Chapter 1

WIRELESS WEB PERFORMANCE ISSUES

Carey Williamson
Department of Computer Science, University of Calgary
carey@cpsc.ucalgary.ca

Abstract

This chapter discusses performance issues that arise for Web content delivery in wireless local area network (WLAN) environments. Two experiments are conducted using IEEE 802.11b WLAN technology. The first experiment considers client-side Web browsing performance for a mobile user with a wireless PDA in an infrastructure-based WLAN. The second experiment considers server-side performance for a wireless Web server delivering Web content in a wireless ad hoc network environment. The results show that the wireless network bottleneck can lead to inefficient HTTP performance, inefficient TCP performance, packet losses, and network thrashing, depending on the characteristics of the Web traffic generated. Solving these issues is important to improve wireless Web performance.

Keywords:

Web performance; Network traffic measurements; IEEE 802.11b WLAN

1. Introduction

Wireless technologies have revolutionalized the way people think about networks, freeing users from the constraints of physical wires, and bringing closer the "anything, anytime, anywhere" promise of mobile networking. At the same time, the Web has made the Internet available to the masses, through its TCP/IP protocol stack and the principle of layering. The next step in the wireless Internet evolution is the convergence of these technologies to enable the "wireless Web" in the classroom, the office, and the home.

This chapter discusses performance issues that arise for Web content delivery in wireless networks. In particular, we consider both client-side and server-side perspectives in the use of wireless LAN technology. From the client perspective,

we study the typical Web browsing performance for a mobile user with a wireless PDA. From the server perspective, we study the performance of an Apache Web server operating in a wireless ad hoc network.

From both of these perspectives we identify protocol performance issues that limit the achievable Web performance. In most cases, the performance problems arise from the wireless network bottleneck, but the bottleneck manifests itself in subtle ways, because of multi-layer protocol interactions. Examples of these interactions include the inefficiencies of non-persistent HTTP over TCP, the congestion response of TCP to wireless packet losses, and the combination of these two effects.

Our work is carried out using experimental measurements. A wireless network analyzer is used to collect and analyze network packet traces, with traffic analysis spanning from the Medium Access Control (MAC) layer to HTTP at the application layer. Multi-layer protocol analysis is used to identify the performance problems that occur.

The remainder of this chapter is organized as follows. Section 2 provides background information on IEEE 802.11b, TCP, and HTTP. Section 3 presents the client-side results for wireless Web browsing performance. Section 4 presents the server-side results for wireless Web content delivery. Finally, Section 5 summarizes the chapter.

2. Background and Related Work

The Web and Web Performance

The Web relies primarily on three communication protocols: IP, TCP, and HTTP. The Internet Protocol (IP) is a connection-less network-layer protocol that provides global addressing and routing on the Internet. The Transmission Control Protocol (TCP) is a connection-oriented transport-layer protocol that provides end-to-end data delivery across the Internet [Stevens 1994]. Among its many functions, TCP has flow control, congestion control, and error recovery mechanisms to provide reliable data transmission between a source and a destination. The robustness of TCP allows it to operate in many network environments. The Hyper-Text Transfer Protocol (HTTP) is a request-response application-layer protocol layered on top of TCP. HTTP is used to transfer Web documents between Web servers and Web clients. Currently, HTTP/1.0 [RFC1945] and HTTP/1.1 [RFC2616] are widely used on the Internet.

Overall Web performance depends on the performance of Web clients, the Web server, and the network in between. The main challenge for Web content delivery in wireless networks is the wireless channel, which has limited bandwidth, high error rates, and interference from other users. The concern is that TCP and HTTP performance may degrade over wireless networks.

Wireless Internet and IEEE 802.11b WLANs

Wireless technologies are playing an increasingly prominent role in the global Internet infrastructure. One of the popular technologies in the wireless LAN market is the IEEE 802.11b standard. This "WiFi" (Wireless Fidelity) technology provides low-cost wireless Internet capability for end users, with up to 11 Mbps data transmission rate at the physical layer.

IEEE 802.11b is just one member of a growing family of IEEE 802.11 WLAN standards. IEEE 802.11g offers data rates of up to 54 Mbps, using more sophisticated modulation schemes in the same 2.4 GHz frequency band as IEEE 802.11b. IEEE 802.11a offers data rates of up to 54 Mbps, operating in the 5 GHz frequency range. Many commercial WLAN products today support IEEE 802.11a/b/g functionality. The emerging IEEE 802.11e standard offers better Quality of Service (QoS) support for WLAN applications, and the future IEEE 802.11n standard promises much higher data rates.

This chapter focuses solely on IEEE 802.11b as a representative example of the IEEE 802.11 WLAN protocols. The IEEE 802.11b standard defines the channel access protocol used at the MAC layer, namely Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). It also defines the frame formats used at the data link layer: 128-bit preamble, 16-bit Start-of-Frame delimiter, 48-bit PLCP (Physical Layer Convergence Protocol) header, followed by a 34-byte Logical Link Control (LLC) header and variable size payload, which can be used for carrying IP packets.

The IEEE 802.11b WLAN technology can be operated in either *infrastructure mode* or *ad hoc* mode. In infrastructure mode, an Access Point (AP) is required to provide connectivity to the Internet. The AP relays IP packets between the mobile users in the WLAN and the external wired Internet. The IEEE 802.11b MAC protocol is used for all the datalink layer frames exchanged between a mobile node and the AP.

In ad hoc mode, there is no external Internet involved, and no AP required. Stations within the WLAN communicate with each other directly in a peer-to-peer fashion. All frames are addressed from the sender to the intended receiver using the corresponding MAC address in the frame header. Frames that are correctly received over the shared wireless channel are acknowledged right away by the receiver. Unacknowledged frames are retransmitted by the sender after a short timeout (a few milliseconds), using the same MAC protocol.

