams from the *traffic classifier* are routed to the *assured forwarding* queue (AF). For example web browsing traffic may be assigned a higher priority that FTP.
- All other traffic is routed to the *default* queue (DE) which usually corresponds to best effort.

The Link Manager defines an event driven interface to the other components in the BACOPT framework. For example, when the Link Manager receives the NEW_VOICE_CALL event, the Link Manager dedicates a configurable amount of bandwidth (e.g. 8kbps) for the new call. On Receiving the VOICE_CALL_RELEASE event, the allocated resources are released.

Different system components, such as HTTPEP use specific port-ranges for its traffic. These port-ranges are defined during the Link Manager configuration and corresponding packet classification rules are defined to forward different traffic streams to the appropriate queue.

5 HTTP Performance Enhancing Proxy (HTTPEP)

Figure 2presents the system components of our HTTP PEP (HTTPEP) [Davern et al., 2011b]. HTTPEP has a “split HTTP proxy” architecture by which the functionality is distributed between two HTTP proxy nodes implemented at both ends of the satellite link. The remote proxy (ROptProxy) appears as an HTTP proxy to the end user web browser at the remote site. The ground proxy (GOptProxy) acts as a typical web client to web servers. The two proxies split the HTTP protocol at the remote side proxy, re-issues it at the ground proxy and reconstructs it again at the remote proxy. The two proxies communicate the web content using our novel bundling mechanism. HTTPEP transforms a GET request for a base web page from the end-user so that the nested web resources in that base page are streamed to the remote site.

5.1 Operational Overview

Referring to figure 2, to serve a HTTP web page request, the system operates as follows:

1. When a new web page is required at a client, an initial GET request is sent by the browser for this page (termed the base page).

2. The remote HTTP Proxy (ROptProxy) intercepts this GET request and forwards the URL to the Ground HTTP Proxy (GOptProxy). GOptProxy fetches the base page associated with the URL from the origin server and returns this back to the Scheduler/Compression Engine. The base-HTML is subsequently transferred to the remote side by the scheduler.

3. GOptProxy parses the base page and finds the set of web resources that are embedded in that page. GOptProxy then invokes a set of worker threads to fetch these nested resources. After fetching a resource, a worker thread passes the resource to Scheduler/Compression Engine, which subsequently transfers the resource to the remote site.

4. When the base HTML is received at the remote side, ROptProxy parses it to determine which resources will be on the way from the ground. Then it passes the base HTML to the client.

5. When the client makes subsequent requests for the associated resources of the base HTML, ROptProxy determines if the resources are on the way from the ground side. If so, the request from the client is delayed until the resource arrives. Otherwise, this request is considered to be a new base page request and the system executes the steps outlined above.

5.2 HTTPEP Optimisations

The HTTPEP split architecture optimizes the web page retrieval in several different ways:

- The expensive three-way handshake over the satellite link is eliminated as the TCP connections to the web server are made at the ground site.

- The initial GET for a particular base HTML causes a bundle of GET requests for its nested resources to be issued at the ground site. The associated resources are then transferred from the ground to the remote site. The subsequent GET requests for nested resources within the base HTML by the client do not cross the high latency link but are served locally. This mechanism overcomes the sequential operation of HTTP.

- The persistent connections from the ground to the remote sites allow TCP to build up its congestion window and so the effects of the slow start could be mitigated.

- The compression assists in reducing the amount of traffic crossing the satellite link.

6 Evaluation of the HTTPEP component

The evaluation took place on a test bed consisting of two PCs with Intel Core 2 Duo CPU E4700 2.60 GHz, running Fedora 10, one running ROptProxy and the other running GOptProxy. The ground and remote nodes, are connected using a direct 1Gbps Ethernet connection over which the satellite link behavior is emulated using *tc* commands in Linux. That is, the data rate is limited by default to 256Kbps and a 300ms one way delay is introduced in each direction. A client PC was connected to the remote node running HTTP-AE.

For the results that are shown, the URLs point to a set of pages that contain the same text and a number of embedded images from Caltech 101 image dataset [Group, 2010]. The number of images in the set of pages varies from 1 to 30. The images are all JPEGs and have an average size of 10K. These pages are served from a web-server co-located in the ground node. All results were determined to fall within a confidence interval of 95%.

Figure 3 plots the traffic crossing the satellite link in bytes versus the number of embedded resources for different operating modes including normal operation (i.e. a direct connection between the browser and the web-server without an intermediate proxy), HTTPEP without compression (HTTPEP), HTTPEP with compression of resources (HTTPEPC). The figure shows a marginal saving in terms of number of bytes crossing the satellite link for the HTTPEP solution in comparison to normal operational mode. This byte reduction is due to several factors. First the elimination of TCP signaling for connection establishment and release. Second the compression of the HTTP headers. Third, the reduction in application layer signaling in that embedded resources are not explicitly requested by the remote site but are automatically transmitted.

