A PROXY ARCHITECTURE TO ENHANCE THE PERFORMANCE OF WAP 2.0 BY DATA COMPRESSION

by

ZHANPING YIN

B.Eng., Tianjin University, P. R. China, 1992
M.Eng., Tianjin University, P. R. China, 1995

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 2002

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical & Computer Engineering

The University of British Columbia
Vancouver, Canada

Date Oct. 10, 2002

Abstract

This thesis presents a novel architecture for the Wireless Application Protocol (WAP) 2.0, employing a proxy to isolate the wireless from the wired domain and an advanced data compression scheme. The advanced data compression scheme combines content compression with robust header compression (ROHC) and minimizes the transmitted air-interface traffic, thus significantly reducing the wireless access time (WAT). By using a proxy at the transport layer to form a TLS tunnel, this architecture also overcomes WAP 1.x end-to-end security problems without deploying a gateway at the server side.

A wireless channel is implemented in Network Simulator (NS-2) to emulate the characteristics of both narrow-band IS-95 and wide-band CDMA 2000 1xRTT wireless channels. The performances of the proposed WAP 2.0 architecture and WAP 1.x protocol stack are tested on it. Emulation results show that while WAP 1.x is optimized for narrow-band wireless channels, WAP 2.0 utilizing TCP/IP gives better performance than WAP 1.x in wide-band channels, even without compression. Huge performance enhancements can be achieved by the proposed data compression scheme in WAP 2.0. The reply contents compression is a major contribution to the improvement, while ROHC can offer further enhancement. The request compression is useful for low-bandwidth networks like IS-95, but gives no benefit in high-speed wireless networks, such as CDMA2000 1xRTT. In IS-95 channels, the reply and request compression with ROHC reduces the WAT of WAP 2.0 by over 70% to give comparable performance to WAP 1.x. In CDMA2000 1xRTT, even the uncompressed WAP 2.0 outperforms WAP 1.x, and the proposed method results in over 46% reduction in WAT.

Although a proxy is optional in WAP 2.0, it can optimize the communication process by isolating the wireless from the wired domain, which gives a performance benefit as well as feature service enhancement. The proxy not only prevents the error propagation between wired and wireless domains, but also eliminates the wireless session delays (WSD) due to a TCP connection establishment by long-live connections between the proxy and wireless terminals.

Table of Contents

List of Tables

List of Figures

Acknowledgement

I want to take the opportunity to thank my supervisor Dr. Victor C.M. Leung for his constant encouragement and valuable guidance, which I was fortunate to receive in the past years. I would like to thank Shailesh Sheoran who gave me many suggestions on the test bed configuration. I also want to thank my wife who paved the way to the successful completion of this thesis with her love and support. Thanks also to my parents who always encourage and support me with whatever I do. Without them nothing would have been possible.

This work was supported by grants from TELUS Mobility and the Advanced Systems Institute of BC, and by the Canadian Natural Sciences and Engineering Research Council under grant CRD 247855-01.

Chapter 1: Introduction

1.1 Wireless Application Protocol (WAP) Background

The Wireless Application Protocol (WAP) is a result of joint efforts taken by companies teaming up in an industry group called the WAP Forum. Ericsson, Motorola, Nokia and Openwave (formerly Phone.com) founded the WAP forum in June 1997. The WAP Forum is the industry association that is responsible for developing and fostering the growth of the WAP. WAP is the open, global de facto standard that allows mobile users of wireless devices to securely access and interact with Internet-based content, applications and services. The WAP Forum is comprised of hundreds of members, representing 99 percent of the handsets sold worldwide, and more than 450 million global subscribers. WAP Forum members include worldwide wireless terminal manufacturers, network carriers, infrastructure providers, software developers and other wireless solution providers.

The scope of the WAP Forum is to define a set of specifications to be used by service applications [1][2]. With the quickly growing of wireless market, more and more customers are reached and new services are emerged and provided to subscribers. To enable operators and manufacturers to meet the challenges in advanced services, differentiation, and fast/flexible service creation, WAP selects and defines a set of open, extensible protocols and content formats as a basis for interoperable implementations. The objectives of the WAP Forum are as follows:

- To bring Internet content and advanced data services to digital cellular phones and other wireless terminals.

- To create a global wireless protocol specification that will work across differing wireless network technologies.
- To enable the creation of content and applications that scale across a very wide range of bearer networks and device types.
- To embrace and extend existing standards and technology wherever appropriate.

WAP is the convergence of three rapidly evolving network technologies: wireless data, telephony, and the Internet. The WAP specifications address mobile network characteristics and operator needs by adapting existing network technology to the special requirements of mass-market, hand-held wireless data devices and by introducing new technology where appropriate.

Most of the original technology developed for the Internet is designed for desktop and larger computers that connect with medium to high bandwidth, generally reliable data networks. Mass-market, hand-held wireless devices present a more constrained computing environment compared to desktop computers. Mass-market handheld devices tend to have less powerful CPUs, less memory, restricted power consumption, smaller displays, and different input devices because of fundamental limitations of power and form-factor.

Similarly, compared to wired networks, wireless data networks present a more constrained communication environment. Due to fundamental limitations of power, available spectrum, and mobility, wireless data networks tend to have less bandwidth, more latency, less connection stability, and less predictable availability. At the same time, wireless applications should

maintain bearer and device independence and be interoperable, scaleable, efficient, reliable, and secure.

Driven by these constrains of wireless environment and wireless devices, the WAP Forum released its first specification, WAP 1.0, in April 1998. In subsequent WAP 1.x specification releases, the WAP Forum addressed interoperability, established a certification program, and added various features in response to changes in market requirements and improvements in networks, devices and new technologies.

WAP 2.0 was released in July 2001. It is a next-generation set of specifications that, like previous releases, marks the WAP Forum's ongoing efforts to adopt the most recent Internet standards and protocols. WAP 2.0 also optimizes the usage of higher bandwidths and packet-based connections of wireless networks worldwide. While utilizing and supporting enhancements in the capabilities of the latest wireless devices and Internet content technologies, WAP 2.0 also provides managed backwards compatibility to existing WAP content, applications and services that complies with previous WAP versions [3].

1.2 Thesis Objectives

Most of the WAP performance evaluations are theoretical simulations of various WAP traffic models. There is little work done in evaluating of WAP performance in real networks using real WAP traffics. Also, since WAP 2.0 was newly released, there is no study of the performance evaluation of WAP 2.0 stack against WAP 1.x stack. Although WAP 2.0 is an evolutional step

forward, by adapting the HTTP/TCP/IP stack, it also has some disadvantages over WAP 1.x, for example, more bits are transmitted than in Wireless Session Protocol (WSP), and there are more transactions than in WTP. The motivation for compressing WAP 2.0 messages to improve performance in wireless networks stems from here.

In this thesis, a novel proxy architecture employing an advanced data compression scheme is introduced in WAP 2.0. The compression scheme combines TCP content compression with Robust Header Compression (ROHC) [34], which minimizes the air-interface traffic without protocol conversions. It also overcomes the end-to-end security problem in WAP 1.x by using Transport Layer Security (TLS) tunneling. Its performance is compared against the standard WAP 2.0 configuration and WAP 1.x stack configuration through measurements over different emulated wireless networks. The objectives of this thesis are the following:

- To evaluate the performance enhancement by the proposed data compression scheme in WAP 2.0, and find out the optimum configuration for different wireless network scenarios and channel conditions.
- To assess the performance of a WAP 2.0 data compression configuration against no compression setting, and WAP 1.x protocol stack in various wireless channel conditions.
- To develop models for modeling the conditions of narrow-band IS-95 and wide-band CDMA2000 1xRTT wireless link and to implement a wireless emulation channel for any IP based traffic.
- To compare and analyze the performance enhancement with the proxy architecture over direct connection in WAP 2.0.

1.3 Previous Work

WAP is a relatively new protocol. Since its existence from 1997, there has been active research on various aspects of the protocol. There are a lot of studies on the deployment of WAP in various industries for mobile applications such as mobile game, robotic control, monitoring system, health and telemedicine and so forth [4]-[10].

As for the WAP protocol performance, researches are mostly working on theoretical evaluation of the protocol by some simulation models. WAP performance over GPRS and GSM were studied in literature, and several WAP traffic models were developed to simulate the WAP application and performance evaluation [11][12][13]. In [14], an alternative WAP configuration with enhanced security is evaluated against the standard configuration by emulation.

Security is extremely important for e-commerce services. Security is provided by WTLS in a WAP 1.x stack, analogous to TLS in the Internet. However the WAP 1.x gateway, used for message transmission and protocol conversion, breaks the concept of end-to-end security. This issue was discussed in [14][15][16][17], and deploying the gateway with the WAP server in the secured enterprise site seems to be the only viable transport layer end-to-end security solution. In [18], a WAE_Sec was proposed to provide end-to-end security at the application layer instead of transport layer. Currently, all these researches are based on WAP 1.x protocol stacks.

WAP, like other new protocols, is still in a rapid development stage. New network bearers and other technologies bring the TCP/IP to the wireless domain. Numerous efforts are undertaken to

improve the TCP performance in wireless environment, several IETF workgroups are working on it and trying to bring some standards to it. These research achievements are leveraged into Wireless Profiled TCP in the WAP 2.0 release.

1.4 WAP 2.0 Developments

WAP protocols are largely based on Internet technologies since WAP is intented to extend Internet technologies to wireless networks, bearers and devices. WAP 1.x protocol stacks are aimed to optimize performance in the low-bandwidth, high latency wireless networks. In WAP 1.x stacks, there are some developments in image displays (e.g. WBMP), public-key infrastructure (PKI), end-to-end security, messaging, push technology, and anticipated a number of W3C and IETF standards. WAP 2.0 brings wireless closer to the Internet by capitalizing on a wide range of the latest new technologies and advanced capabilities, such as [3]:

Networks and Network Bearers – Carriers worldwide are upgrading their existing networks with higher-speed bearers, such as General Packet Radio Service (GPRS) and High-Speed Circuit-Switched Data (HSCSD). Third-generation (3G) wireless networks such as WCDMA and CDMA2000 are deploying, and commercially available with higher bandwidth. These higher capable network bearers permit the delivery of new contents (e.g., streaming media) and provide an "always on" availability. These new aspects of the wireless networks in service allow some new operational activities and services come into reality.

TCP/IP as Transport Protocol – Most of the new wireless network technologies today provide IP packet support as a basic data transport protocol. WAP 2.0 develops a mobile profile of TCP for wireless links based on the IETF work in the Performance Implications of Link Characteristics (PILC) Working Group. This profile is optimized for wireless telecommunication environments and is fully interoperable with the standard TCP that operates over the Internet today. This permits the wireless devices to utilize existing Internet technologies. This is a key feature in the WAP 2.0 stack development.

Processors – Manufacturers worldwide continue to introduce smaller devices with faster and more power-efficient processors and dipoles that are higher-definition and in color. With the Moore's Law still in effect, more efficient packaging technology permits smaller integrated circuits and more sophistication in a given size of device. All these technologies provide the new wireless devices with more power and capabilities to enhance the services and to introduce new feature functions delivered to the user.

Mobile-friendly Technologies – With the growth in usage of mobile devices, there is an increased awareness of the needs specific to the mobile user. The WAP Forum has worked with the W3C and the IETF to help characterize the key issues that impact wireless usage of the web. Through this involvement, and from the interest of their own membership, the W3C recently presided over advances in more mobile-friendly technologies, including: the adoption of a Basic profile for the Extensible Hypertext Markup Language (XHTML), the updates to the Composite Capabilities/Preference Profiles (CC/PP), provides basis for UAProf function; and the release of the Cascading Style Sheets (CSS) Mobile Profile.

1.5 WAP 2.0 Architectural Components

In consideration of these developments, WAP 2.0 adopts the most recent Internet standards and protocols. It also provides backward compatibility for existing WAP versions. The following items represent the major architectural components of WAP 2.0 [3]:

Protocol Stack Support – In addition to the WAP 1.x introduced before, WAP 2.0 adds support and services on a stack based on the common Internet stack including support for TCP, TLS and HTTP. By encompassing both stacks, WAP 2.0 devices can work on a broader range of networks and wireless bearers. This thesis concentrates on these two protocol stacks, which are discussed in more detail in Chapter 2 and Chapter 3.

WAP Application Environment – Nominally viewed as the "WAP Browser", the WAP 2.0 Application Environment adopts the developing standards for the Internet browser markup language, and leads to the definition of the XHTML Mobile Profile (XHTMLMP) based on XHTML. WAP 2.0 specifies markup languages XHTMLMP for new contents and WML, a fully conformant XML language in its own right, to support legacy WAP 1.x content. In addition, WAE provides for other content types, such as WBMP images and vCard and vCalender. The WAE in WAP 2.0 also supports style sheets based on the mobile profile CSS from W3C. There is no in-depth discussion on WAE in this thesis since it has no effect on the transmission of contents over air-interface. Since WAP 1.x stack only recognizes WML format, WML is used at WAE during test experiments in order to compare the performance for both legacy WAP 1.x stack and new WAP 2.0 stack.

Additional Services and Capabilities – The WAP specifications have some items that belong to neither the "WAP Stack" nor the "WAP Browser", but help to enrich the environment defined in the WAP specifications. WAP push and Wireless Telephony Application (WTA) are the most important additional services. WAP 2.0 also supports other features, such as User Agent Profile (UAProf) and External Functionality Interface (EFI) to improve user experience. These features expand the capabilities of the wireless devices and improve the ability to deliver useful applications and services. Since the thesis focuses on WAP protocol stacks, these features are not discussed.

1.6 Thesis Outline

In this thesis, an advanced data compression scheme is introduced into WAP 2.0 stack to improve performance. The WAP 1.x and WAP 2.0 protocol stacks are described in Chapter 2. The proposed data compression proxy configuration is discussed together with legacy WAP 1.x network configurations in Chapter 3. Chapter 4 describes the simulation model and test-bed implementation for our experiments. Particularly, in Chapter 5, the wireless channel modeling and implementation for IS-95 and CDMA 2000 1xRTT are illustrated in detail. The performance evaluations of those configurations in Wireless Access Time (WAT) over both IS-95 and CDMA2000 1xRTT wireless channels are presented and analyzed in Chapter 6. It clearly shows that the introduced data compression scheme significantly improves the WAP 2.0 performance in all cases. Finally, some further discussions and optimizations are discussed, followed by a conclusion.

Chapter 2: WAP Protocol Stacks

WAP 2.0 continues its support for the legacy WAP 1.x stack, which is used over those networks that do not provide IP as well as low-bandwidth IP bearers. In addition, Internet protocol support is introduced in the WAP 2.0 release when IP connectivity is available to the mobile device. Both stacks are supported in WAP 2.0 and provide similar services to the application environment. They are discussed in this chapter with emphasis on the transport layer protocol WTP in WAP 1.x and Wireless Profiled TCP (WP-TCP) in WAP 2.0.

2.1 Legacy WAP 1.x Protocol Layers

WAP 1.x protocol stack is optimized for low-bandwidth bearer networks with a relatively long latency (Figure 2.1). It works on all existing bearers. The WAP 2.0 release continues support for the WAP 1.x stack. The layers are briefly introduced in the following paragraphs.