Related Work

There is a growing set of literature on wireless traffic measurement and Internet protocol performance over wireless networks [Balachandran et al. 2002, Bennington et al. 1997, Cheng et al. 1999, Singh et al. 2002, Tang et al. 1999, Tang et al. 2000]. For example, Tang and Baker [Tang et al. 1999, Tang et

al. 2000] discuss wireless network measurements from two different environments: a metropolitan area network, and a local area network. More recently, Balachandran *et al.* [Balachandran et al. 2002] report on network performance and user behaviour for general Internet access by several hundred wireless LAN users during the ACM SIGCOMM conference in San Diego in 2001. They find that for this set of technology-literate users a wide range of Internet applications are used, user behaviours are diverse, and overall bandwidth demands are moderate. Kotz and Essien [Kotz et al. 2002, Henderson et al. 2004] characterize campus-wide wireless network usage at Dartmouth College, for infrastructure mode using access points. Schwab and Bunt [Schwab et al. 2004] present a similar study for the University of Saskatchewan.

Our work differs from these in that we consider both wireless Web clients and a wireless Web server in a WLAN, in either infrastructure mode [Omotayo et al. 2004] or ad hoc mode [Bai et al. 2003, Bai et al. 2004, Oladosu 2003]. Our main focus is on the multi-layer protocol interactions that occur, and their impact on user-perceived Web performance.

3. Wireless Web Browsing Performance

Today's mobile computing devices offer users unprecedented connectivity and convenience. Many mobile users rely on a wireless laptop, Personal Digital Assistant (PDA), or cell phone for Internet access, with Web browsing a primary application.

Wireless Web access, however, is not without its performance problems. Wireless channel bandwidth is often limited compared to desktop wired-Internet access, and the wireless channel is typically shared amongst multiple users. In addition, the wireless channel quality can vary significantly with time, with an inherent error rate much higher than that for wired network technologies.

This section presents a detailed analysis of Web browsing performance for a mobile user with a wireless PDA. Our analysis studies the multi-layer protocol interactions that occur when HTTP and TCP/IP operate over an IEEE 802.11b WLAN. A wireless network analyzer is used to collect TCP/IP packet traces of Web traffic generated by the mobile user's wireless PDA. Trace analysis focuses on server response time, HTTP transfer time, TCP performance, and wireless channel quality.

Our results identify several protocol-related issues that affect wireless Web browsing performance. These results provide insight into performance enhancements for Web content providers, Web servers, HTTP, TCP, and IEEE 802.11b.

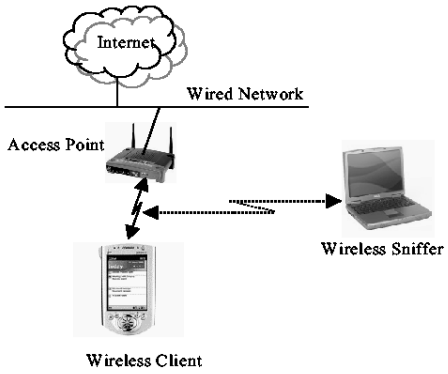


Figure 1.1. Experimental setup for study

Experimental Setup

Our client-side Web browsing experiments are conducted on an IEEE 802.11b WLAN in the Department of Computer Science at the University of Calgary. The experimental setup is illustrated in Figure 1.1. The network operates in *infrastructure mode*, with the Access Point (AP) providing access to the external Internet. The AP is a NetGear WAB 102. The wireless client is a PDA that communicates directly with the AP. These are the only devices present in the WLAN during our experiments.

The PDA is a Compaq iPAQ 3600 Pocket PC running Windows CE (version 3.09348) as the operating system and Internet Explorer as the Web browser. This device has an ARM SA1110 processor and 64 MB Flash RAM. The PDA has a Proxim wireless network interface card (NIC), and a Maximum Transmission Unit (MTU) size of 1500 bytes.

Our study uses a very simple workload: a single user with a wireless PDA browsing selected sites on the Internet. The client makes requests for Web pages by typing a URL request into the Web browser, or clicking on a hyperlink. Each request generates TCP packets, which are sent across the WLAN as encapsulated IEEE 802.11b data frames. The WLAN traffic generated during our study is captured using a wireless network analyzer.

The trace analyzed in this chapter was collected over a period of 35 minutes on March 3, 2004. During this time, the user browsed Web sites offering news, yellow pages, driving directions, stock quotes, educational resources, and downloadable PDA software. Table 1.1 lists several of the Web sites used in this study.

The network traffic measurements were collected using Sniffer Pro 4.60.01, a wireless network analyzer from Sniffer Technologies. This measurement tool

Table 1.1. Partial List of URLs Used in Web Browsing Experiments

Site	*URL*	*Description*
1	www.cpsc.ucalgary.ca	University
2	www.cemonster.com	Internet Services
3	www.cnn.com	News
4	www.forecaster.ca	Sports
5	www.cnet.com	Computers
6	www.quickdrive.com	Travel Info
7	www.handmark.com	Software
8	www.ehosting.ca	Domain Hosting
9	mobile.canada.com	Information
10	weather3.cmc.ec.gc.ca	Weather Info

passively monitors and records all WLAN traffic, enabling protocol analysis at MAC, IP, TCP, and HTTP layers. Packet timestamps are recorded with microsecond resolution.

The Sniffer software runs on a wireless laptop. Our laptop is a Compaq Armada E500 with a 1.0 GHz Mobile Intel Pentium III processor, 128 MB of 100 MHz RAM, 9.36 GB disk, and a Cisco Systems Aironet 350 wireless NIC. In *promiscuous mode*, the NIC records all WLAN traffic.

Table 1.2 provides a statistical summary of the network trace collected during our experiment. The Web browsing session lasted just over 35 minutes, with 394 successful TCP connections observed. Both persistent (13%) and non-persistent (87%) connections are observed.

Table 1.2. Statistical Summary of WLAN Web Browsing Trace

Item	*Value*
Trace Duration (min:sec)	35:33.212
Total TCP Packets	13,705
Total Data Bytes	7,216,491
Total TCP Connections	398
Successful TCP Connections	394

Statistical analysis of the trace shows that the network is lightly loaded. The average (user-level) data rate for the entire trace is about 25 Kbps (6.4 pkt/sec), with a peak transfer rate of 1.2 Mbps (180 pkt/sec) on the 11 Mbps WLAN.