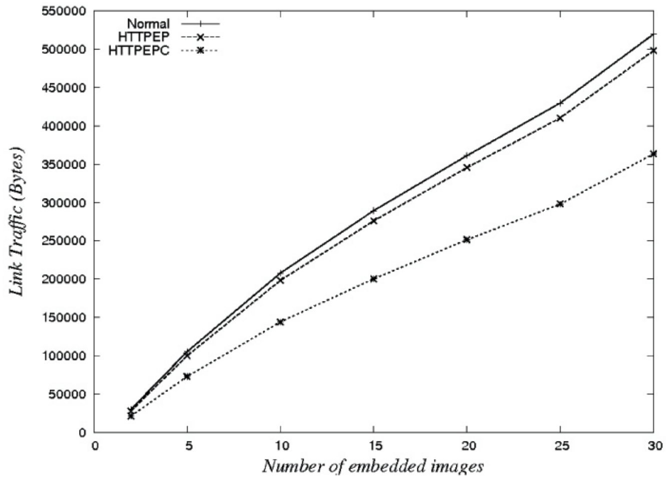

Figure 3: Traffic crossing the link in bytes

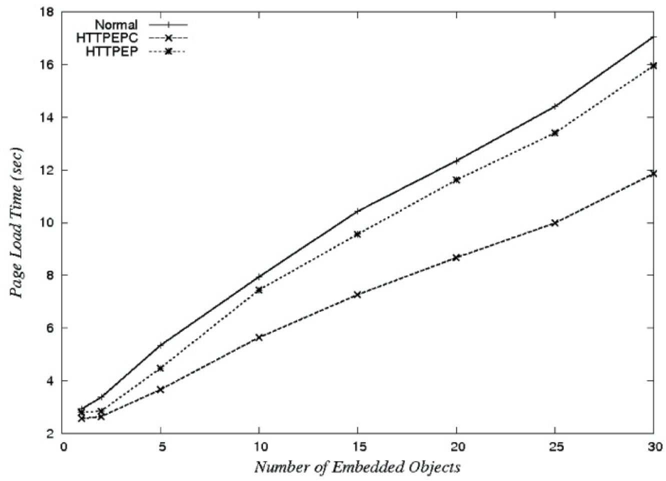

Figure 4: Average page load delay

Figure 4 shows an average page load delay reduction of 27% using compression (HTTPEPC) in comparison to an average saving of 10% without compression over the normal operation (i.e. a direct connection between the browser and the web-server without an intermediate proxy).

7 Conclusions

The main goals and achievements of the BACOPT project are: the design of a system architecture for Back-haul Optimisation in Heterogeneous Wireless Access Networks; the design of a HTTP acceleration technique for high latency links; the demonstration a successful improvement of user perceived latency and a saving in bandwidth usage, for Web access over high latency links; the design of a tool for evaluating HTTP performance. Future work will focus on optimisations for other traffic types such as video and P2P and considering other forms of backhaul links

Acknowledgments

This work is supported by Enterprise Ireland Grant Number IP/2009/0026 and by Altobridge. The authors wish to acknowledge the contribution of Ahmed Zahran to this work.

References

[Allman et al., 1997] Allman, M., Hayes, C., Kruse, H., and Osterman, S. (1997). TCP Performance over satellite links. In *5th International Conference on Telecommunication Systems*.

[Border et al., 2001] Border, J., Kojo, M., Griner, J., Montenegro, G., and Shelby, Z. (2001). RFC3135: Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations.

[Caini et al., 2009] Caini, C., Firrincieli, R., and Lacamera, D. (2009). Comparative Performance Evaluation of TCP Variants on Satellite Environments. *Communications, 2009. ICC '09. IEEE International Conference on*, pages 1 –5.

[Carlo and Firrincieli, 2004] Carlo and Firrincieli, R. (2004). TCP Hybla: a TCP enhancement for heterogeneous networks. *International Journal of Satellite Communications and Networking*, 22.

[Chakravorty et al., 2005] Chakravorty, R., Clark, A., and Pratt, I. (2005). Optimizing web delivery over wireless links: design, implementation, and experiences. *Selected Areas in Communications, IEEE Journal on*, 23(2):402 – 416.

[Davern et al., 2011a] Davern, P., Nashid, N., Sreenan, C. J., and Zahran, A. H. (2011a). Httpep: a http performance enhancing proxy for satellite systems. *IJNGC*, 2(3).

[Davern et al., 2011b] Davern, P., Nashid, N., Zahran, A., and Sreenan, C. J. (2011b). HTTP Acceleration over High Latency Links. In *New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on*, pages 1 –5.

[Davern et al., 2011c] Davern, P., Nashid, N., Zahran, A., and Sreenan, C. J. (June 2011c). Towards an Automated Client-Side Framework for Evaluating HTTP/TCP Performance. *Proc. of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*.

[Group, 2010] Group, C. C. V. (2010). Images. http://www.vision.caltech.edu/html-files/archive.html.

[Kota and Marchese, 2003] Kota, S. and Marchese, M. (2003). Quality of service for satellite IP networks: a survey. *International Journal of Satellite Communications and Networking*, 21:303–349.

[Riverbed, 2011] Riverbed (2011). Extreme Savings: Cutting Costs with Riverbed. http://www.riverbed.com/us/index.php.

[Ronga et al., 2003] Ronga, L. S., Pecorella, T., Re, E. D., and Fantacci, R. (2003). A gateway architecture for IP satellite networks with dynamic resource management and DiffServ QoS provision. *International Journal of Satellite Communications and Networking*, 21(4-5):351–366.

[Sadre, 2008] Sadre, Ramin, H. B. R. (2008). Changes in the web from 2000 to 2007. In *Proceedings of the 19th IFIP/IEEE international workshop on Distributed Systems: Operations and Management: Managing Large-Scale Service Deployment*, DSOM '08, pages 136–148, Berlin, Heidelberg. Springer-Verlag.