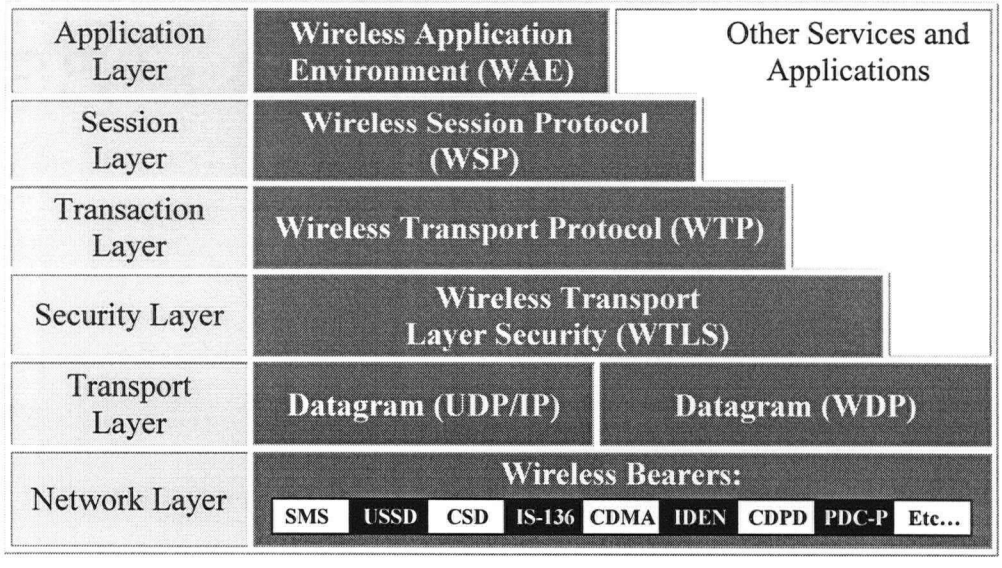


Figure 2.1: WAP 1.x Protocol Stack

2.1.1 Wireless Session Protocol (WSP)

WSP provides HTTP/1.1 functionality and incorporates new features, such as long-lived sessions, session suspend/resume, header caching and push. WSP provides the upper-level application layer of WAP with a consistent interface to a server and exchange messages. WSP has been optimized for providing session services in a much higher latency and unstable wireless networks [19].

WSP provides two types of session services. The first is a connection-mode service that operates above the transaction layer protocol and ensures the reliable exchange of messages between the mobile device and the server. The second is a connectionless service that operates above a secure or non-secure datagram transport service. No acknowledgement of the message is expected from the server in this case.

2.1.2 Wireless Transaction Protocol (WTP)

WTP is defined as a light weight transaction oriented protocol that is suitable for implementation in "thin" clients (mobile stations), and operates efficiently over wireless datagram networks. The benefits of using WTP include the following [3]:

- Improved reliability over datagram services. WTP relieves the upper layer from re-transmissions and acknowledgements that are necessary when datagram services are used.
- Improved efficiency over connection oriented services. WTP has no explicit connection set up or tear down phases.

- The advantage of using a message oriented protocol, designed for services oriented towards transactions, such as "browsing."

WTP supports three classes of transaction services [20]. Transaction classes cannot be negotiated. The WTP provider initiating a transaction is referred to as the Initiator. The WTP provider responding to a transaction is referred to as the Responder. The transaction class is set by the Initiator and indicated in the invoke message sent to the Responder. The basic transaction processes of the 3 classes are shown in Figure 2.2.

Class 0: Unreliable invoke message with no result message

Class 0 transactions provide an unreliable datagram service. It can be used by applications that require an "unreliable push" service. The basic behavior is as follows. One invoke message is sent from the Initiator to the Responder. The Responder does not acknowledge the invoke message and the Initiator does not perform re-transmissions. At the Initiator, the transaction ends when the invoke message has been sent. At the Responder, the transaction ends when the invoke has been received. The transaction is stateless and cannot be aborted.

Class 1: Reliable invoke message with no result message

Class 1 transactions provide a reliable datagram service. It can be used by applications that require a "reliable push" service. The basic behavior for class 1 transactions is as follows. One invoke message is sent from the Initiator to the Responder. The invoke message is acknowledged by the Responder. The Responder maintains state information for some time after the acknowledgement has been sent to handle possible re-transmissions of the acknowledgement if it

gets lost and/or the Initiator retransmits the invoke message. At the Initiator, the transaction ends when the acknowledgement has been received. The transaction can be aborted at any time.

Class 2: Reliable invoke message with exactly one reliable result message

The Class 2 transaction is the basic request/response transaction service. This is the most commonly used transaction service. For example, it is used by WSP for method invocations. The basic behavior for class 2 transactions is as follows: One invoke message is sent from the Initiator to the Responder. The Responder replies with exactly one result message that implicitly acknowledges the invoke message. If the Responder takes longer to service the invoke than the Responder's acknowledgement timer interval, the Responder may reply with a "hold on" acknowledgement before sending the result message. This prevents the Initiator from unnecessarily re -transmitting the invoke message. The Responder sends the result message back to the Initiator. The result message is acknowledged by the Initiator. The Initiator maintains state information for some time after the acknowledgement has been sent. This is done in order to

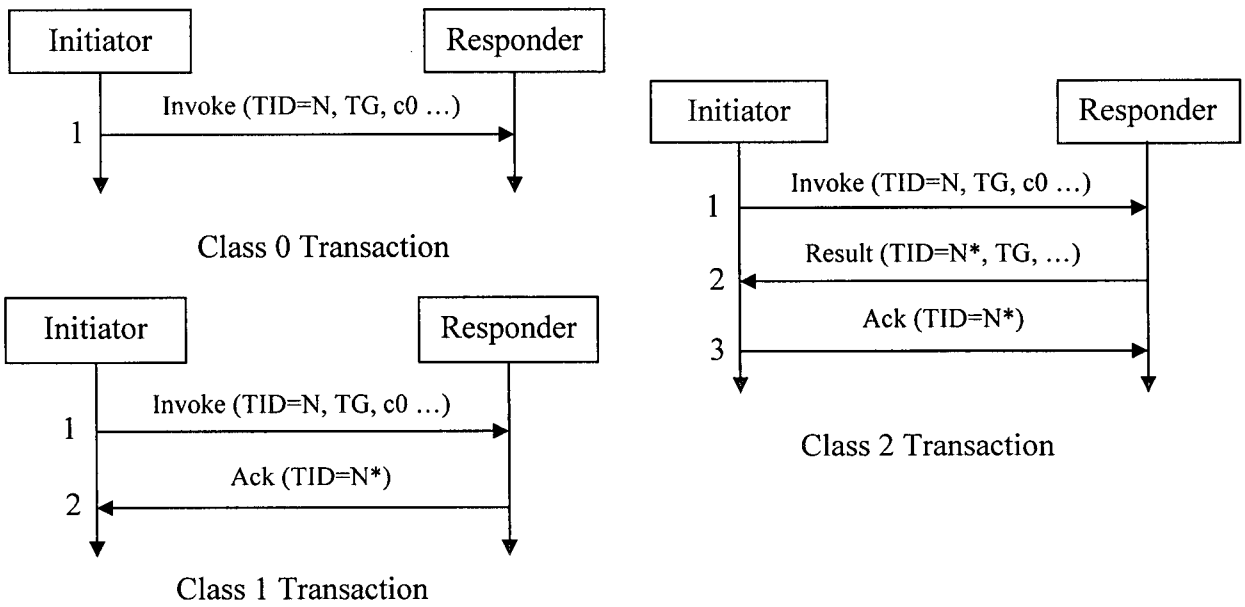


Figure 2.2: WTP Basic Transactions

handle possible re-transmissions of the acknowledgement if it gets lost and/or the Responder re-transmits the result message. At the Responder the transaction ends when the acknowledgement has been received. The transaction can at any time be aborted. If the User acknowledgement function is enabled, the WTP user at the Responder confirms the invoke message before the result is generated. The WTP user at the Initiator confirms the result message before the acknowledgement is sent to the Responder.

2.1.3 Wireless Transport Layer Security (WTLS)

WTLS operates on the top of a datagram service and is an optional layer. The WTLS layer is designed to provide privacy, data integrity and authentication between two communicating applications. It provides the upper-level layer of WAP with a secure transport service interface that preserves the transport service interface below it. In addition, WTLS provides an interface for managing (e.g., creating and terminating) secure connections. It provides a functionality similar to TLS 1.0 and incorporates additional features, such as datagram support, optimized handshake and dynamic key refreshing [3][21].

2.1.4 Wireless Datagram Protocol (WDP)

WDP is a general datagram service located at the base of the WAP1.x protocol stack, offering a consistent service to the upper layer protocols and communicating transparently over one of the available underlying bearer services. This consistency is provided by a set of adaptations to specific features of these bearers, thus providing a common interface to the upper layers. This

allows the upper layers to function independently of the services of the wireless network [22]. If WAP is used over a bearer supporting User Datagram Protocol (UDP), WDP layer is not needed. Since our test experiments are on IP enabled CDMA networks, UDP is used instead of WDP.

2.2 WAP 2.0 Protocol Layers for Networks Supporting IP

The support of Internet protocols is a key feature development of the WAP 2.0 environment. This support is motivated by the emergence of high-speed wireless networks (e.g. 2.5G and 3G) that provide IP support directly to the wireless devices.

2.2.1 Wireless Profiled HTTP (WP-HTTP)

WP-HTTP specification is a profile of HTTP for the wireless environment, and is fully interoperable with HTTP/1.1 [23]. The core of the WP-HTTP specification is the HTTP specification. The basic model of interaction between the WAP terminal and WAP Proxy or WAP Server is a HTTP request/response transaction. The WAP device is capable of interacting with WAP HTTP proxies and origin servers. This transfer layer provides a service access point which may be used in both the "pull" and "push" data transfer models. Pull is achieved using the request/response mechanism from HTTP/1.1. Push functionality is achieved by changing the role of the WAP terminal and considering it as an HTTP server, it can be modeled as a request/response towards the WAP terminal. WP-HTTP supports message body compression of responses and the establishment of a tunnel using the CONNECT method which enables end-to-end security.

2.2.2 Transport Layer Security (TLS)

A wireless profile of the TLS protocol permits manageable interoperability and improved over-the-air efficiency for secure transactions. The WAP profiled TLS includes cipher suites, certificate formats, signing algorithms and the use of session resume. The profile also defines the method for TLS tunneling to support end-to-end security at the transport level if proxy is used between a WAP client and an origin server [24].

2.2.3 Wireless Profiled TCP (WP-TCP)

WP-TCP provides connection-oriented services. It is optimized for wireless environments and is fully interoperable with standard Internet TCP implementations [25]. Compared to wireline connections, cellular networks are characterized by high bit error rates, relatively long delays and variable bandwidth and delays. TCP performance degrades in such environments for the following reasons:

- Packet losses on account of corruption are treated as congestion losses, and lead to the reduction of the congestion window and slow recovery.
- TCP window sizes tend to stay small for long periods of time in high BER environments.
- The use of exponential back-off retransmission mechanisms increases the retransmission timeout resulting in long periods of silence or connection loss
- Independent timers in the link and transport layer may trigger redundant retransmissions.
- Periods of disconnection because of handoffs or the absence of coverage.

Research in optimizing TCP has resulted in a number of mechanisms to improve performance. This includes work by the IETF PILC group that has recommended the use of some of these mechanisms for TCP implementations in long thin networks. Some of the optimizations recommended in WP-TCP are briefly described below.

WP-TCP implementations *should* support large window sizes based on the Bandwidth Delay Product (BDP). In real networks it is very difficult to pick the appropriate window size because of the dynamic characteristics of bandwidth, delay, and congestion. However if the maximum window size is large enough to overwhelm the network (and the necessary buffer sizes are available), the TCP congestion control algorithms will find a congestion window size that is appropriate for the network path. If the window sizes are larger than or equal to 64 KB, WP-TCP *must* support the Window Scale Option and *should* support Timestamps Option for Round Trip Time Measurement (RTTM) [50]. If the window sizes are less than 64 KB, the Window Scale Option and Timestamps Option *may* be used.

The Slow Start algorithm requires that a sender *must* use a Large Initial Window (i.e. the initial size of the congestion window) of up to two segments [51]. An Initial Window greater than two, as described in [52], *may* be supported. WP-TCP implementations *must* support Selective Acknowledgement (SACK) [53], which is especially useful when there is a considerable probability of multiple segment losses per window; such losses are likely when using large windows, or when there is a high possibility of burst errors and congestion losses on the wireless link. WP-TCP implementations *should* implement Path MTU Discovery, which allows a sender to determine the maximum end-to-end transmission unit for a given routing path [54]. WP-TCP

implementations *may* support Explicit Congestion Notification (ECN) [55], which allows a TCP receiver to inform the sender of congestion in the network by setting the ECN-Echo flag. A receiver sets this flag upon receiving an IP packet marked with the CE bit. The TCP sender can then reduce its congestion window. This proposal is still in an experimental/informational state and is believed to provide performance benefits [56].

2.3 Summary

In this chapter, the WAP protocol stacks are briefly introduced. WAP 2.0 continues support for the legacy WAP 1.x stack and brings wireless closer to the Internet by adding support for standard Internet communication protocols. Continuing the work of WAP 1.x, WAP 2.0 permits applications and services to operate over all existing and foreseeable air interface technologies and their bearers, especially for those do not have IP support. The WP-TCP and WP-HTTP provide interoperable optimizations suitable to wireless telecommunications environments and permit the wireless devices to utilize existing Internet technologies. It should be noted that it is expected that these stacks would operate separately, i.e., there would not be mixing and matching of protocols in accomplishing an end-to-end transaction. Figure 2.3 depicts a device that supports both WAP 1.x and WAP 2.0 protocol stacks. The switching of stacks may occur as the device moves in and out of coverage of the different network areas.

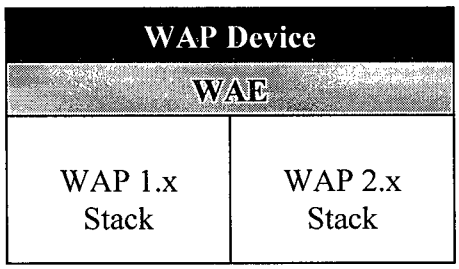


Figure 2.3: Dual WAP Stack Support

Chapter 3: Novel WAP 2.0 Proxy Configuration with Advanced Data Compression

In this chapter, we proposed a novel WAP 2.0 proxy configuration by introducing an advanced compression scheme. The WAP programming model is first introduced, followed by the feature of the performance enhancing proxy model. Various network configurations using a proxy model are presented for the legacy WAP 1.x and WAP 2.0 protocol stacks. The gateway security issue in WAP 1.x is mentioned, and the only viable solution in WAP 1.x is to put the gateway in a secure network with the WAP server. In a WAP 2.0 stack, this security issue can be overcome by using TLS tunneling. An advanced compression scheme in introduced into the WAP 2.0 stack, which can minimize the air-interface traffic and still ensure end-to-end security.

3.1 The WAP Model

3.1.1 WAP Programming Model

The WAP programming model (Figure 3.1), closely aligned with the Web programming model, uses Pull model [2]. However WAP extends the Web model with a few enhancements, the most significant is push functionality and telephony support (WTA). Adopting the WWW programming model provides several benefits to the application developer community, including a familiar programming model, a proven architecture, and the ability to leverage existing tools (e.g., Web servers, XML tools, etc.). Where possible, existing standards are adopted or used as the starting point of the WAP technology. Optimizations and extensions are made in order to match the characteristics of the wireless environment.