Figure 1.2 provides a graphical representation of the network traffic during the trace. The solid vertical lines in the graph show the number of TCP packets transmitted on the WLAN in each 1 second interval of the trace, while the lower horizontal dashed line shows the number of simultaneously active TCP connections from the client.

Network usage is clearly bursty, as is typical of Web browsing activity. The browser supports parallel TCP connections, typically with 3-4 connections ac-

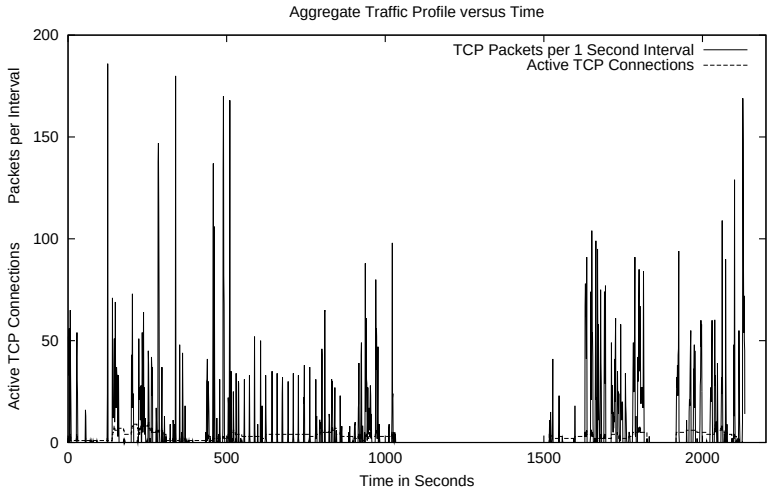


Figure 1.2. Time Series Representation of Web Browsing Traffic

tive at a time, though as many as 10 were observed at one point in the trace. The default socket buffer size was 32 KB for each connection. A total of 56 different IP addresses were seen in the trace, since many commercial Web sites use server clusters, and many Web pages contain advertising banners. About 52% of the TCP packets in the trace were transmitted by the client PDA, which suggests that the TCP implementation in Windows CE does per-packet acknowledgements, rather than the usual TCP Delayed ACKs [Stevens 1994]. The use of Delayed ACKs or some form of ACK consolidation would economize on wireless network usage, and conserve battery power for the wireless device.

Two idle periods appear in the trace. These occurred when the PDA was rebooted and reassociated with the AP.

HTTP-layer Analysis

Our first analysis focuses on protocol performance at the HTTP layer. We are interested in issues such as Web server response time, HTTP transfer times for Web objects, and Web object sizes.

Figure 1.3 provides a time series representation of the Web server response time in our experiments. Server response time is defined as the elapsed time between the HTTP "GET" request sent by the client and the first packet of the HTTP response from the server.

Figure 1.3 shows the server response time for each TCP connection initiated in the trace. The graph shows sustained Web browsing to several Web sites, with multiple TCP connections used for most of these sites. Server response time is quite consistent for a given Web site, since the network round trip time

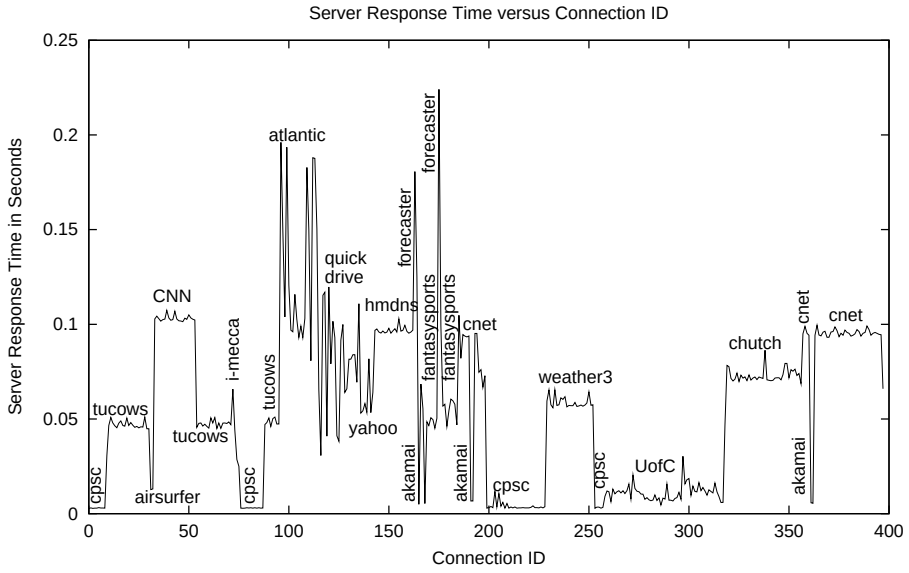
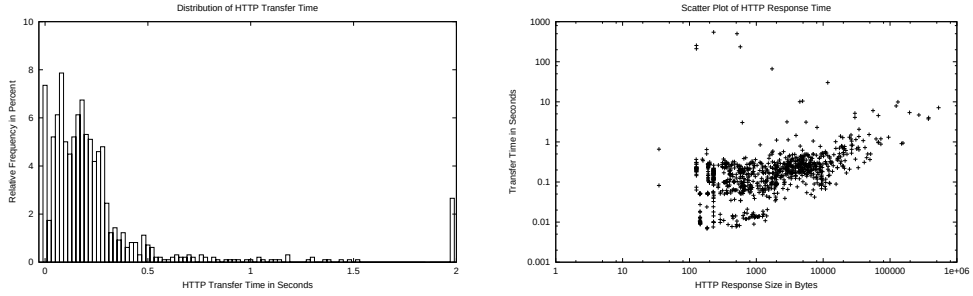


