Practical Experience with HTTP and TCP over GPRS

Rajiv Chakravorty and Ian Pratt
Email: {rajiv.chakravorty, ian.pratt}@cl.cam.ac.uk
University of Cambridge Computer Laboratory, Cambridge CB3 0FD, U.K.

1. INTRODUCTION

Performance of protocols like HTTP and TCP are known to degrade over wireless links. This abstract discusses experimental results of TCP and HTTP performance over General Packet Radio Service (GPRS), an extension to the GSM standard which is being deployed rapidly throughout much of the world. The research is aimed at:

1. understanding GPRS link characteristics
2. identifying protocol performance problems experienced by TCP and HTTP over GPRS,
3. devising solutions that can overcome or even optimise such bottlenecks and,
4. experimentally demonstrating the overall efficacy of our solution.

2. TEST ENVIRONMENT AND TOOLS

Our experimental setup for characterizing GPRS links is shown in figure 1. The measurements reported were all performed over Vodafone UK's national GPRS network, though similar results were measured using a number of other European GPRS networks. A number of different handsets from different vendors were tried; we found no significant performance variation between handsets of the same GPRS device class. Measurements were repeated at many different locations resulting in a wide range of radio and network conditions to try and gain a view of 'typical' performance experienced by a user.

In an example test set-up (figure 1), a laptop connects to a Motorola T260 GPRS (3+1) (3 downlink, 1 uplink channels) phone through a serial PPP (point-to-point) link to act as a GPRS mobile terminal. Complying to the usual GPRS architecture, the base stations (BSs) are linked to the SGSN which is then connected to a GGSN. In the current Vodafone configuration, both SGSN and GGSN node is co-located in a CGSN (Combined GPRS Support Node).

Since we were unable to install equipment next to the CGSN we made use of a well provisioned IPSec (IP Security) VPN tunnel to route all traffic via the Computer Laboratory. The measurement terminal (see figure 1) was then located at the end of the tunnel, with routing configured so that all packets flowing to and from the mobile host are passed to it for processing. A RADIUS server was used to authenticate mobile terminals and assign IP addresses.

All the characterization tests were performed using a version of the `ttcp` program modified (`ttcp+` [4]) to enable traffic streams

MOBICOM'02, September 23–28, 2002, Atlanta, Georgia, USA.

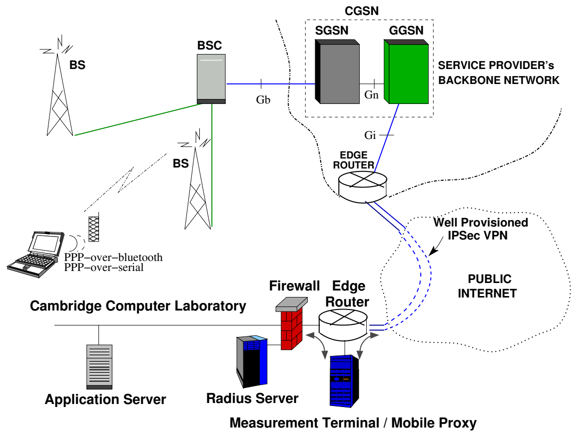

Figure 1: Test Bed Set-Up

to be generated at specified rates and with particular burst characteristics. We also inserted time stamps and sequence numbers in packets to track time-in-flight between sender and receiver (using NTP synchronized clocks) and to detect packet loss and reordering.

3. EXPERIMENTAL EVALUATION

We conducted a series of link characterization measurements over GPRS [3]. Results from the measurements reveal that GPRS links have very high RTTs (>1000ms) and fluctuating available bandwidth. Packet losses are relatively rare when the mobile host is stationary. However, link blackouts can still occur, resulting in stalls of typically 5-30 seconds. Longer outages are more likely to cause packet loss. ACK compression is fairly common over both uplink and downlink, caused by the link-layer protocol.

Due to such link inconsistencies, experimental evaluation of TCP over GPRS uncovered a number of performance problems (see [1] for more information):

- a sluggish slow-start that can take many seconds (due to high RTTs) for the window to ramp-up and achieve full downlink utilization,
- excess queue build up over the downlink with long-lived TCP sessions - this can cause gross unfairness to other TCP flows and a high probability of timeouts during intial connection request,
- spurious TCP timeouts due to occasional link outages and,
- a high *drain time* to recover after TCP timeouts.

We also investigated ways to improve web (HTTP) performance over GPRS [2]. Our approach towards this problem was not only from the perspective of improving TCP, but also from the point of optimizing web browser performance. Web browser behaviour (the way it implements the HTTP protocol) has a substantial bearing on page download times over GPRS. In an effort to improve response times on wired-Internet links, most browsers open multiple concurrent TCP connections. However, this leads to the saturation of the downlink buffers at the GPRS's CGSN node, and an increased overhead that negatively impacts page download times over GPRS.

4. MEASUREMENT RESULTS

We have shown that using an interposed mobile proxy located close to the wired-wireless boundary can give substantial benefits to both TCP and Web (HTTP) performance over GPRS. We offer performance enhancements to the mobile proxy at two different layers: transport layer and also at the application layer.