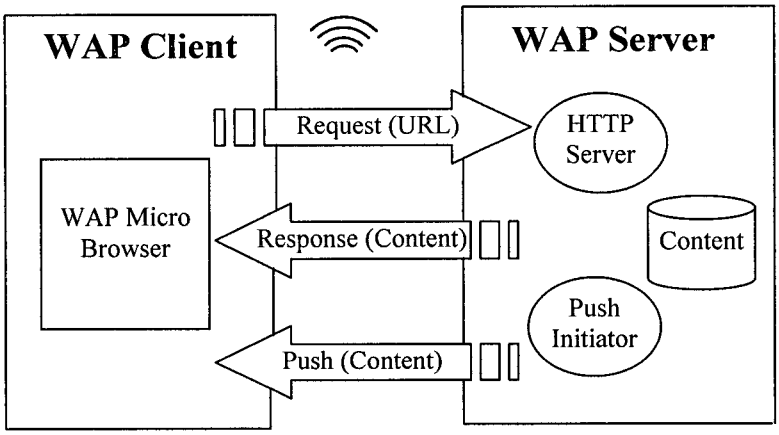


Figure 3.1: WAP Programming Model

3.1.2 WAP Proxy Model

WAP 2.0 does not require a WAP proxy since the communication between the client and the origin server can be conducted using HTTP/1.1. However, deploying a WAP proxy can optimize the communication process and may offer mobile service enhancements [2][3]. In addition, a WAP proxy is necessary to offer Push functionality. Figure 3.2 shows the typical WAP deployment scheme using the WAP proxy model.

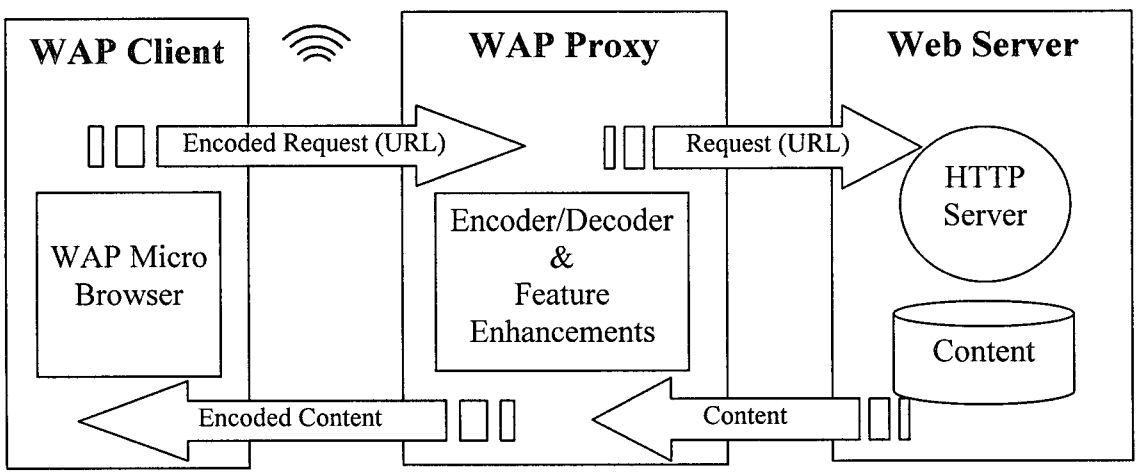


Figure 3.2: WAP Proxy Model

The WAP proxy may provide a variety of functions, including the following:

- **Protocol Gateway** The protocol gateway translates requests from a wireless protocol stack (e.g., the WAP 1.x stack) to the WWW protocols. The gateway also performs DNS lookups of the servers named by the client in the request URLs.
- **Content Encoders and Decoders** This can be used to translate WAP content into a compact format that allows for better utilization of the underlying link due to its reduced size.
- **User Agent Profile Management** User agent profiles describing client capabilities and personal preferences are composed and presented to the applications.
- **Caching Proxy** A caching proxy can improve perceived performance and network utilization by maintaining a cache of frequently accessed resources.

This proxy configuration ensures that mobile terminal users can access a wide variety of Internet content and applications. On the other hand, application developers can build content services and applications that run on a large base of mobile terminals. The WAP proxy allows content and applications to be hosted on standard WWW servers and to be developed using proven WWW technologies, such as CGI scripting.

The proxy model is used in our experiments for the introduced WAP 2.0 data compression proxy configuration. In order to evaluate the improvement from the advanced data compression scheme, both WAP 1.x stack and WAP 2.0 protocol stack without compression scheme are also presented as references. In the next sections, the legacy WAP 1.x and WAP 2.0 protocol network configurations are discussed, followed by our proposed proxy with an advanced data

compression scheme. Something to be mentioned is that although the nominal configuration of WAP service includes a web server, WAP proxy and WAP client, the WAP architecture can quite easily support other configurations.

3.2 Legacy WAP 1.x Protocol Configurations

The WAP 1.x configuration consists of a WAP device, WAP gateway and Web Server. The standard configuration is to use the gateway in the wireless network operator side. However under some extremely high security applications, an alternative configuration must be deployed which migrates the gateway to the application server provider side.

3.2.1 WAP 1.x Standard Configuration

In WAP 1.x, a WAP proxy (often referred to as a WAP gateway) is required to handle the protocol interworking between the client and the origin server. The WAP proxy communicates with the client using the WAP 1.x protocols (WSP, WTP, WDP/UDP), and it communicates with the origin server using the standard Internet protocols (HTTP and TCP). The standard configuration of WAP 1.x architecture is shown in Figure 3.3. The standard WAP 1.x configuration is also used as our test configuration for the performance of WAP 1.x.

The typical deployment of WAP 1.x architecture is shown in Figure 3.4, in which the wireless network operators host their own gateways and charge their subscribers for internet access through their pre-configured WAP-enabled phones.

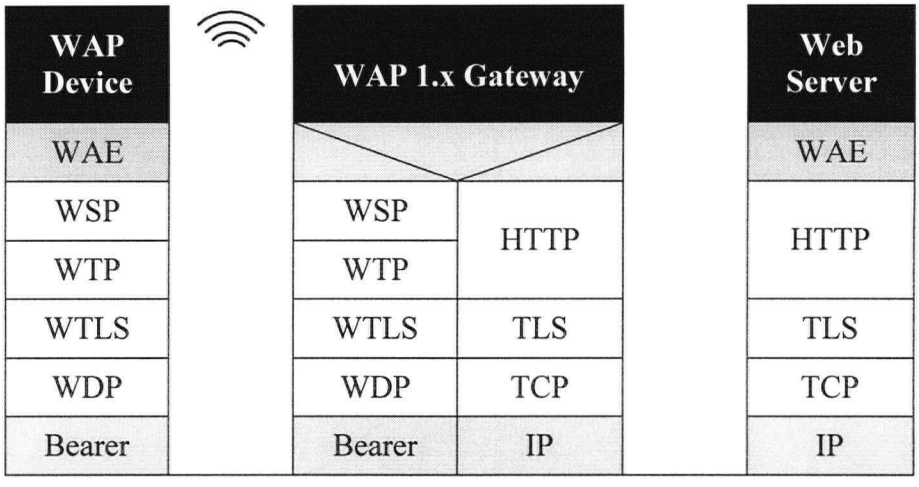


Figure 3.3: Standard WAP 1.x Network Configuration

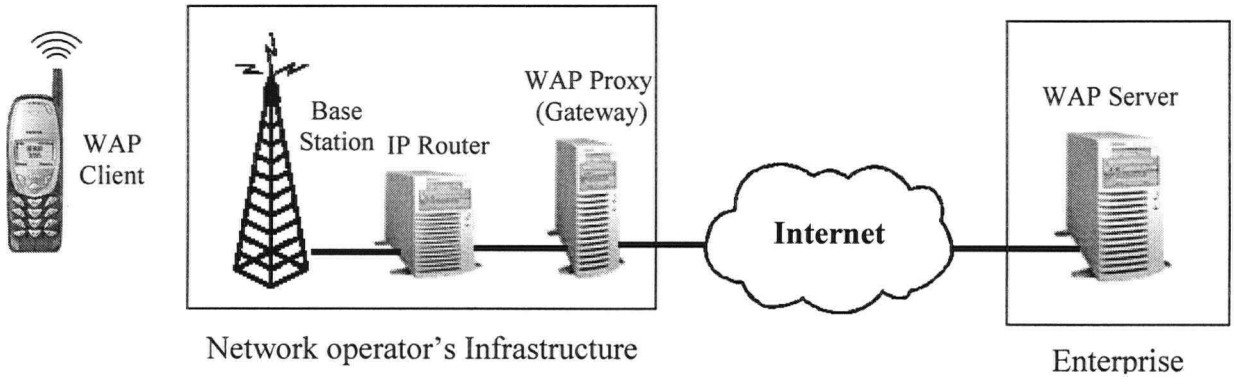


Figure 3.4: Typical WAP 1.x Deployment Scheme

The gateway is the most important part in WAP 1.x networks. A WAP 1.x gateway essentially implements both the WAP 1.x protocol stack and the IP stack in a single entry. It is used for the protocol conversion between these two protocol stacks. The gateway performs the following functions to facilitate a seamless protocol conversion:

- Conversion between WSP and HTTP
- Conversion between WTLS and TLS
- Encoding and decoding between text-based WML markup document in Internet domain and binary-encoded bytecode in wireless domain for transmission over wireless networks

- Compilation and handling of WMLScript
- Limited caching and content optimization facilities
- Session and state management and high level access control and authorization

3.2.2 Security Problems in WAP 1.x Gateway

Although WAP 1.x protocol conversion and content encoding minimizes the air-interface traffic, WTLS can only give end-to-end security between the gateway and the handset [21]. The gateway, which translates messages from one protocol to another, is a security "gray zone" for end-to-end applications.

Although all WAP messages are secured using the light-weight WTLS protocol between the WAP client and the gateway, these messages need to be decrypted and rendered in their original form in the gateway's memory before they can be converted to TLS and transmitted over the wired network. The same happens when a message arrives from the WAP server; it has to decrypt WML TLS-encoded messages, convert the content into binary format and encrypt it using WTLS, and then send it to the air interface. Although the conversion happens in the memory of the gateway and is completed quickly, the concept of end-to-end security between the WAP client and the application server is violated.

In some cases, companies and application providers need complete end-to-end security for their applications, e-commerce and e-banking are the most appealing examples. The only viable

solution for the WAP 1.x protocol stack is to deploy the gateway in its own secure private network with the WAP application server as shown in Figure 3.5.

In [14], this alternative configuration was evaluated against the standard configuration under varying IS-95 wireless link and Internet channel conditions. It was found that from the access delay point of view, the alternative configuration is plausible unless the wireless channel conditions are frequently bad with throughput under 1000 bps. The standard configuration performs better than the alternative configuration on bad wireless conditions, but its throughput is instead severely limited by degradations in Internet conditions.

Despite the feasibility of this alternative configuration, its drawbacks are also obvious. First of all, the wireless bearer must be IP-enabled. Secondly, the WAP application providers must setup their own gateway and auxiliary infrastructure such as modem pools for handling mobile users accessing the application. Aside from the hardware and human resources investment for each secure service provider, the WAP devices also have to be configured to switch gateways to access various secure WAP applications. The latter, like having to switch ISPs when accessing different web sites, is definitely undesirable for most users.

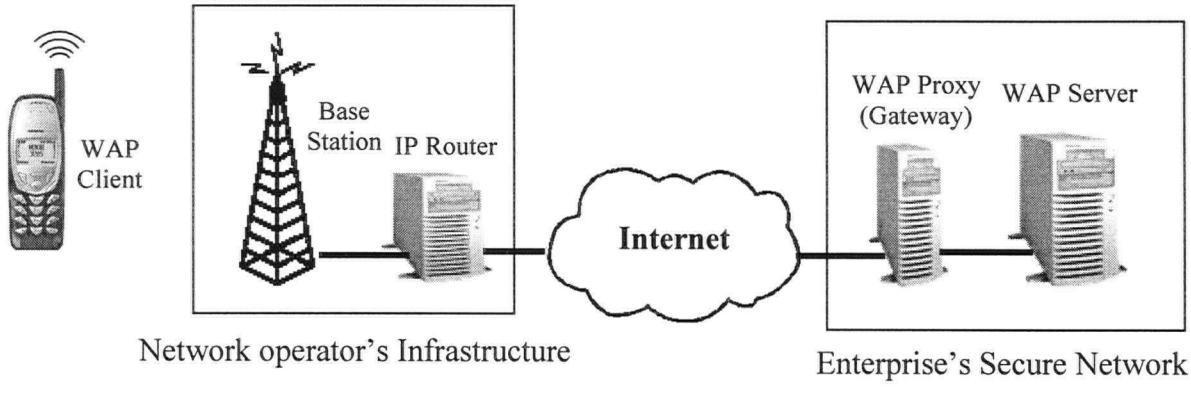


Figure 3.5: End-to-end Security Configuration in WAP 1.x Protocol Stack

3.3 WAP 2.0 Protocol Stack Configurations

In a WAP 2.0 proxy model, the HTTP proxy also works as the bridge between the WAP client and WAP server, as shown in Figure 3.6. This configuration locates the WAP Proxy between wireline and wireless networks; the proxy communicates with the WAP client using WP-TCP to enhance performance. In addition to TCP optimizations, the wireless profile of HTTP allows for further performance enhancements. Both profiles comprise well-defined IETF options that provide for efficient operation over wireless networks, as within the scope of WAP. The proxy then connects to the WAP server using Internet standard HTTP/TCP protocol.