Figure 1.3. Web Server Response Time Results



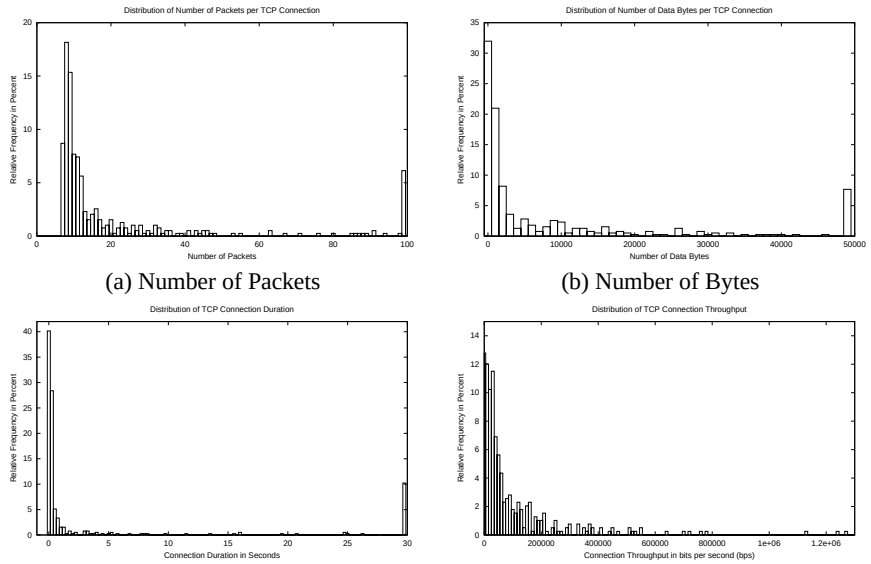
(a) HTTP Transfer Time Distribution (b) Scatter Plot of HTTP Transfer Times

Figure 1.4. HTTP Transfer Time Analysis

(RTT) is a dominant component of this latency. However, server response time can vary a lot from one Web site to another.

The next analysis studies HTTP transfer time: the elapsed time from when a mobile client makes a GET request to when the client has all of the corresponding response data from the Web server (using 1 or more network packets).

Figure 1.4(a) shows the distribution of HTTP transfer times. In general, the HTTP transfer times are low. About 96% of the transfers complete in less than 1 second, and only 2.5% require longer than 2 seconds.



(a) Number of Packets (b) Number of Bytes

(c) Connection Duration (d) Connection Throughput

Figure 1.5. TCP Connection-level Analysis

Figure 1.4(b) shows a scatter-plot of the HTTP transfer time versus the Web object size for each transfer. The expected trend is that larger Web objects take longer to download. While this trend is present in the data, the density of points in the graph makes it hard to see. There are also several outliers among the data points. A few small Web objects had excessively long HTTP transfer times.

TCP-layer Analysis

Figure 1.5 presents results from TCP connection-level analysis. Figure 1.5(a) shows the distribution of packets per connection, while Figure 1.5(b) shows data bytes per connection, and Figure 1.5(c) shows TCP connection durations.

The main observation from Figure 1.5 is that most TCP connections are brief. Approximately 75% of all connections sent fewer than 20 packets, and only 6% sent more than 100 packets. The mean was 35 packets per connection; the fewest sent was 8 packets, and the most was 653 packets. About 80% of the connections sent fewer than 10 KB, and only 8% sent more than 50 KB. Approximately 75% of all connections lasted less than 1 second, and 87% lasted less than 10 seconds. This observation is consistent with the earlier observations that most TCP connections are non-persistent, and most Web object transfers are small. In other words, Web content designed for mobile users is small and usually downloads quickly. While 10% of connections last longer than 30 seconds, over 98% of the connections complete within 100 seconds.

The final TCP-layer analysis combines HTTP transfer size information with TCP connection duration to study the average throughput for TCP connections. Throughput is expressed in Kilobits per second (Kbps). Higher values reflect more efficient usage of the network.

Figure 1.5(d) shows TCP connection throughput results. About 95% of the connections had throughputs below 400 Kbps. This throughput is low considering the 11 Mbps physical-layer data rate in IEEE 802.11b.

To better understand the low throughput, the mobile client's HTTP requests were studied. Analysis shows that many TCP connections request a single embedded object from a Web page with many embedded objects. The cost of making multiple TCP connections to obtain the embedded objects limits the effective throughput. In other words, the low throughput is a consequence of small HTTP transfer sizes, non-negligible RTTs, TCP slow start effects, and non-persistent TCP connections. Using persistent connections would significantly improve the throughput.

MAC-layer Analysis

Our final analysis in the Web browsing study focuses on performance anomalies related to the wireless channel quality.

The total number of MAC-layer retransmissions observed was 533. These retries affected 407 packets (3.0%), from 159 connections (40%). In addition, the wireless analyzer reported CRC errors for 0.04% of the total packets. CRC errors are evenly distributed throughout the trace, reflecting the randomness of wireless channel errors.

These observations suggest that the wireless channel quality does not have a major impact on wireless Web browsing performance in our experiment. Rather, it is the HTTP and TCP inefficiencies that limit the Web performance.

4. Wireless Web Server Performance

The second part of our experimental study focuses on wireless Web servers. Wireless Web servers play a valuable role in *short-lived networks*. A short-lived (or *portable*) network is created in an *ad hoc* fashion at a particular location in response to some event (scheduled or unscheduled). The network operates for some short time period (minutes to hours), before being disassembled, moved, and reconstituted elsewhere.

There are several distinguishing characteristics of a portable short-lived network. Often, the location of the needed network is not known *a priori*. There may not be *any* existing network infrastructure, either wired or wireless, at the needed location. Deployment may need to be spontaneous, with unknown (but often bounded) operating duration. The number of users is typically small (e.g., 10-100), bandwidth requirements are modest, and the geographic coverage area

is limited. There is a need for either data dissemination or data collection at the network site, typically involving a "closed" set of specialized content, rather than general Internet content.