The transport layer enhancement (TL-E) offers a transparent split TCP approach and uses a fixed size congestion window (cwnd). This approach eliminates the slow start phase and clamps TCP's cwnd to the value of GPRS downlink bandwidth-delay product. The approach has shown to clearly improve TCP performance over GPRS without requiring modification to either client or the server. The scheme has shown to offer (see [1]) the following benefits:

1. *Minimization of Queuing Delays :–* excessive queuing over the downlink is minimized by limiting data of long TCP flows over the link. As a consequence, RTT inflation and its impact on retransmit timer values are also minimized,
2. *Faster Startup :–* slow-start is avoided, which improves start-up performance and transfer times (especially for short web transfers),
3. *Quick Recovery from Losses :–* reduces drain time after losses leading to quicker TCP recovery. By limiting data over the link, spurious retransmission cycles due to sudden delay fluctuations can be avoided. This also reconciles with other negative effects such as stale (or leftover) TCP data due to abnormal disconnections.

Our application level enhancement (AL-E) offers data caching at the mobile proxy and can service pipelined requests from pipelined capable web browsers. For test purposes, we created a mock-up of a popular news web-site, CNN `www.cnn.com`, on a locally provisioned web server. This was done to avoid the effects of fast changing web-content and to eliminate measurement 'noise' that could be introduced by changes in Internet performance. We refer to our dynamic web-site as local CNN, or in short LCNN. To compare the download performance of our scheme, we used the mozilla web browser. The latest version from mozilla supports pipelining. However, after analysing browser traces we found few instances where the browser actually pipelined requests on persistent connections, even when doing so was clearly possible (for detailed discussion on this, see [2]).

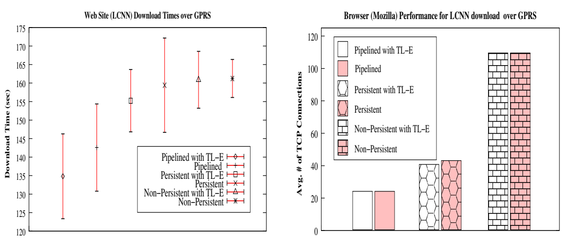

Figure 2: Mozilla Web Browser (a) Mean download times and, (b) Average Number of Connections for LCNN web site from 20 successful runs over GPRS.

Figure 2(a) shows that mean download time (and deviation) for 20 successful runs. The use of TL-E with non-persistent connections show very little performance gains. The same is true for persistent connections that shows only slight performance improvement in mean download times when compared to non-persistent connections. We believe this is because the number of connections made by the browser in persistent mode is relatively high (an average of more than 40 connections made, see figure 2(b)); therefore, gains made by eliminating slow-start is hidden by a high connection (control and transactional) overhead. The browser achieved a reasonable improvement with pipelined mode using the transport level enhancement (TL-E). We found an average 15% reduction in mean download times with the browser when using the pipelined mode with TL-E. The average number of connections from 109 in non-persistent mode, reduced to an average of 24, an overall reduction of more than 75% in the total number of connections when using pipelined connections. A 15% improvement over download times is modest but justifiable taking into account mozilla's low pipelining efficiency [2].

Further, we find that a reduction in the number of connections is advantageous not only due to low-bandwidth nature of the GPRS links but also because it mitigates the overall control and associated transactional (3-way TCP handshake) overhead with each additional connection. Thus we infer that aggressive pipelining of requests over browser connections reduces overall connection (control and transactional) overhead, and combined with the transport level enhancement (TL-E) can result in significant improvement in web download times.

5. FUTURE RESEARCH

There are many interesting artifacts resulting from this piece of research. Specifically, our research entails further questions:

1. How can web-clients be made to dynamically adapt to the underlying network heterogeneity?
2. How do we minimize the possibility of Head of Line (HOL) blocking effects with pipelining?
3. How to maximize pipelining efficiency in presence of resource inter-dependencies with dynamic web-content?

In the future, we also plan to investigate a number of optimization techniques such as application data compression, delta encoding and deterministic prefetching to improve protocol performance over low bandwidth GPRS. Please visit our web-page, `http://www.cl.cam.ac.uk/users/rc277/gprs.html` for more information on our research and related published papers.

Acknowledgements

We thank Vodafone Group R&D, Sun Microsystems Inc. and Benchmark Capital for supporting this work.

6. REFERENCES

[1] Rajiv Chakravorty, Joel Cartwright, Ian Pratt, "Practical Experience With TCP over GPRS", in IEEE GLOBECOM 2002, Novemver 17-21, Taipei, Taiwan http://www.cl.cam.ac.uk/users/rc277/gprs.html

[2] Rajiv Chakravorty and Ian Pratt, "WWW Performance over GPRS", in IEEE 4th International Conference on Mobile and Wireless Communication Networks (MWCN 2002), September 9-11, Stockholm, Sweden http://www.cl.cam.ac.uk/users/rc277/gprs.html

[3] Online Technical Report,"GPRS Link Characterization", http://www.cl.cam.ac.uk/users/rc277/linkchar.html

[4] ttcp+ program, (http://www.cl.cam.ac.uk/Research/SRG/netos/netx/)