As WPHTTP/WPTCP are interoperable with HTTP/TCP, no complex protocol conversion is required in WAP 2.0 proxy. WP-HTTP also supports the establishment of a connection-oriented tunnel using the CONNECT method. Thus end-to-end security at the transport level can be strictly guaranteed by TLS tunneling between mobile terminal and origin server, as shown in

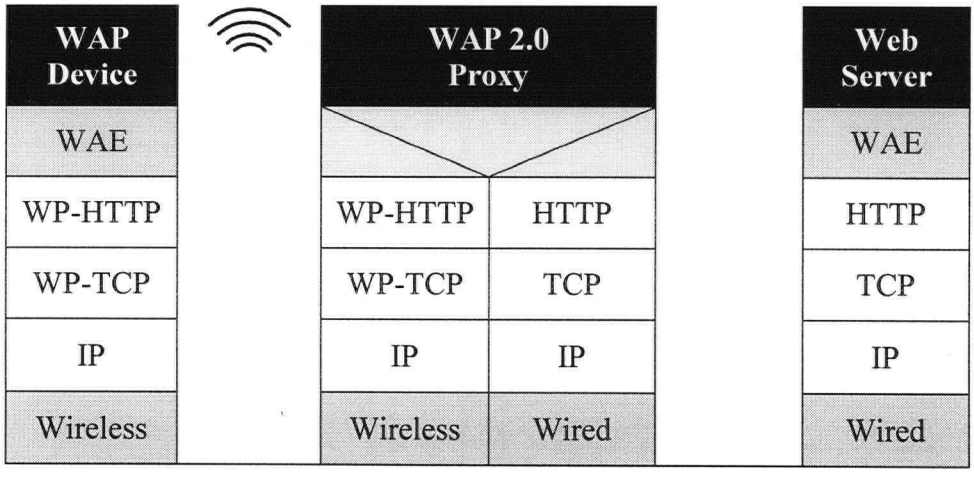


Figure 3.6: WAP 2.x HTTP Proxy Configuration

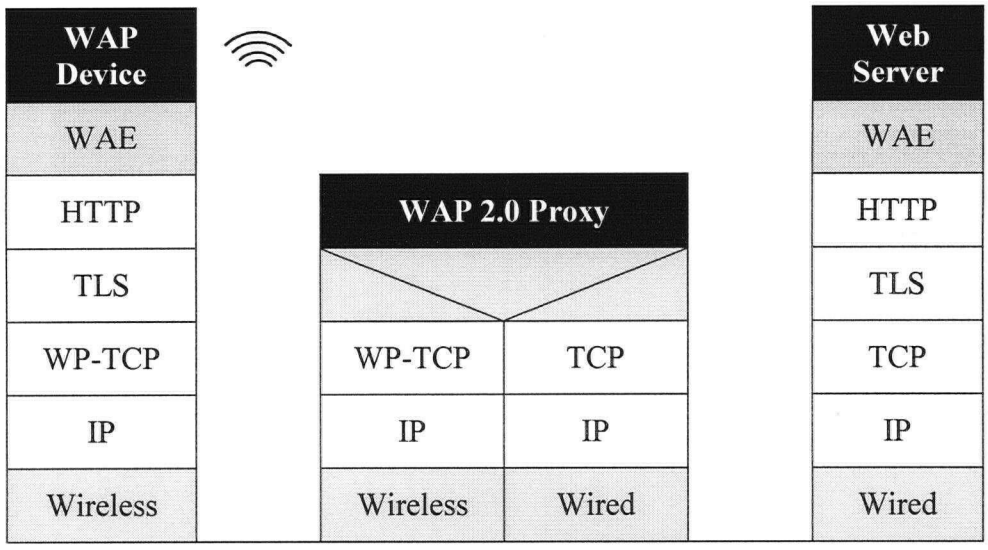


Figure 3.7: WAP 2.0 End-to-end Security with TLS Tunneling

Figure 3.7, which in turn eliminates the gateway security problem in the WAP 1.x stack. The client has a direct connection at the transport layer to the proxy, and the proxy has a direct connection at the transport layer to the WAP server. The proxy relays the data flow at the transport layer between two connections so that a direct TLS session between the client and the origin server is established. The proxy connects to the WAP server using standard TCP over the Internet and communicates with WAP clients using WP-TCP over the wireless domain. The WP-HTTP optimization is not available in this case because standard HTTP must be used to connect to the original WAP server. E-commerce is a compelling use case for end-to-end security.

3.4 Proposed Advanced Data Compression Scheme in WAP 2.0

Although the WAP 2.0 is an evolutionary step forward, since there is no such data encoding mechanisms as in WAP 1.x on the gateway, the transmitted packets are much larger than the encoded bytecode in WAP 1.x. In order to improve the performance of WAP 2.0, we propose a

novel architecture by introducing an advanced data compression scheme between the proxy and client to reduce the packet size and conserve bandwidth over the air-interface.

Figure 3.8 depicts our proposed WAP HTTP proxy. We introduce the data compression scheme between the WAP proxy and the WAP client over the basic configuration. The introduced advanced data compression scheme includes two separate compression processes: TCP content compression and Robust Header Compression (ROHC). This configuration minimizes the traffic over the wireless channel, while preserving the optimizations of WP-HTTP and WP-TCP.

Above the WP-TCP layer, we introduce a content compression and decompression process that compresses data at the TCP socket level. Since the compression works at the transport layer, it compresses all higher layer headers, including the HTTP header. This compression works better than when only HTTP content compression is employed at WP-HTTP, and results in a maximum content compression for IP packets. At the same time, ROHC is applied below the IP layer to

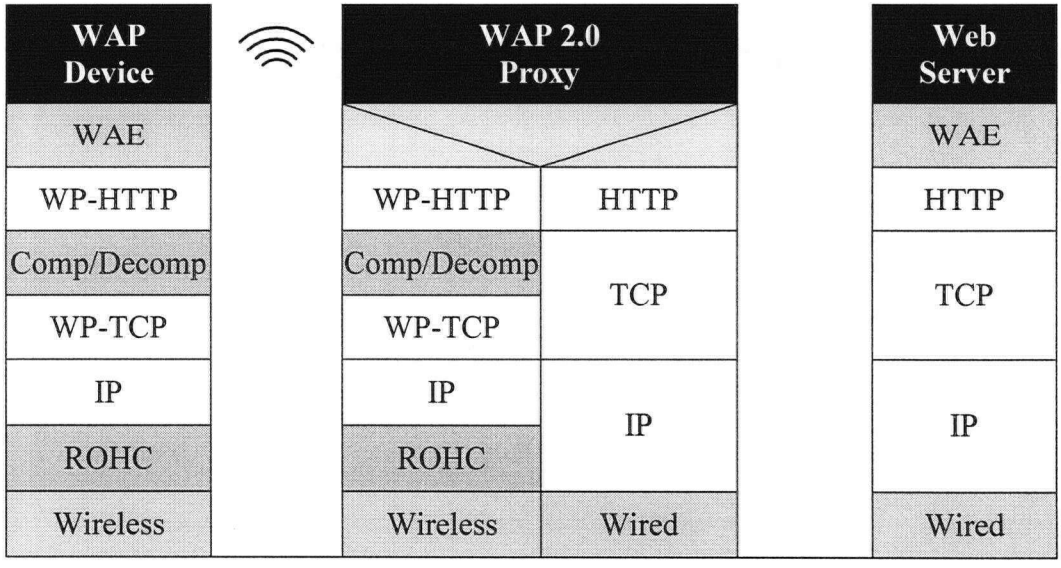


Figure 3.8: WAP 2.0 HTTP Proxy with Proposed Data Compression Scheme

compress the TCP and IP header. This combination gives the best compression results and minimizes the transmitted data over a wireless channel.

This configuration also guarantees strict end-to-end security by using the proxy at the transport layer (Figure 3.9). In this case, TLS tunneling is used between the client and the origin server. The proxy functions as a transport level data relay element and is isolated from the TLS session between the client and the origin server. The data compression scheme can be applied to minimize the air-interface traffic without breaking the end-to-end security requirements. The WP-HTTP optimization may not be available in this scenario. This configuration is used in our test experiments to evaluate the advanced data compression scheme.

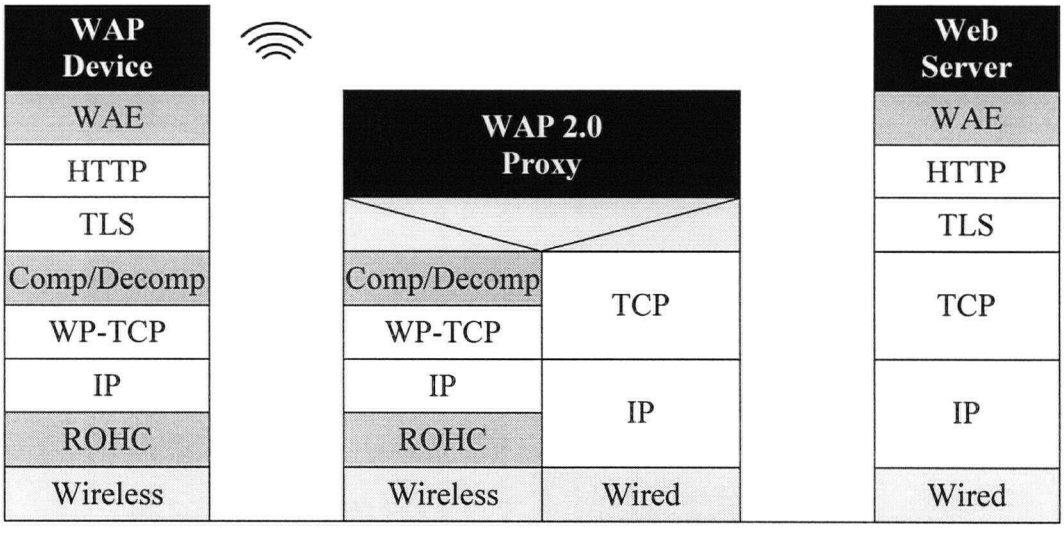


Figure 3.9: Data Compression Proxy supporting end-to-end Security with TLS Tunneling

3.5 Summary

In this chapter, the WAP model is discussed. By utilizing proxy technology, the WAP proxy model enhances performance and provides a variety of functions. In WAP 1.x configuration, the proxy, also known as the WAP gateway, must be present. However the presence of the gateway violates the concept of end-to-end security, and the only option is to host the WAP gateway on secure networks with the application server. WAP 2.0 is an evolutionary step forward which can overcome the WAP 1.x security problem by TLS tunneling, but the air-interface traffic is much larger than WAP 1.x's binary format WML code. A novel WAP 2.0 proxy configuration is introduced in this thesis, which inherits the advantages of the WAP 2.0 stack and minimizes the transmission data by using an advanced data compression scheme. The data compression scheme combines TCP content compression with robust header compression, and maximizes data compression without sacrificing end-to-end security.

Chapter 4: WAP 2.0 Data Compression Proxy Model Implementation

In order to evaluate the proposed WAP 2.0 data compression scheme, a test bed is setup to emulate WAP transactions using real WAP traffics. Another test bed for the WAP 1.x protocol stack is also implemented for comparison. The discussions emphasize the data compression scheme introduced in WAP 2.0.

Because both of the test configurations use proxies to connect the wireless domain and Internet domain, and because the Internet section is identical to both, for simplicity we omit the Intenet domain and assume no delay and no error over it. This allows our performance comparison to focus on the effects of the wireless channels. Therefore the simulation model consists of a WAP server, a WAP proxy (WAP gateway for WAP 1.x), a client, and an emulated wireless channel between the proxy and client.

To do the test experiments, a full functional WAP server and proxy (gateway) are built up. Phone emulators running on a PC are used to access the WAP server via proxy/gateway through the emulated wireless channel. The wireless channel simulates a wireless connection between a mobile terminal and a base station. It is implemented using Network Simulator (NS-2) running on a FreeBSD machine [26]. The FreeBSD operating system is used for best emulation compatibility since it has a built-in Berkeley Packet Filter (BPF), which is used to filter packets into the simulator. The internal process and implementation of the wireless channel will be described in detail later in Chapter 5.

4.1 WAP 1.x Simulation Test-Bed Implementation

The WAP 1.x network test bed is used to compare the performance of WAP 2.0 with the introduced data compression scheme. To set up the standard WAP 1.x protocol network, a test bed consisting of 4 machines running different operating systems is deployed. The Nokia WAP Toolkit, running under a Windows operating system is used as the WAP client [27]. It works as a WAP browser to access the WAP gateway. Between the browser and gateway, another machine running FreeBSD is used as a UDP timer to record the delay for WTP class 2 transactions. The timer can also be included in the Windows machine, but Windows can only give a time resolution of 10ms, which dose not satisfy our precision time recording requirement. UNIX systems are precise to nano-second, while we use milli-seconds for our delay timer. Between the UDP timer and the gateway, the wireless channel runs in NS-2 on FreeBSD. Since the Internet cloud is omitted, the WAP gateway and server are installed in one Linux box. The Kannel open source WAP gateway is deployed for the WAP gateway and Apache HTTP server is installed for the WAP server in our implementation [28][29]. The machines running for the UDP timer and wireless channel have 2 network interface cards. The IP forwarding in the

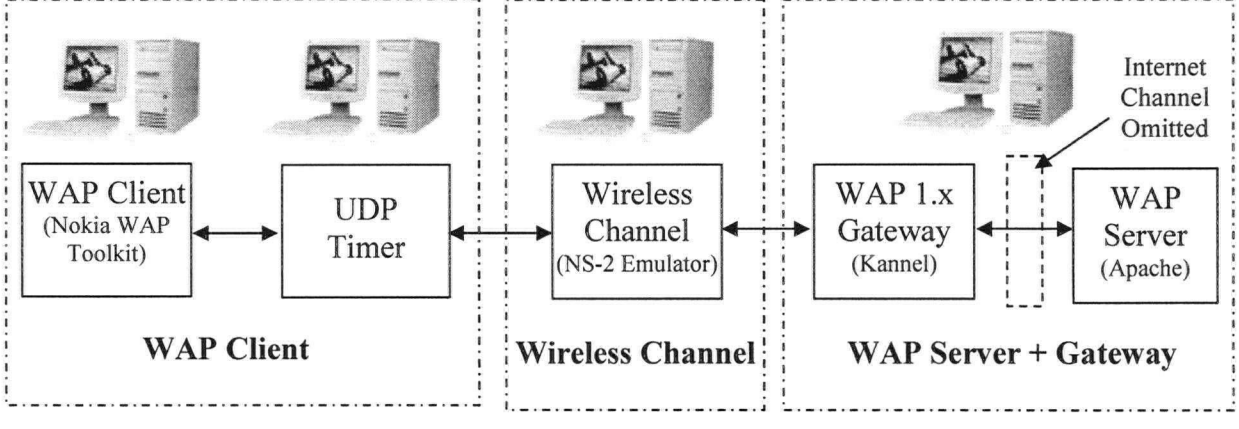


Figure 4.1: WAP 1.x Test Bed Configuration

machine for wireless channel is disabled, which lets the IP packets only get through via the emulated wireless channel. The test bed configuration and connection for WAP 1.x protocol stack network is shown in Figure 4.1.

4.2 WAP 2.0 with Data Compression Scheme Test-Bed Implementation

In the WAP 2.0 test bed, the WAP 1.x test bed machine connection configuration is reserved. The same wireless channel and WAP server is adopted as for WAP 1.x. A program is designed to work as the WAP 2.0 compression proxy. This proxy replaces the WAP 1.x gateway, and is deployed at the same machine with the WAP server since the Internet channel is removed in the test configuration. The proxy can do content compression and decompression according to user compression option input. The WAP client includes a WAP browser and the content decompressor. The Klondike WAP Browser from Apache Software Consulting Inc. is selected [30]. The browser is used to fetch the requested file using the HTTP 1.1 protocol. A content decompressor is implemented by another program running on another FreeBSD machine replacing the WAP 1.x UDP timer, it also works as a TCP timer to record the round-trip delay for each transaction, this delay is recorded at the socket level.