Examples of deployment scenarios for short-lived networks are sporting events, press conferences, conventions and trade shows, disaster recovery sites, and classroom area networks. The potential for entertainment applications (e.g., media streaming, home networking, multi-player gaming) is also high. In many of these contexts, an ad hoc wireless network, with a wireless Web server as an information repository, provides a suitable solution.

Our experiments focus on the HTTP transaction rate and end-to-end throughput achievable in an ad hoc wireless network environment, and the impacts of factors such as number of clients, Web object size, and persistent HTTP connections. The results show the impacts of the wireless network bottleneck, either at the client or the server, depending on the Web workload. Persistent HTTP connections dramatically improve the performance for mobile clients accessing content from a wireless Web server.

Experimental Setup

Our wireless Web server experiments were conducted on an IEEE 802.11b WLAN in our research laboratory. The simple testbed (shown in Figure 1.6) consists of several mobile clients and one Web server. In addition, we use a wireless network analyzer to monitor the wireless channel.

Each of the client and server machines is a Compaq Evo Notebook N600c running RedHat Linux 7.3 and X windows, using a 1.2 GHz Mobile Intel Pentium III. All unnecessary OS processes were disabled prior to conducting measurements, to reduce contention for system resources.

Each laptop has a Cisco Aironet 350 Series Adapter for access to the IEEE 802.11b wireless LAN. The wireless cards are configured in ad hoc mode. During our experiments, these are the only machines operating on the wireless LAN. For simplicity, we do not consider node mobility, multihop, or ad hoc routing issues in our experiments.

The Web server in our experiments is an Apache HTTP server (Version 1.3.23). This version is a process-based implementation of Apache, which is a flexible and powerful HTTP/1.1-compliant Web server [Hu et al. 2001, Nahum et al. 2001]. Apache is currently widely deployed on the Internet, used by approximately 60% of all Web sites [Netcraft].

Network traffic measurements are collected using a WLAN analyzer. Decoding of the captured traces enables protocol analysis at the MAC, IP, TCP, and HTTP layers.

Table 1.3. Experimental Factors and Levels

Factor	*Levels*
Number of Clients	**1**, 2, 3, 4
Per-Client TCP Connection Request Rate (per second)	**10**, 20, 30, . . ., 160
HTTP Transfer Size (KB)	**1**, 2, 4, 8, . . ., 64
Persistent Connections	**no**, yes
HTTP Requests per Connection	**1**, 5, 10, 15, . . ., 60

The IEEE 802.11b wireless LAN is the performance bottleneck in our experimental environment. The experiments are designed to demonstrate how the wireless bottleneck affects network-level and user-level Web performance.

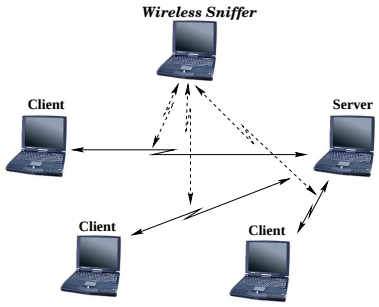


Figure 1.6. Experimental Setup

Experimental Design

A one-factor-at-a-time experimental design is used to study the impacts of many factors on wireless Web server performance, including HTTP transaction rate, number of clients, transfer size, and HTTP protocol version. The experimental factors are summarized in Table 1.3. The values in bold font show the default levels used.

In our experiments, `httperf` [Mosberger et al. 1998] is used to generate client requests to the Web server. `httperf` is a Web workload generation tool developed at Hewlett-Packard Laboratories for Web performance measurement. It provides a flexible means to generate HTTP workloads and measure server performance. We use synthetic Web workloads that are easy to generate, analyze, and reproduce. Our goals are to determine an upper bound on achievable performance, and to understand behaviour under overload conditions, using the simplest scenarios possible.

The experiments are conducted using `httperf` as an *open-loop* workload generator. We invoke `httperf` on the client machine, and send requests to the

server at a specified rate to retrieve a target Web object repeatedly. Each test lasts 2 minutes, with each TCP connection issuing one or more HTTP requests, depending on the test. The "user abort" timeout in `httperf` is set to 5 seconds.

Performance data are collected using `httperf` and the WLAN analyzer. The `httperf` tool reports application-layer statistics on HTTP behaviours (e.g., reply rate, throughput, response time, error rate), providing a user-level view of performance. Detailed measurements from the WLAN analyzer enable traffic analysis from the MAC layer to the HTTP layer. These traces are used to assess wireless channel contention, TCP protocol behaviours, and HTTP transaction performance.

Stress-Testing and Overload Performance

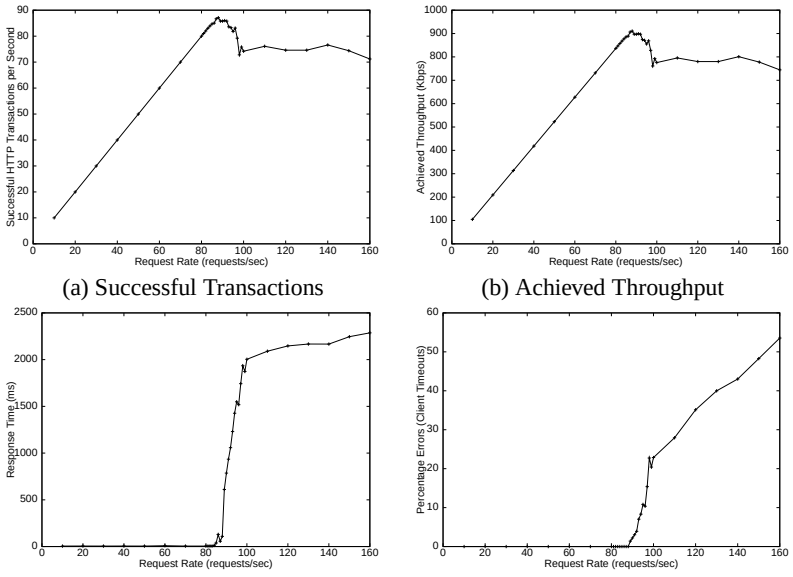
The purpose of the first experiment is to determine the sustainable load for a wireless Web server. Initially, only a single Web client machine is used. The client, server, and Sniffer laptops are all on the same desk in the same office, less than 1 meter apart. The wireless channel is assumed to be excellent. The Web object size is 1 KB.