As described in previous chapters, the introduced data compression scheme includes content compression and ROHC, and the content compression and decompression are processed at the WAP 2.0 compression proxy and decompressor. ROHC is simulated in wireless channel and will be described in the next chapter. The content compression algorithm and ROHC process are depicted independently in the following sections. The test bed configuration for the WAP 2.0

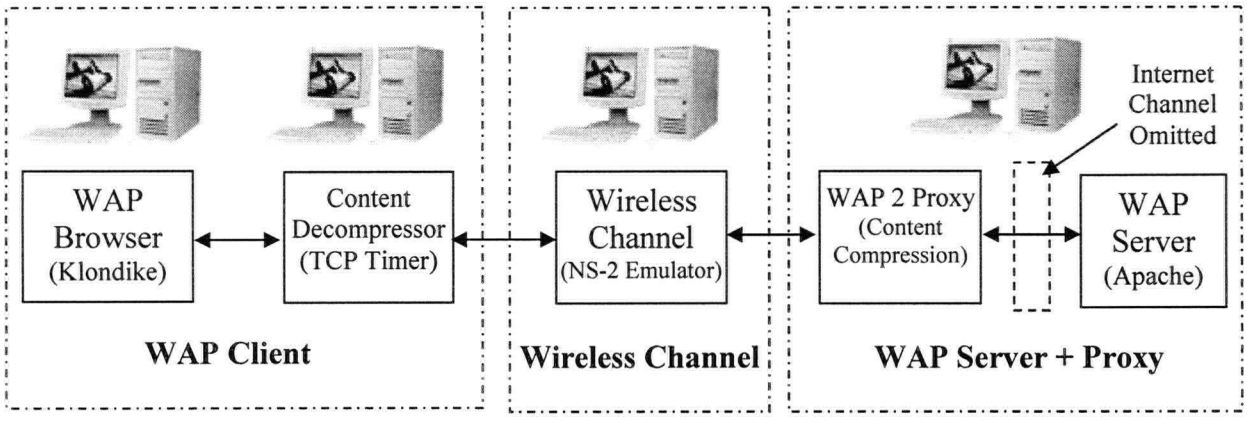


Figure 4.2: WAP 2.0 Test Bed Configuration

stack with the advanced data compression scheme is shown in Figure 4.2. When the content compression and ROHC options are disabled, the test bed becomes a standard WAP 2.0 proxy.

4.2.1 Content Compression Scheme

The content compression and decompression employed at the proxy and client is the deflate algorithm [31]. The deflate algorithm is probably one of the most famous lossless compression methods applicable to any type of data and is widely used in several compression utility programs, such as gzip. The deflation compression algorithm defines a lossless compressed data format that compresses data using a combination of the Lempel-Ziv 77 (LZ77) algorithm with secondary compression from Huffman encoding, with efficiency comparable to the best currently available general purpose compression methods. The data can be produced or consumed, even for an arbitrarily long sequentially presented input data stream, using only an apriori bounded amount of intermediate storage. The deflate compressor is given a great deal of flexibility as to how to compress the data. There are three modes of compression that the compressor has available [59]:

- Not compressed at all. This is an intelligent choice if the data has already been compressed. Data stored in this mode expands slightly, but not by as much as it would if it were already compressed and one of the other compression methods was used upon it.
- Compression, first with LZ77 and then with Huffman coding. The trees that are used to compress in this mode are defined by the Deflate specification itself, so no extra space is needed to store these trees.
- Compression, first with LZ77 and then with Huffman coding with trees that the compressor creates and stores along with the data.

The input data is divided into a series of blocks, then compressed into a set of successive blocks corresponding to it. The block sizes are arbitrary, except that non-compressible blocks are limited to 65,535 bytes. Each block is compressed using a combination of the LZ77 algorithm and Huffman coding. Each block is compressed using its own Huffman tree, which is independent of those used for previous or subsequent blocks; the LZ77 algorithm may use a reference to a duplicated string occurring in a previous block, up to 32K input bytes before. Each block consists of two parts: a pair of Huffman code trees that describe the representation of the compressed data part, and a compressed data part. (The Huffman trees themselves are compressed using Huffman encoding.) The compressed data consists of a series of elements of two types: literal bytes (of strings that have not been detected as duplicated within the previous 32K input bytes), and pointers to duplicated strings, where a pointer is represented as a pair <length, backward distance>. The representation used in the "deflate" format limits distances to 32K bytes and lengths to 258 bytes, but does not limit the size of a block, except for uncompressible blocks, which are limited as noted above. Each type of value (literals, distances,

and lengths) in the compressed data is represented using a Huffman code, using one code tree for literals and lengths and a separate code tree for distances. The code trees for each block appear in a compact form just before the compressed data for that block [31].

The compression and decompression is done in TCP socket streams using in-memory compression and decompression functions in the "zlib" compression library [32][33], which also includes integrity checks of the uncompressed data. The current "zlib" version of the library supports only the deflation compression method. Zlib is designed to be a free, general-purpose, lossless data-compression library for use on virtually any computer hardware and operating system. The zlib data format is itself portable across platforms. Unlike the LZW compression method used in Unix and in the GIF image format, the compression method currently used in zlib essentially never expands the data. (LZW can double or triple the file size in extreme cases.) Zlib's memory footprint is also independent of the input data and can be reduced, if necessary, at some cost in compression.

4.2.2 Robust TCP/IP Header Compression Scheme (ROHC)

Robust Header Compression was studied extensively in the literature [34][35][36]. ROHC will be used in all 3G cellular systems, substantially improving spectrum efficiency and service quality for IP services such as voice and video in the mobile Internet. The goal of ROHC is to develop generic header compression schemes that perform well over links with high error rates and long roundtrip times, as well as related signaling compression schemes. The schemes must perform well for cellular links built using technologies such as WCDMA, EDGE, and CDMA-

2000. However, the schemes should also be applicable to other future link technologies with high loss and long roundtrip times. Ideally, it should be possible to compress over unidirectional links.

The main reason why header compression can be done is the fact that there is significant redundancy between header fields, both within the same packet header, but in particular between consecutive packets belonging to the same packet stream. By sending static field information only initially, and by utilizing dependencies and predictability for other fields, the header size can be significantly reduced for most packets. Relevant information from past packets is maintained in a context. The context information is used to compress (decompress) subsequent packets. The compressor and decompressor update their contexts upon certain events. Impairment events may lead to inconsistencies between the contexts of the compressor and decompressor, which in turn may cause incorrect decompression. A ROHC scheme needs mechanisms for avoiding this inconsistencies and making the contexts consistent if they are not.

Existing header compression schemes ([57][58]) do not perform well over cellular links due to high error rates and long link roundtrip times, particularly as topologies and traffic patterns become more complex. In addition, existing schemes do not compress TCP options such as SACK or Timestamps. The ROHC Network Working Group in IETF specifies a highly robust and efficient header compression scheme in the standard tracking memo RFC 3095 [34]. Here follows is a brief overview of the compression and decompression process defined in it.

Header compression with ROHC can be characterized as an interaction between two state machines, one compressor machine and one decompressor machine, each instantiated once per

context. The compressor and the decompressor have three states each, which in many ways are related to each other, even if the meaning of the states is slightly different for the two parties. Both machines start in the lowest compression state and transit gradually to higher states. Transitions need not be synchronized between the two machines. In normal operation, only the compressor temporarily transits back to lower states. The decompressor transits back only when context damage is detected.

For ROHC compression, the three compressor states are the Initialization and Refresh (IR), First Order (FO), and Second Order (SO) states (Figure 4.3). The compressor starts in the lowest compression state (IR) and transits gradually to higher compression states. The compressor always operates in the highest possible compression state, under the constraint that the compressor is sufficiently confident that the decompressor has the information necessary to decompress a header compressed according to that state. Decisions about transitions between the various compression states are taken by the compressor on the basis of variations in packet headers, feedback from the decompressor, and periodic timeouts.

The decompressor starts in its lowest compression state, "No Context" and gradually transits to higher states. The decompressor state machine normally never leaves the "Full Context" state once it has entered this state (Figure 4.4). Initially, while working in the "No Context" state, the decompressor has not yet successfully decompressed a packet. Once a packet has been decompressed correctly (for example, upon reception of an initialization packet with static and dynamic information), the decompressor can transit all the way to the "Full Context" state, and only upon repeated failures will it transit back to lower states. However, when that happens it

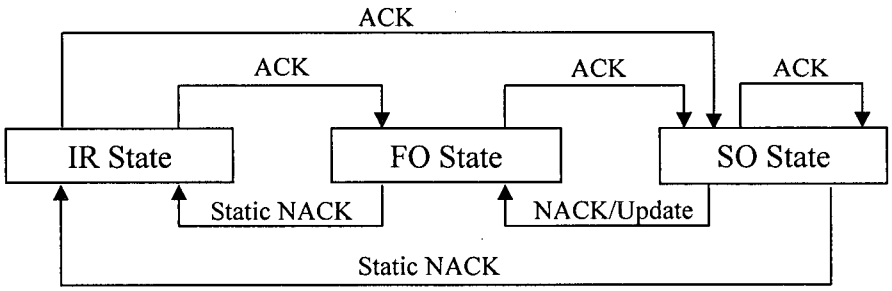


Figure 4.3: ROHC Compressor States

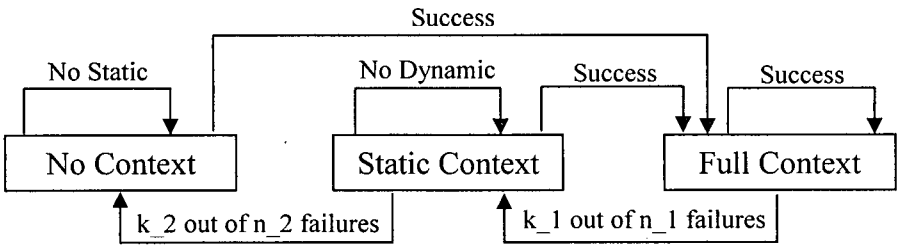


Figure 4.4: ROHC Decompressor States

first transits back to the "Static Context" state. There, reception of any packet sent in the FO state is normally sufficient to enable transition to the "Full Context" state again. Only when decompression of several packets sent in the FO state fails in the "Static Context" state will the decompressor go all the way back to the "No Context" state.

4.3 Summary

In this chapter, the proposed WAP 2.0 data compression proxy simulation model is illustrated. The test-bed configuration and running environment are specified and implemented to establish a fully functional WAP application network. The test bed implementation of the WAP 1.x protocol network is also described since WAP 1.x stack is used for result comparison. The deflate compression algorithm deployed for TCP content compression is introduced, and the principles of ROHC are also given. These two parts compose the proposed data compression scheme.

Chapter 5: Wireless Channel Modeling

The wireless channel forms the link between the client and the proxy. Most of the efforts in WAP are designed to optimize the performance on the wireless link. The wireless channel is implemented in NS-2 using emulation functions. In this chapter, a brief wireless communication history and associated techniques are presented, and the characteristics of wireless channels are discussed. IS-95 and CDMA 2000 1xRTT are chosen as the "test" wireless channels. The physical channel structures of both IS-95 and CDMA 2000 1xRTT are briefly described. The simulation model of the wireless channel is also described which can be used to simulate both the IS-95 channel and the CDMA2000 1xRTT channel.

5.1 Wireless Communications Background

Wireless communications have evolved remarkably since Guglielmo Marconi first demonstrated the wireless telegraph to the English telegraph office and was awarded the patent in 1897. However the growth of wireless communications was quite slow until Bell Laboratories developed the cellular concept in the 1960s and 1970s. The booming of cellular radio and personal communication systems has occurred only in the last 15 years. Cellular systems evolved from first generation analog systems to second generation (2G) digital cellular system, and now carriers worldwide are upgrading their networks to higher bandwidth and speed networks. The wireless industry has made a great deal of effort to develop solutions for 2.5G and third generation (3G) wireless systems which provide subscribers with high-speed Internet and multimedia services.

Due to technical, regulatory and historical reasons, various analog and digital system standards have been proposed around the world, for example, AMPS and TACS for analog systems, GSM and IS-95 for 2G systems, 2.5G GPRS system. In IMT-2000 3G standards, there are 2 major groups: WCDMA (UMTS) and CDMA 2000. Different standards may employ different modulation method, multi-access scheme, use different spectrum, and provide different data rates and quality of service (QoS).

In order to allow many users to share the finite amount of radio spectrum simultaneously, multi-access schemes are used to achieve high capacity. Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA) and Code Division Multiple Access (CDMA) are the three major multi-access techniques.

CDMA is based on spread spectrum technology. In CDMA systems, the narrowband message signal is multiplied by a very large bandwidth signal called spreading signal. This spreading signal is a pseudo-noise code sequence that has a chip rate which is orders of magnitudes greater than the data rate of the message. All users use the same frequency and may transmit simultaneously. Each user has his own pseudorandom codeword which is approximately orthogonal to all other codewords. The receiver performs a time correlation operation to detect only the specific desired codeword. All other codewords appear as noise due to decorrelation. For detection of the message signal, the receiver needs to know the codeword used by the transmitter. Each user operates independently with no knowledge of the other users. CDMA has some advantages over FDMA and TDMA including frequency reuse, soft capacity, anti-jamming, anti-interference, soft handoff and so on. At the same time, CDMA has its own problems

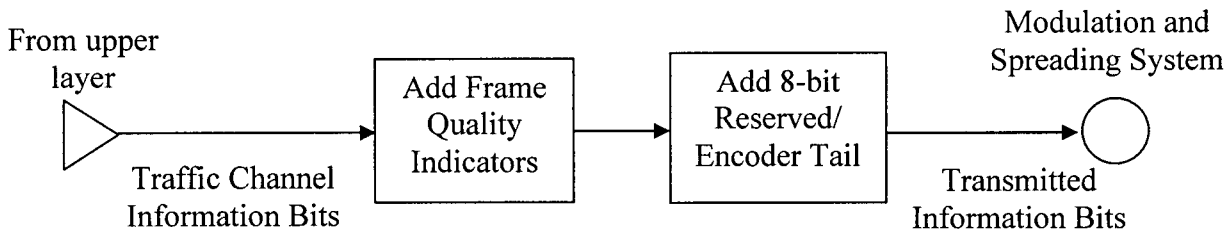
including cross-correlation, which increases the noise level, and near-far problem, which requires the deployment of power control algorithms.

WAP is designed to operate over a variety of bearers, including short message, circuit-switched data and packet data. Since the mobile cellular system is the most widely used, we evaluate the WAP performance on some typical 2G and 3G cellular systems. We select CDMA systems, which are widely used in North American and are used in all 3G networks. The IS-95 standard for 2G and CDMA 2000 for 3G systems are selected for the test bed. Since we are testing data transmission in our experiments, we will focus on the traffic channel only in the coming sections with emphasis on the data transmission interface to upper layers. The following two sections give a simple description of traffic channel characteristics and structure of these two standards.

5.2 IS-95 CDMA Standard

IS-95 (Interim Standard 95) is a U.S. 2G digital wireless standard proposed by the U.S. Telecommunications Industry Association (TIA) based on Direct Sequence CDMA [37]. IS-95 channel occupies 1.25MHz on each one-way link with a channel chip rate of 1.2288 Mcps, and supports variable data rate in real-time with a maximum of 9600 bps.

The traffic link is capable of accepting information bits at the rate of 0.8 Kbps, 2.0 Kbps, 4.0 Kbps and 8.6 Kbps. Channel symbols are transmitted in 20ms frames, with each frame containing a number of information bits and a 8-bit convolutional encoder tail sequence to drive the convolutional encoder into a known state at the end of each frame. Additionally, the frame



Bits/Frame	Data Rate
172 bits	8600 bps
80 bits	4000 bps
40 bits	2000 bps
16 bits	800 bps

Bits
12-bit
8-bit

Bits/Frame	Data Rate
192 bits	9600 bps
96 bits	4800 bps
48 bits	2400 bps
24 bits	1200 bps

Figure 5.1: Information Bits Transmission in IS-95

incorporates a frame quality indicator sequence (CRC encoder) for the two highest data rates, 12-bit and 8-bit respectively. The frame quality indicator sequence is nothing but a set of parity check digits that detect errors in the frame. It is also used to establish the incoming data rate at the subscriber station. This results in transmitted information bits at the rate of 1200, 2400, 4800, and 9600 bps respectively, as shown in Figure 5.1.

IS-95 uses different modulation and spreading techniques for the forward and reverse links. For example, the forward traffic link utilizes a convolutional code with a code rate of 1/2 and BPSK modulation, while the reverse channel employs a convolutional code with a code rate of 1/3 and offset QPSK.

5.3 CDMA2000 1xRTT Standard

Developed by The Third Generation Partnership Project 2 (3GPP2) [38], CDMA 2000 Radio Transmission Technology (RTT) is a wideband, spread spectrum radio interface that uses

CDMA technology to satisfy the needs of 3G wireless communication systems. CDMA2000 provides full backward compatibility with TIA/EIA-95-B. Backward compatibility permits CDMA2000 infrastructure to support TIA/EIA-95-B mobile stations and permits cdma2000 mobile stations to operate in TIA/EIA-95-B systems.

This first phase of CDMA2000, also known as CDMA2000 1xRTT, employs 1.25MHz of bandwidth and is designed to double current voice capacity and support always-on data transmission speeds 10 times faster than that typically available today, some 153.6 kbps on both the forward and reverse links. This speed exceeds the 144 kbps 3G peak throughput threshold defined by the ITU.

Technologies known as CDMA 2000 1xEV-DO (Evolution of CDMA 2000 Data Only) uses a separate 1.25MHz carrier, and can offer a peak rate of 2.4Mbps. Phase two of CDMA 2000, also known as 3xRTT will use 5MHz (or 3 times the 1.25Mhz) of bandwidth. The first commercial CDMA2000 networks were launched in South Korea in early 2001 and are already providing service to over one million paying subscribers. Now it has been commercially launched in several companies worldwide. In Canada, Telus and Bell Mobility have already released their 1xRTT services.

The CDMA2000 architecture is shown in Figure 5.2 [39]. The upper layers contain three basic services: voice services, end-user data-bearing services and signaling. The link layer provides varying levels of reliability and QoS characteristics according to the needs of the specific upper layer service. It gives protocol support and control mechanisms for data transport services and

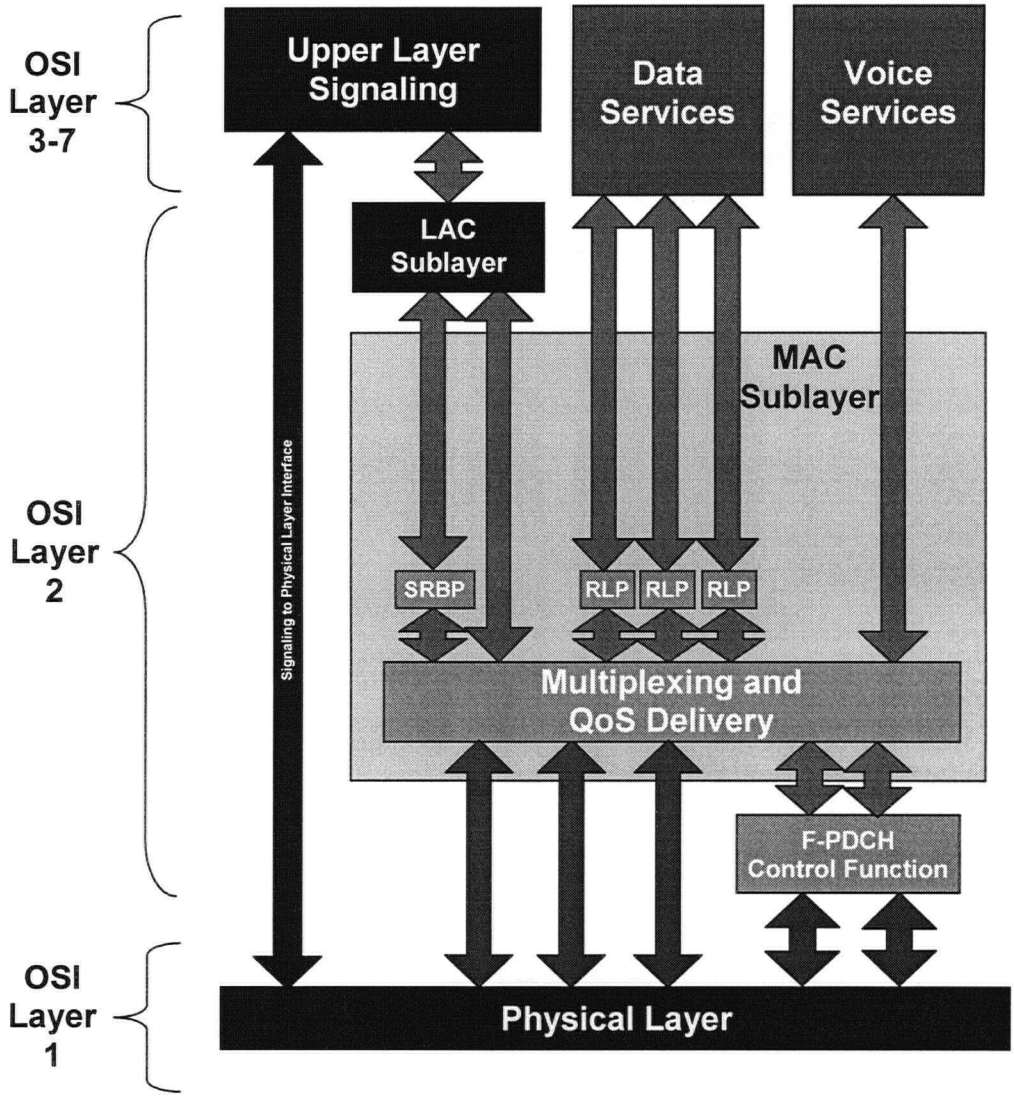
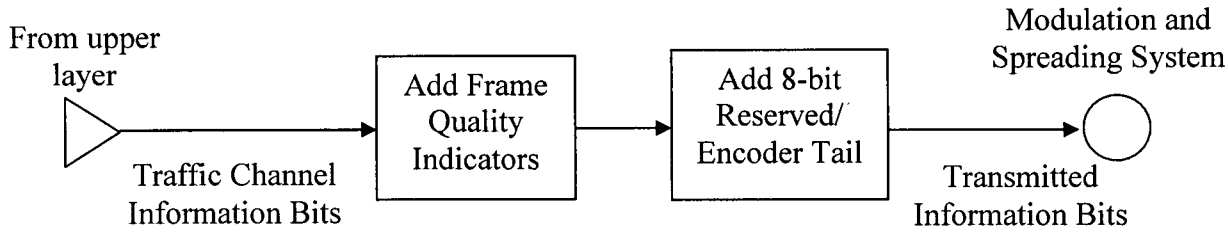


Figure 5.2: CDMA 2000 Architecture

performs all the functions necessary to map the data transport needs of the upper layers into specific capabilities and characteristics of the physical layer. The link layer is subdivided into Link Access Control (LAC) and Media Access Control (MAC) layer. The Radio Link Protocol (RLP) provides best effort data delivery with ARQ scheme. Multiplexing and QoS sublayer combines/separates application instances' traffics to/from physical layer blocks. The physical layer provides coding and modulation services for these blocks. Since our experiments focus on data service, only the traffic channel is mentioned. The modulation, spreading, I-Q mapping



Bits/Frame	Bits	Data Rate
16 Bits	6	1.5 kbps
40 Bits	6	2.7 kbps
80 Bits	8	4.8 kbps
172 Bits	12	9.6 kbps
360 Bits	16	19.2 kbps
744 Bits	16	38.4 kbps
1512 Bits	16	76.8 kbps
3048 Bits	16	153.6 kbps

Figure 5.3: Forward/Reverse Supplemental Channels in CDMA2000 1xRTT

schemes are ignored in this thesis. Figure 5.3 depicts the physical channel structure of the Forward Supplemental Channel (F-SCH) and Reverse Supplemental Channel (R-SCH) that are used for high-speed data traffics [40]. Both forward and reverse channels use Spreading Rate 1 and radio configuration 3 (RC-3) to achieve the 153.6 kbps peak rate in CDMA2000 1xRTT. The data are transmitted in 20ms frames with a R = 1/4 Turbo encoder. In the reverse link, BPSK modulation is used with a pilot, while in the forward link QPSK pre-spreading symbols are used and Transmit Diversity is allowed.

5.4 Errors in Cellular Wireless Channel

Lossy behavior is the most important characteristic of wireless links. While a bit error rate (BER) as high as 1e-3 must be accepted to efficiently utilize radio resources, in severe operating conditions, the BER can be as high as 1e-2. Errors in wireless channels mainly come from fading and interference. Fading happens due to random phase from multi-path time delay spread and

Doppler shift from the vehicular speed. Depending on the parameters of the transmitted signal and the channel, different transmitted signals will suffer different types of fading. Multi-path delay spread leads to flat fading and frequency selective fading, Doppler spread leads to fast fading and slow fading. The multi-path fading on mobile radio channels can be modeled by a Rayleigh distribution. In a cellular system, there are two major types of system generated cellular interference: co-channel interference and adjacent-channel interference. Co-channel interference comes from the frequency reuse in a given coverage area. Adjacent channel interference results from imperfect receiver filters which allow nearby frequencies to leak into the passband.

Several techniques are employed to restrict the errors on the channel within an acceptable bound. Since the errors on wireless channels tend to occur in bursts, block interleaving is employed to mitigate the effects of burst errors. Interleaving helps in spreading the burst errors over a larger data fragment by interleaving the data bits. Various modulation techniques are developed for data transmission. Several signal processing techniques, such as equalization, diversity and channel coding can improve the received signal quality. Equalization compensates for intersymbol interference (ISI) created by multipath within time dispersive channels by using adaptive filtering at receiver. Diversity techniques use the fact that deep fades on one signal path do not necessarily imply that all radio paths are equally faded. The receiver can recover the signal from several paths simultaneously, thus providing an improvement in the received signal SNR. Channel coding techniques insert redundancy into transmitted data streams so that the receiver can detect and possibly correct errors that occur during transmission. Forward Error Correction (FEC) and Cyclic Redundant Check (CRC) are widely used in wireless communication networks. Convolutional code is the most commonly used FEC coding method.

Viterbi algorithm, the optimal decoding algorithm for convolutional code, is used at the channel decoder. This technique is applied in both the IS-95 and the CDMA2000 systems. Instead of convolutional code, turbo code is used on SCH for high-speed data services in CDMA 2000 networks. Power control is another important technique used in cellular systems which can dramatically reduces the reverse channel S/I in the system, it is especially important in CDMA systems since all users transmit data over the same spectrum.

Modeling errors on wireless channels has been well studied in the literature. In [41], a bit error model has been developed based on error sequences derived from waveform simulations of wireless link performance, with various modems operating under varying propagation, noise, and interference conditions that enable simulations of the digital error performance of wireless communication links.

In most CDMA systems, data to be transmitted are grouped into blocks (frames), for example, both the IS-95 and the CDMA 2000 networks use 20ms physical layer frame for data services. Frame Error Rate (FER) or Block Error Rate (BLER) is another wireless link performance measurement. To the upper layers, FER is more realistic than BER since the use of interleaver and coding techniques can lead to the detection and recovery of some bit errors. In [42], an analytical method is proposed to evaluate the average block error probability in a DS/CDMA packet radio transmission system under a Rayleigh fading channel, and the result is derived as a function of the number of total interfering users in each group. Several works have used first-order Markov chains to model block error processes in transmissions over wireless channels [43][44][45]. Both the block Markov model (BMM) and hidden Markov model (HMM) have

been suggested as approaches to it. The equivalent of these 2 approaches is discussed in [46]. In certain sets of parameters, the Markov chain leads to a unique stationary distribution, which means a uniform FER. FER is employed in our experiments to evaluate performance under various FER conditions. The FER parameter represents the unrecoverable error rate after the decoder. The frame is considered erroneous and needs to be retransmitted when error occurs, as the error cannot be corrected by the FEC decoder.

5.5 Emulation Channel Deployment

In our experiments, we use a WAP client to access the WAP server in the test bed, the live packets go through the emulation channel which simulate a real-life wireless channel. Emulation refers to the ability to introduce the simulator into a live network. The Network Simulator 2 (NS-2) [26] is used to emulate the packet level behaviors of IS-95 and CDMA2000 1xRTT wireless channels. Special objects within the simulator are capable of introducing live traffic into the simulator and injecting traffic from the simulator into the live network.

5.5.1 Emulation Principle

NS2 has an excellent built-in support for network emulation [47]. In emulation mode, a special version of the system scheduler is used: the RealTime scheduler. This scheduler implements a soft real-time scheduler which ties event execution within the simulator to real time.

The interface between the simulator and live network is provided by a collection of objects including tap agents and network objects. Tap agents embed live network data into simulated packets and vice-versa. Network objects provide access to a live network. Generally, network objects are installed in tap agents and provide an entry point for the sending and receipt of live data at a particular protocol layer (e.g. link, raw IP, UDP, etc.) and with a particular access mode (read-only, write-only, or read-write). Each tap agent can have at most one associated network object, although more than one tap agent may be instantiated on a single simulator node. The simulator provides Pcap/BPF Network objects. These objects provide an extended interface to the LBNL packet capture library (libpcap). This library provides the ability to capture link-layer frames in a promiscuous fashion from network interface drivers. It is optimized for use with the Berkeley Packet Filter (BPF), and provides a filter compiler for the BPF pseudomachine machine code. The simulator taps the packets entering the PC from a Network Interface Card (NIC) via Pcap/BPF filter. The packets are then filtered so that only the traffic from the desired PCs passes through the simulator. The internal functioning of a typical ns2 emulation process is depicted in Figure 5.4.