The experiments begin with a request rate of 10 requests per second, using non-persistent connections. That is, there is exactly one HTTP "GET" request per TCP connection; the terms "TCP connection rate" and "HTTP transaction rate" are thus synonymous for this experiment. When one test is complete, the test with the next higher HTTP transaction rate (from 10 to 160 requests per second) begins. The network trace shows that each HTTP transaction generates 10 TCP packets (6 from the client, and 4 from the server). Each TCP packet requires access to the IEEE 802.11b WLAN for the transmission of the frame and its corresponding MAC-layer ACK.

Figure 1.7 shows the application-layer `httperf` results for this experiment. The plots show the successful HTTP transaction rate (Figure 1.7(a)), the achieved user-level throughput (Figure 1.7(b)), the user-perceived response time (Figure 1.7(c)), and the "user abort" error rate (Figure 1.7(d)). In all four graphs, there are two regimes: the "normal" operating regime for feasible loads, and the "overload" regime from the open-loop workload.

Figure 1.7(a) shows the successful HTTP transaction rate as the offered load increases. Initially, the HTTP transaction rate increases linearly with offered load (as expected), up to about 85 requests per second. Beyond this point, there is some instability, and a drop to a lower plateau. Qualitatively similar results are observed in experiments with the same client and server in a 10 Mbps wired-Ethernet LAN, though the peak HTTP transaction rate is 380 requests per second. The peak performance in the WLAN is lower by a factor of 4.5.

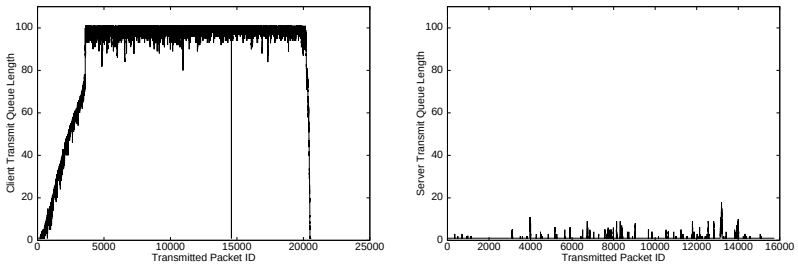
The low HTTP transaction rate is explained by the bottleneck at the *client* network interface, where packets wait at the link-layer queue for medium ac-



(a) Successful Transactions (b) Achieved Throughput

(c) Response Time (d) Error Rate

Figure 1.7. `httperf` Results for Experiment 1 (1 client, 1 KB, non-persistent)



(a) Client (b) Server

Figure 1.8. Link-Layer Transmit Queue for Experiment 1 (1 client, 1 KB, non-persistent)

cess on the WLAN. Figure 1.8 shows this behaviour for high load on a specially instrumented Linux kernel. With the default Linux queue size of 100, the client queue (Figure 1.8(a)) fills in about 10 seconds. Many packet drops occur from this link-layer queue, *before* the packets make it onto the wireless network. These packet losses cause the dropoff in HTTP performance. The server does not receive enough requests to keep it busy, so its queue (Figure 1.8(b)) does not fill. Increasing the client queue size is pointless, since there is no backpressure mechanism to prevent `httperf` from overflowing it; while each TCP connection sends few packets, the sheer number of active TCP connections eventually overwhelms the queue.

With multiple clients, the wireless network bottleneck shifts from the client to the server. Experiments with 2 or more clients show that the server can handle 110 requests per second for 1 KB objects. Beyond this load, the server's link-layer queue fills and overflows, leading to lost packets and erratic TCP performance.

TCP Protocol Overhead

Figure 1.7(b) shows the application-layer throughput as a function of offered load. The peak throughput achieved is just under 1 Mbps, far from the nominal 11 Mbps capacity of the IEEE 802.11b wireless LAN. Experiments on a 10 Mbps wired-Ethernet LAN achieve 3.8 Mbps.

With non-persistent HTTP connections, most of the WLAN packets are small control packets, and the TCP connection establishment overhead is high relative to the connection lifetime. Each HTTP transaction requires a three-way handshake for TCP connection setup, followed by a 74-byte HTTP request, a 1 KB response, and then a three-way handshake to close the TCP connection. A typical HTTP transaction (10 packets) takes about 9 msec on the wireless LAN. This HTTP transaction time is about 4 times slower than that observed in similar tests of the same client and server on a 10 Mbps Ethernet LAN. Clearly, the wireless MAC protocol overhead limits HTTP transaction performance.

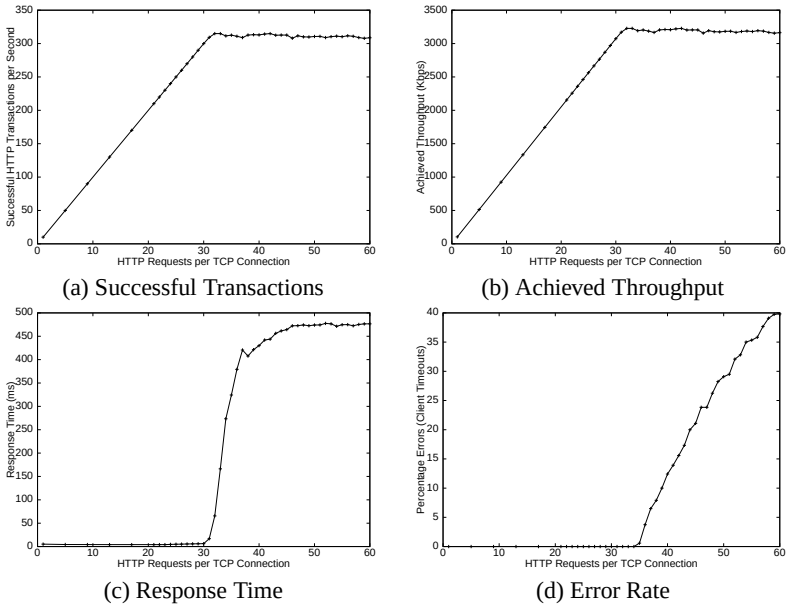
Figure 1.7(c) shows the average response time for the successful HTTP transactions. The response time remains near 9 ms as the offered load increases from 10 to 85 requests per second. At higher loads, the response time increases significantly, eventually exceeding 2 seconds. Figure 1.7(d) shows `httperf` "user abort" errors from client-side timeouts. Under overload, aborts occur frequently.

Persistent HTTP Connections

The next experiment considers persistent connections, with multiple HTTP transactions sent on the same TCP connection, prior to it being closed [RFC2616]. This approach amortizes TCP overhead across multiple HTTP transactions, improving HTTP server performance [Nielsen et al. 1997, Padmanabhan et al. 1994, Spero].

Figure 1.9 shows the `httperf` results for this experiment. In all cases, the TCP connection rate is 10 requests per second, and the transfer size is 1 KB. The number of HTTP transactions per TCP connection is varied.

Figure 1.9(a) shows that the successful transaction rate increases with the number of HTTP requests per connection. The highest rate achieved is 320 HTTP transactions per second. User-level throughput (Figure 1.9(b)) reaches a peak of 3.2 Mbps with 32 HTTP transactions per TCP connection. Beyond that



(a) Successful Transactions (b) Achieved Throughput

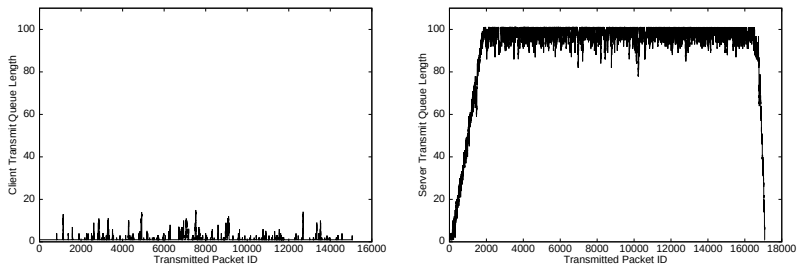
(c) Response Time (d) Error Rate

Figure 1.9. `httperf` Results for Experiment 2 (1 client, 1 KB, 10 conn/sec, persistent)

point, server throughput is relatively stable, though the average HTTP response time increases sharply (see Figure 1.9(c)).

Compared to the results in Figure 1.7(b), the maximum throughput has increased from 900 Kbps to 3.2 Mbps. These results show that persistent connections offer a 350% improvement in performance over non-persistent connections. These performance improvements are even more dramatic than those reported in the previous literature for HTTP/1.1 [Cheng et al. 1999, Nielsen et al. 1997]. For example, in 10 Mbps wired-Ethernet experiments, persistent connections (merely) double the performance from 380 to 760 HTTP transactions per second. The user-level throughput reaches 7.8 Mbps.

Clearly, persistent connections offer many advantages: fewer control packets (TCP SYN and FIN) on the network, and amortization of the TCP handshakes over many HTTP transactions. These advantages apply to any network environment, wired or wireless, but they are particularly important when the wireless LAN is the bottleneck. With persistent connections, the first HTTP transaction inside the TCP connection requires only 4 TCP packets (GET, ACK, DATA, ACK) instead of 10, while all subsequent HTTP transactions in the same TCP connection require only 2 packets (since TCP can piggyback ACKs on outbound GET and DATA packets). This five-fold reduction in the number of TCP packets per HTTP transaction dramatically reduces the demand on the wireless LAN medium access bottleneck, improving HTTP performance accordingly.



(a) Client (b) Server

Figure 1.10. Link-Layer Transmit Queue for Experiment 3 (1 client, 64 KB, non-persistent)

Network Thrashing Problem

The final experiment studies the impact of HTTP response size on network throughput, for a single client issuing 10 requests per second to the server. The transfer size is 1 KB for the first run, and is then increased to 2 KB, 4 KB, and so on in the subsequent runs.

For space reasons, we discuss the results from this experiment only for two selected transfer sizes: 32 KB and 64 KB. These values represent medium load and overload conditions for the wireless Web server, respectively. The 32 KB transfers complete in about 67 msec, for an average throughput of 3.9 Mbps. A 64 KB transfer, however, takes (on average) well over 100 msec, leading to system overload.

In this experiment, the WLAN bottleneck is at the *server* network interface, since the server transmits more packets than the client, and larger packets as well. The `httperf` request rate is modest (10 requests per second), placing little stress on the client-side queue. Figure 1.10 illustrates the queue buildup at the server, which increases the HTTP response time by more than an order of magnitude. The large delay is due to the sizes of the queued data packets.

Detailed analysis of the 64 KB scenario shows that about 50% of the TCP connections are aborted with a TCP reset prior to completion. Fewer than 2% of these connections failed during the opening TCP handshake. Most were aborted partially through the transfer. On average, each reset connection sent 68 packets and 47 KB of data.

Network bandwidth is the scarce resource in this experiment. The main concern is “network thrashing”: a large portion of the wireless channel bandwidth is consumed by partial TCP transfers that eventually abort. While the average throughput at the network layer exceeds 5 Mbps, the effective user-level *goodput* is less than 2.5 Mbps.

5. Summary and Conclusions

This chapter studies the performance of Web content delivery in a wireless network environment. Application-layer and network-layer measurements are used to assess performance in an IEEE 802.11b WLAN, both from wireless client and wireless server perspectives. The experiments focus on HTTP performance, TCP performance, and the impacts of the wireless network bottleneck. In many cases, subtle multi-layer protocol interactions occur.