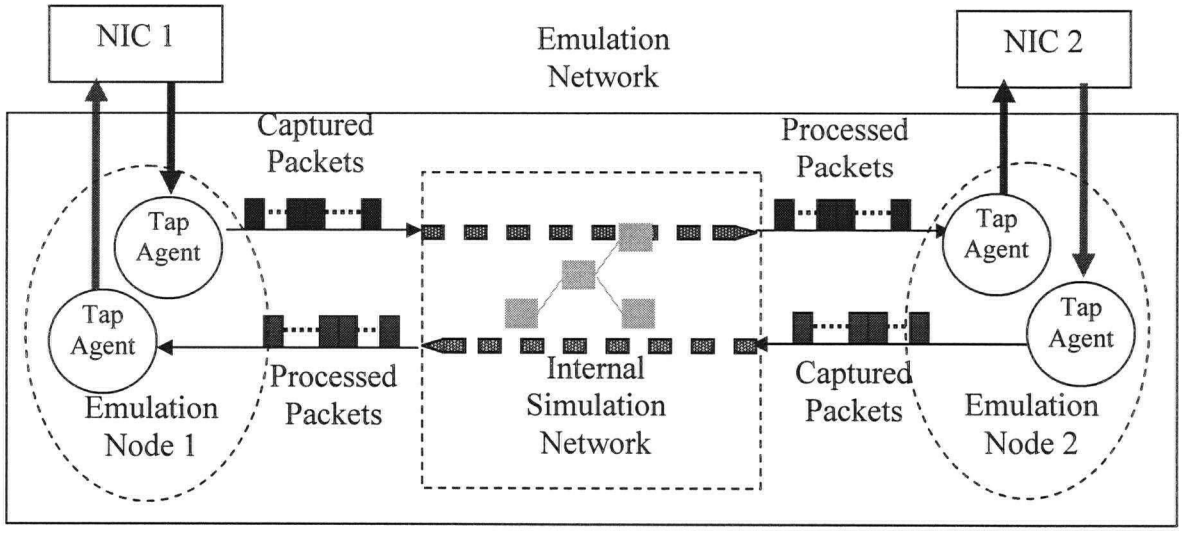


Figure 5.4: Typical NS-2 Emulation Process

5.5.2 Wireless Channel Emulation

The wireless channel is emulated in NS-2 with cmu-mobile extensions. Emulation is not supported in the original NS-2 system. Modifications are made on the source code to let live traffic get through the mobile networks. The emulated channel consists of two nodes, one BaseStationNode for the base station, and one MobileNode for the mobile client, as shown in Figure 5.5. Each node has 2 tap agents and 2 network objects on it. One tap agent is used to capture live packets from the NIC via a Pcap live network object; the other tap agent is used to write the processed packets from the other node to the NIC via an IP network object. Thus, 2 simplex links are formed between two NICs by the tap agents to process the packets in both directions. The captured packets are then fragmented at the link layer and transmitted between the nodes. An error model and retransmission mechanism is used in the channel. After all fragments are transmitted and received by the other node, they are defragmented and injected back to live network. The packet transmission process and error model inserted into the

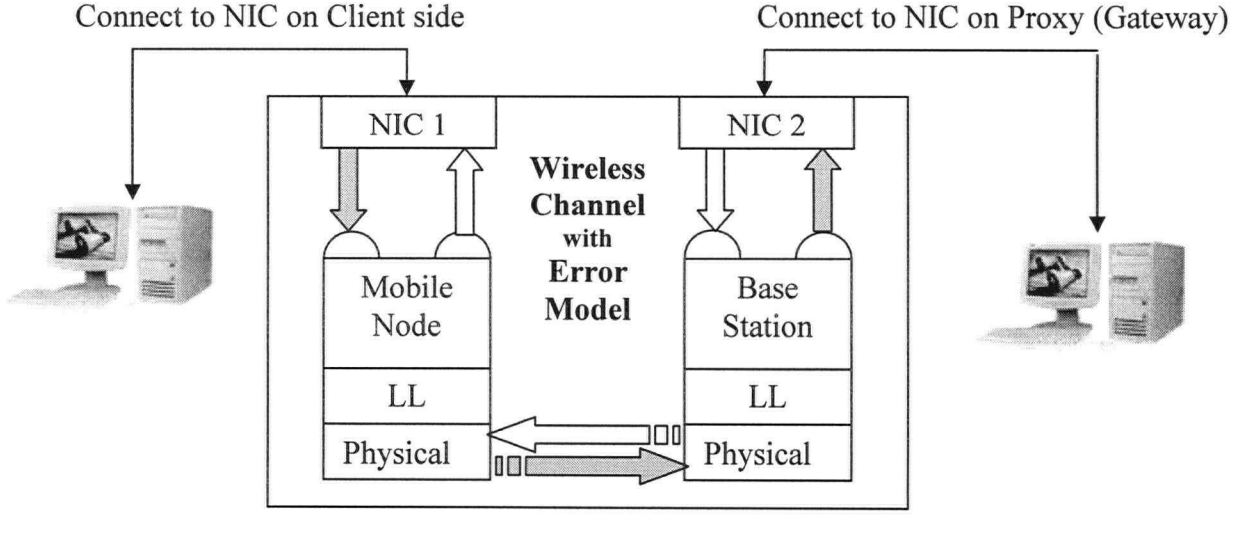


Figure 5.5: Wireless Channel Emulator Structure

processed packets are described later.

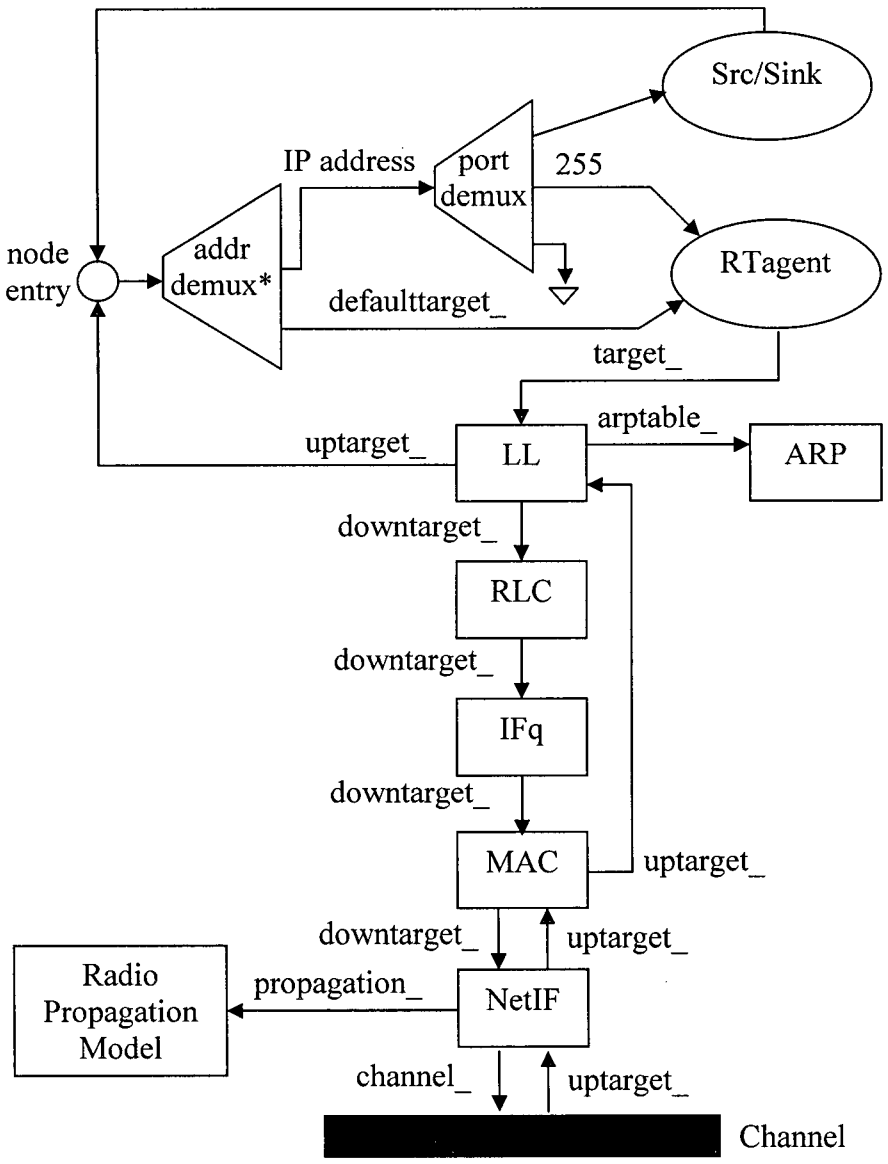
Since there is only one mobile node in our experiments, NS-2 NOAH extension by Jörg Widmer (widmer@icsi.berkeley.edu) is installed for simplicity. NOAH is a wireless routing agent that only supports direct communication between base stations and mobile nodes. This allows for the simulation of scenarios where multi-hop wireless routing is undesired. Furthermore, NOAH does not send any routing related packets.

MobileNode is the basic NS Node object with added functionalities such as movement and the ability to transmit and receive on a channel. This allows it to be used to create mobile, wireless simulation environments. BaseStationNode is essentially a MobileNode supporting Hierarchical addressing. It is a gateway for the wired and wireless domains and is responsible for delivering packets into and out of the wireless domain.

The network stack for a mobilenode consists of a link layer (LL), an ARP module connected to LL, an interface priority queue (IFq), a mac layer (MAC), and a network interface (netIF), all connected to the channel. The default LL is a simply a link delay from the send target to the receive target without the potential functionalities. The default MAC in the NS-2 wireless simulation is the IEEE 802.11 distributed coordination function MAC protocol.

In our emulation, we employed some given settings in NS-2 by using Channel/WirelessChannel for Channel type, Propagation/TwoRayGround for radio-propagation model, Phy/WirelessPhy for network interface type and Antenna/OmniAntenna for antenna model.

In order to simulate the wireless channel characteristics of IS-95 and CDMA 2000, we modified the default LL and MAC layers in NS-2. The modified schematic diagram of mobile nodes is shown in Figure 5.6. Adapting from the GPRS patch by Richa Jain, a new LL sublayer called Radio Link Control (RLC) is added to the wireless node, between the LL and IFQ. The main features included at RLC are fragmentation and assembly, along with fragment retransmissions.



* Hierarchical addressing in BaseStation Node

Figure 5.6: Schematic of Modified MobileNode and BaseStaionNode

The retransmit mechanism is a simplified form of Selective Repeat ARQ. The fragmentation-reassembly and acknowledgements may be configured ON or OFF by the user and the RLC fragment size can be set. In case an RLC fragment is dropped by the MAC, in acknowledged mode, a duplicate acknowledgment for the last correctly received RLC fragment is sent back to the sender, which then retransmits the expected RLC fragment. In unacknowledged mode, if an RLC fragment is missing, the RLC does not pass any of the lost fragments to the LL. If the LL ARQ is employed, it will retransmit the whole LL packet, otherwise the higher layers, in our case TCP will handle it. The RLC layer retransmission is especially important when the wireless link FER is high and link bandwidth is low.

In IS-95, the full rate is 172 information bits in each 20ms frame as shown in Figure 5.1. The RLC fragment size is set to 21 bytes, which is 168 bits. The addition of the 12 bits of CRC and 8 bits of encoder tail from physical channel result in a transmission rate of maximum 9600 bps.

The CDMA2000 system also uses 20ms data frames in the traffic channel F/R-SCH. Each 20ms data frame consists of 381 bytes (3048 bits) of data with 16 bits of frame quality indicator and 8 bits of Reserved/Encoder Tail bits. In emulation, 5ms frames are used instead of 20ms frames to simulate the MAC layer multiplexer and to have more frames for the error module. Since the error module is used to generate frame error rate, if too few frames are used in the emulation, the error will be poorly distributed. In addition, 5ms frames are used for control information in CDMA2000 networks. Thus, the RLC fragment in CDMA2000 1xRTT channel is set to 95 bytes. The 16 bits of frame quality indicator and 8 bits of encoder tail for each 20ms frame, result in a speed near the maximum transmission rate of 153.6 kbps.

Since in CDMA systems all users use the same radio channel, there is no contention between users, so the MAC layer is simplified using a slotted structure. Each slot is set to be 20 ms in IS-95 and 5ms in CDMA2000 1xRTT, as in the preceding two paragraphs. If there is RLC packet to be transmitted in the Ifq, it will be transmitted in the next available timeslot. One timeslot can transmit only one RLC frame. In order to avoid control packets, such as ARP and routing packets being transmitted in the traffic channel, which will cause delay in the data traffic, a separate channel is used to transmit these control packets.

5.5.3 ROHC Implementation in Emulator

Because the channel implemented in our emulation only simulates the wireless transmission behavior, ROHC should be implemented on machines connecting to the emulation channel. However, the emulation uses a Pcap/BPF packet filter, and filters the packet according to the IP header information, e.g., IP address, port number. If the IP header is modified by ROHC, the packet filter cannot recognize the header information; thus the emulation wireless channel does not work. On the other hand, the ROHC should only be applied on the wireless link between the proxy and the WAP client, so simulating the ROHC in a wireless channel is a reasonable approach.

To overcome this problem, ROHC is simulated in the wireless channel by giving the first LL fragment a bigger size than the others. One static variable is defined to set the header compression bytes. This assumes an average header compression ratio that is statistically stationary and fixed in a long run.

5.5.4 Error Model

In our emulation, the emulated packets are captured, processed and then injected back into the live network at the link layer, therefore the channel error model should be introduced in the emulator under LL. The error model provided by NS-2 can introduce errors into packets created by Agents such as TCP or UDP. It cannot introduce errors at the lower layers. But the packets captured are not attached to any agents in the emulation channel. In order to simulate the errors at lower levels, an error model is introduced that produces random errors in the emulated RLC PDUs, which can have errors and retransmit on physical layer frames. This leads to frame error, so FER with a uniform parameter is used to simulate wireless channel errors in our experiments.

This error model marks a randomly chosen RLC frame as erroneous. This causes the RLC packet to be dropped at its destination. At each drop, a random number is generated to decide the next drop. Since the uplink and downlink in wireless channels may not have the same modulation and coding shemes, the error characteristic may be differnent. Two static integer variables are defined in the emulation program to set different error rates for reverse and forward links. Suppose we give an integer n for the drop rate variable, and m is the generated random number for droping, which is evenly distributed between 1 and n. The propability of any number between 1 and n is $1/n$. If the random number m=1, then the next frame after the current transmitted one will be dropped, the drop rate is 100%, if m=2, drop rate is 1/2, and so on. So the FER obtained by an integer n is calculate as follows:

$$\varepsilon(n) = (1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}) * \frac{1}{n} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{i} \qquad (5.1)$$

In our experiements, the wireless channel is emulated for good and bad error conditions. Five different FER settings are given ranging from 1% to 40%. The settings for drop rate variable are summarized in the table below. For simplicity, we use the same FER on both forward and reverse channels.

FER	1%	5%	10%	20%	40%
Error Rate Variable n	715	105	43	17	6

Table 5.1: Wireless Channel FER Setting in Emulation

The deployment of the IS-95 and CDMA2000 systems in our emulation can simply be configured by the variables defined in the tcl script running in the NS-2 simulator environment. Both the frame time and size can be reconfigured to fit other network standards. The ROHC header compression ratio can be modified by changing the relevant header compress variable. Different FER on forward and reverse channels can easily be set as well. The Link layer and the RLC layer ARQ can be enabled and disabled also. This emulation channel can be used for any IP based traffic. Both UDP and TCP transport layer packets are accepted. The channel can be used to do emulation testing for various wireless link throughput and error patterns with minimum modifications. Mobile node movement is supported, but no handoff functions are available now in our emulation channel, since only one base station is present.

5.6 Summary

In this chapter, the implementation of the wireless channel between the WAP proxy and WAP client is described. The channel is deployed in the NS-2 simulator using emulation functions. Modifications are made to let emulations working in mobile networking extensions. The channel consists of two nodes, one for mobile client, and one for base station. The simulator emulates the packet level behaviors of IS-95 and CDMA2000 1XRTT wireless channels. The maximum user data transmission rate for IS-95 is 9.6 Kbps, and 153.6 Kbps for CDMA2000 1XRTT. The transmitted packets are captured by tap agents and injected into the simulator. A RLC sublayer is introduced into Link Layer (LL). The packets are fragmented and defragmented at RLC according to emulated wireless network parameters. The channel errors are modeled by a uniform Frame Error Rate (FER), which is achieved by a random number generator. A Selective Repeat (SR) Automatic Repeat Request (ARQ) error recovery mechanism is employed at the RLC for the RLC fragments. A simplified MAC layer is used which sends RLC fragments every 20ms in IS-95 and 5ms in CDMA2000 1xRTT. ROHC is simulated in the wireless channel by giving the first RLC fragment a bigger size than the others. This assumes an average header compression ratio that is statistically stationary and fixed in a long run.