For the client-side Web browsing experiments, two main observations are evident from our results. First, persistent HTTP connections are much more efficient than non-persistent HTTP connections. It is important that Web browser software and Web site developers enable this feature to improve wireless Web browsing performance. Second, the TCP implementations can be further optimized on wireless PDA devices. Simple features such as TCP Delayed ACKs and the caching of TCP connection state parameters could conserve battery power, reduce wireless network bandwidth usage, and reduce the number of network RTTs incurred.

For the wireless Web server experiments, there are three main observations. First, a wireless Web server in an IEEE 802.11b ad hoc WLAN can support up to 110 transactions per second for non-persistent HTTP and 320 HTTP transactions per second for persistent connections. Typical throughput ranges from 1-3 Mbps. These results again illustrate the large performance advantages of persistent connections in a WLAN environment. Second, TCP performance under overload can be poor. Packet losses occur even before they make it onto the wireless LAN. These lost packets can seriously degrade TCP handshaking performance, or even cause aborted HTTP transfers. Third, the wireless network bottleneck can manifest itself in several different ways, depending on the multi-layer protocol interactions that arise. In our experiments, we have observed wireless LAN bottlenecks at either the client or the server, depending on the workload.

Finding practical approaches to improve Web content delivery on wireless networks remains as a high priority on our research agenda.

Acknowledgements

Financial support for this research was provided by iCORE (Informatics Circle of Research Excellence) in the Province of Alberta, as well as NSERC (Natural Sciences and Engineering Research Council) and CFI (Canada Foundation for Innovation). The author is grateful to Adesola Omotayo, Guangwei Bai, and Kehinde Oladosu for carrying out much of the wireless network measurement work described in this chapter, and to Nayden Markatchev for incomparable technical support for this work.

References

G. Bai and C. Williamson, Simulation Evaluation of Wireless Web Performance in an IEEE 802.11b Classroom Area Network. *Proceedings of the Third International Workshop on Wireless Local Networks* (WLN), Bonn, Germany, pp. 663-672, October 2003.

G. Bai, K. Oladosu, and C. Williamson, Performance Issues for Wireless Web Servers. *Proceedings of the International Workshop on Mobile, Wireless, and Adhoc Networks* (MWAN), Las Vegas, NV, pp. 59-65, June 2004.

A. Balachandran, G. Voelker, P. Bahl, and P. Rangan, Characterizing User Behavior and Network Performance in a Public Wireless LAN. *Proceedings of ACM SIGMETRICS Conference*, Marina del Rey, CA, pp. 195-205, June 2002.

B. Bennington and C. Bartel, Wireless Andrew: Experience Building a High Speed, Campus-Wide Wireless Data Network. *Proceedings of ACM MOBICOM Conference*, Budapest, Hungary, pp. 55-65, September 1997.

S. Cheng, K. Lai, and M. Baker, Analysis of HTTP/1.1 Performance on a Wireless Network. Technical Report CSL-TR-99-778, Stanford University, February 1999.

T. Henderson, D. Kotz, and I. Abyzov, The Changing Usage of a Mature Campus-Wide Wireless Network. *Proceedings of ACM MOBICOM Conference*, Philadelphia, PA, pp. 187-201, September 2004.

Y. Hu, A. Nanda, and Q. Yang, Measurement, Analysis, and Performance Improvement of the Apache Web Server. *International Journal of Computers and Their Applications*, Vol. 8, No. 4, December 2001.

D. Kotz and K. Essien, Analysis of a Campus-Wide Wireless Network. *Proceedings of ACM MOBICOM Conference*, Atlanta, GA, September 2002.

D. Mosberger and T. Jin, httperf—A Tool for Measuring Web Server Performance. *ACM Performance Evaluation Review*, Vol. 26, No. 3, pp. 31-37, December 1998.

E. Nahum, M. Rosu, S. Seshan, and J. Almeida, The Effects of Wide-Area Conditions on WWW Server Performance. *Proceedings of ACM SIGMETRICS Conference*, Cambridge, MA, pp. 257-267, June 2001.

Netcraft, http://www.netcraft.com/survey

H. Nielsen *et al.*, Network Performance Effects of HTTP/1.1, CSS1, and PNG. *Proceedings of ACM SIGCOMM Conference*, Cannes, France, pp. 155-166, September 1997.

A. Omotayo and C. Williamson, Multi-Layer Analysis of Web Browsing Performance for Wireless PDAs. *Proceedings of IEEE Workshop on Wireless Local Networks* (WLN), Tampa Bay, FL, pp. 660-667, November 2004.

V. Padmanabhan and J. Mogul, Improving HTTP Latency. *Computer Networks and ISDN Systems*, Vol. 28, pp. 25-35, December 1995.

K. Oladosu, *Performance and Robustness Testing of Wireless Web Servers*, M.Sc. Thesis, Department of Computer Science, U. of Calgary, August 2003.

RFC 1945: Hypertext Transfer Protocol – HTTP/1.0. http://www.ietf.org/rfc/rfc1945.txt

RFC 2616: Hypertext Transfer Protocol – HTTP/1.1. http://www.ietf.org/rfc/rfc2616.txt

H. Singh and P. Singh, Energy Consumption of TCP Reno, TCP NewReno, and SACK in Multihop Wireless Networks. *Proceedings of ACM SIGMETRICS Conference*, Marina Del Rey, CA, pp. 206-216, June 2002.

D. Schwab and R. Bunt, Characterising the Use of a Campus Wireless Network. *Proceedings of IEEE INFOCOMM Conference*, Hong Kong, March 2004.

S. Spero, Analysis of HTTP Performance Problems. `http://sunsite.unc.edu/mdma-release/http-prob.html`

W. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, Addison-Wesley, 1994.

D. Tang and M. Baker, Analysis of a Metropolitan-Area Wireless Network. *Proceedings of ACM MOBICOM Conference*, Seattle, WA, pp. 13-23, August 1999.

D. Tang and M. Baker, Analysis of a Local-Area Wireless Network. *Proceedings of ACM MOBICOM Conference*, Boston, MA, pp. 1-10, August 2000.