Chapter 6: Performance Evaluation of WAP Networks

In this chapter, the performance evaluation criteria of WAP networks are discussed, followed by a presentation of the assumptions and limitations of our emulation experience. The test results of both narrow-band IS-95 and wide-band CDMA2000 1xRTT are presented. The evaluations have been done for varying wireless channel FERs. Proxy configurations for WAP 1.x and WAP 2.0 with the proposed compression scheme are evaluated. Then an analysis and comparison between the proxy configuration and direct connect is discussed.

6.1 WAP Performance Evaluation Criteria

A general definition of performance can be described as the ability of the network to provide functions related to communications between users. The aim of the performance is to provide the quality of service (QoS) offered to users/customers. For a request and reply model adopted by WAP and web services, the end-to-end round trip delay is an important parameter of QoS.

The general WAP transaction works as follows: a request is invoked from the WAP client, which is sent to the proxy. The proxy processes the request (encoding or decoding if applicable) and forwards it to the WAP server. The WAP server processes the request and gives a reply with the appropriate result. The reply is received by the proxy and then processed. The encoded result is then sent back to the WAP client. The WAP client receives the result and processes it and shows the result in its browser.

The round trip delay, also known as Access Time (AT) can be used to evaluate the WAP performance for the transaction process. The access time includes the wireless transmission time (WTT, including LL retransmissions), Internet transmission time (ITT, including retransmissions if applicable), and the processing delay (PD). The PD can be divided into queuing delay (QD) at the WAP server and proxy and the processing time (PT) at the server, proxy and client.

$$PD = QD_{Server} + QD_{Proxy} + PT_{Server} + PT_{Proxy} + PT_{Client} \tag{6.1}$$

Since our test configurations utilize identical Internet clouds between the WAP proxy (gateway) and the WAP server, Internet delay is removed from our experimental model. The performance parameter considered is the average end-to-end wireless access time (WAT) or wireless round trip delay for a sample WML file [48], the WAT is given by the sum of WTT and PD.

$$WAT = WTT + PD \tag{6.2}$$

$$AT = WTT + ITT + PD = WAT + ITT \tag{6.3}$$

WML files are used in our test since we want to compare the WAP 2.0 configurations with WAP 1.x protocol stack. While WAP 2.0 continues its support for WAP 1.x based WML, the WAP 1.x stack does not recognize the new WAE definitions, such as XHTML. In our experiments, several WML files are tested and the results presented in this thesis are the average of those files.

The WAT is recorded in our timer program. For WAP 2.0, our introduced compression and decompression process works at the transport layer. The WAT is the time difference between

when the client makes a request and when it receives (and decompresses if necessary) a reply at the TCP socket. For WAP 1.x, the sessions are based on the most used class 2 WTP transactions, based on UDP as discussed in Chapter 2. The WAT is the time difference between the Invoke message and the Ack message, both from the client side.

6.2 Assumptions and Limitations

Due to the limitations of our test bed configuration and emulation channel, there are some assumptions and limitations in our experiments. They are summarized below:

- The emulated wireless channel is used solely by the WAP application during the experiments. There could be other applications sharing the channel in real life. A extra delay exists if the channel is shared with other traffic streams.
- Assume only one WAP session in progress at any given time, that is, no new WAP request is generated until the result from the former request is received. This results in no queuing delays at both the WAP server and the WAP proxy.
- A fixed link layer FER is assumed on both the uplink and downlink, whereas the traffic conditions in the CDMA channel may cause fluctuation in noise level and FER. The uplink and downlink may have different FER characteristics.
- The Internet delay is ignored. Had Internet delay been included in the evaluations, the same ITT would be added to both scenarios. It has no effects on the comparison of WAT, as both scenarios recover Internet errors over the Internet domain independently of the wireless domain. The overall access time may vary due to Internet variations.

- ROHC compression and decompression is simulated by a fixed compression ratio with zero processing time. In real conditions, the ROHC compression ratio is higher under low error rate and lower in poor channel conditions, so the processing time may vary accordingly.
- The content compression and decompression are implemented on Pentium PCs. The processing time is less in real gateways, and higher in less-powerful handsets.
- The WTLS layer in WAP 1.x and TLS layer in WAP 2.0 are not used during the testing. This would produce some extra delay.
- Not all optimizations suggested by the WP-TCP are implemented due to the emulation test-bed environment constrains. Handoff process and delay is not considered.

6.3 Performance Results

The experiments can be divided into two categories. First, the performance of WAP 1.x and WAP 2.0 is tested as a benchmark to assess the introduced compression scheme. Results show that the compression gives huge gains in WAT. In addition, direct connection and proxy configuration are compared to evaluate the performance enhancement by the proxy technology.

6.3.1 WAP Enhancement with Compression Scheme

The performance is evaluated by comparing the WAT of different compress options in various wireless FER conditions for both narrow-band IS-95 and wide-band CDMA2000 1xRTT channels. WAP 1.x is also tested under the same conditions as a comparison.

Eight typical WML pages are used for the testing, and the size of the pages ranges from 1k to 4k bytes. In fact most of the WML pages are around 2k bytes due to the limited screen size. The results used in the thesis are the average of these pages.

To assess the improvement of the proposed compression scheme, the content compression and ROHC are applied separately. This gives a clear view of contributions from each compression component. The content compression has 3 options: no compression, reply compression and request & reply compression. When no compression is used, the proxy becomes a standard WAP 2.0 proxy. ROHC can be applied together with content compression. Thus five different compression combinations are tested for each FER setting.

The processing behaviors of these settings are summarized in Table 6.1 below.

Compression Setting	Decompressor		Compression Proxy		ROHC (Y/N)
	Request	Reply	Request	Reply	
No comp	Forward	Forward	Forward	Forward	No
Reply comp	Forward	Decompress	Forward	Compress	No
Reply comp w/ ROHC	Forward	Decompress	Forward	Compress	Yes
Request & reply comp	Compress	Decompress	Decompress	Compress	No
Request & reply comp w/ ROHC	Compress	Decompress	Decompress	Compress	Yes

Table 6.1: Compression Scheme Processing Options

First, the transmission time with a LAN connection is tested to estimate the processing delay (PD) (Table 6.2). Since ROHC is simulated in wireless channel, it has no delay effect on the PD. The

PD in WAP 1.x comes from the protocol conversion, and data encoding and decoding at the gateway and client. Results show that in good conditions the WAP 2.0 using HTTP/TCP stack is more efficient than the WAP 1.x WSP/WTP/UDP stack. The introduced compression scheme gives a several milliseconds processing delay.

WAP 1.x	WAP 2.0 (HTTP, TCP/IP)		
	No compress	Reply compress	Request & Reply compress
180.05ms	4.02ms	6.69ms	11.51ms

Table 6.2: Processing Delays

Another important aspect of the test results shows that: since a proxy is used in our configuration, a long live connection between the WAP client and the WAP proxy can be formed, so no handshaking connection phase is included in the test results within section 6.3.1. The effect of the reconnection will be discussed later in section 6.3.2.

6.3.1.1 IS-95 Low-speed Bandwidth Performance Comparison

In an emulated IS-95 channel with a maximum bandwidth of 9.6 Kbps, Figure 6.1 shows that the WAT of WAP 1.x is 20-29% of the WAT of WAP 2.0 employing HTTP/TCP without data compression. That is, the WAT using the HTTP/TCP protocol stack is 3 to 5 times higher than the WAP 1.x protocol stack. This shows HTTP/TCP is not suitable for low-bandwidth high latency wireless channels. The different comes mainly from the reduced size of WSP encoding. Results clearly show the advantage of WAP 1.x over low-bandwidth networks.

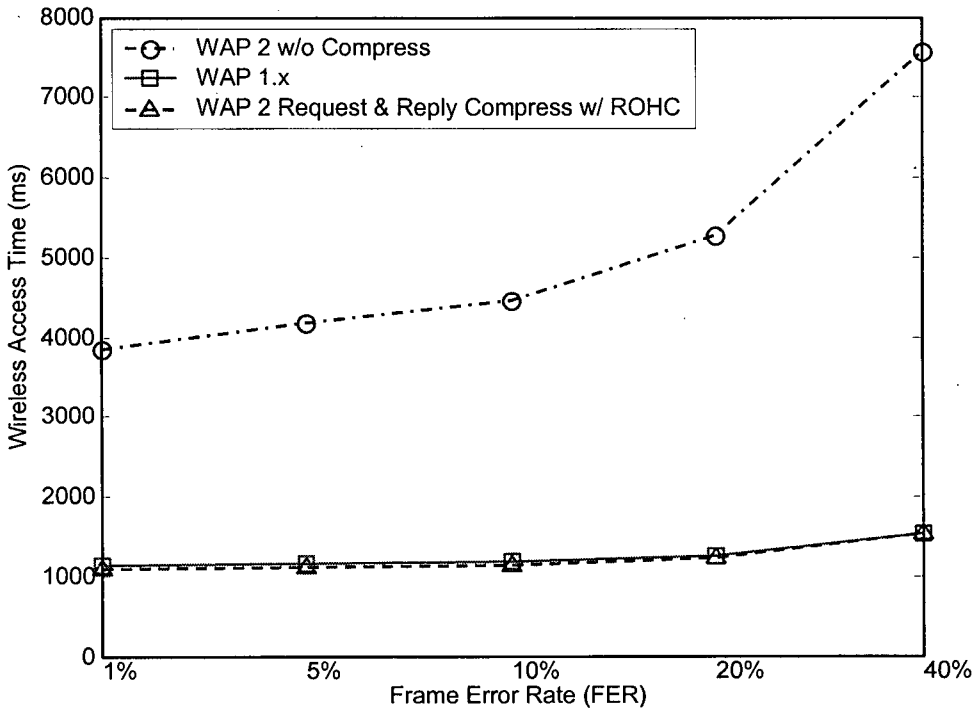


Figure 6.1: WAP Performance in IS-95

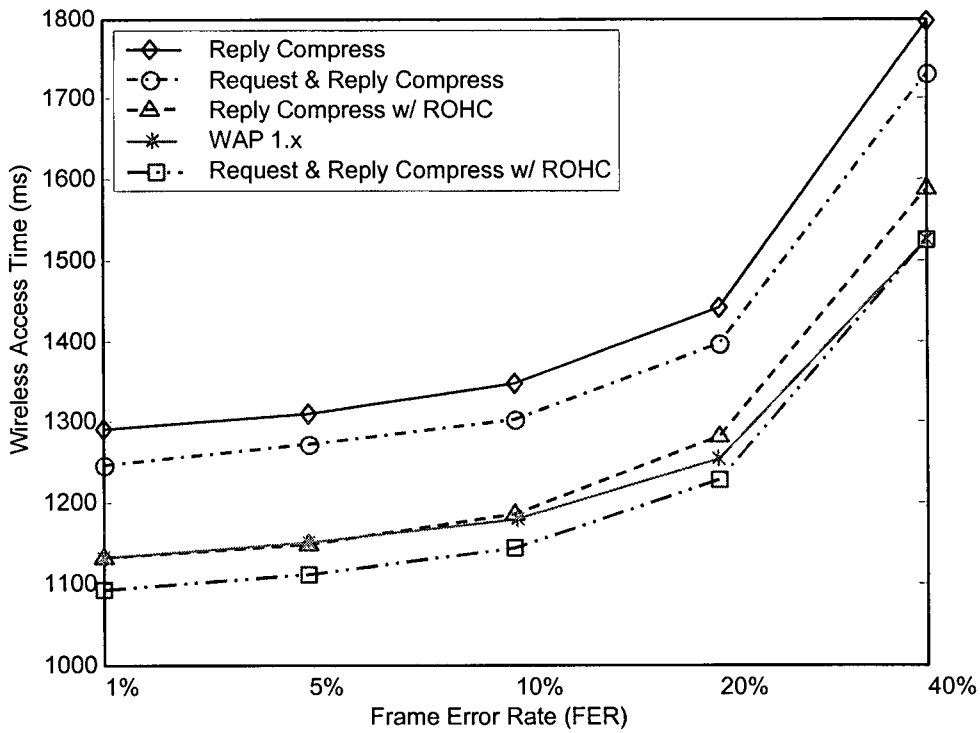


Figure 6.2: Performance of WAP 2.0 with Compression in IS-95

The proposed compression scheme is used in order to reduce the transmitted information over wireless channel. Figure 6.2 shows that data compression dramatically improves the performance of WAP 2.0. With only reply compression, the WAT reduces 66.5% at 1% FER and 76.2% at 40% FER. By combining request and reply compression, there is another 3% improvement over reply compression, which leads to an overall enhancement of 67.6% at 1% FER to 77.1% at 40% FER in WAT. If compared with WAP 1.x stack, WAP 1.x still performs better than content compression only. This shows WAP 1.x encoding is more efficient than the proposed content compression method in compression ratio.

To give further compression, ROHC is included in the compression. By applying ROHC, another 12% gain is achieved over content compression, yielding performance comparable to WAP 1.x. The overall enhancements for request and reply compression with ROHC, over no compression, are 71.6% at 1% FER and 79.8% at 40% FER, which are better than WAP 1.x, except in very bad channel conditions (FER > 40%). This is because the amount of data transmitted in WAP 1.x is smaller than even the content-compressed packets with ROHC. When FER is very high, the saved processing delay cannot compensate for the extra data transmission time.

The results show that WAP 1.x is optimized for low-bandwidth, high latency, high error rate wireless networks. With the advanced data compression scheme, the WAT performance of WAP 2.0 stack improves dramatically, and can match that of WAP 1.x.

6.3.1.2 CDMA 1xRTT High-speed Bandwidth Performance Comparison

The same set of experiments is tested in the CDMA 1xRTT channel. Results show that in an emulated CDMA2000 1xRTT channel with a maximum bandwidth of 153 Kbps, WAP 2.0 performs better than WAP 1.x, even if no compression is applied, with 32.5% and 9.4% less WAT at 1% and 40% FER respectively, as shown in Figure 6.3. This is because in a high-speed network, the large protocol conversion delay in the WAP 1.x gateway and client become significant. The lower processing time of HTTP/TCP makes it more appropriate for high-speed networks. Since the transmitted packet in WAP 2.0 is larger than in WAP 1.x stack, when FER is high, packet retransmissions degrade WAP 2.0 performance more severely than WAP 1.x.

The data compression scheme, greatly enhances the WAP 2.0 performance in WAT. Because WML (or XHTML) files are text-based, content compression yields high compress ratios. The compressed packets need much fewer LL fragments to transfer. Figure 6.4 indicates that content compression reduces the transmission delay by 44-47%. When ROHC is employed, another 3% enhancement can be achieved over content compression, which results in an overall improvement of 46%-49% over the no compression scenario.

Results also show that reply compression works better than the combined request & reply compression in CDMA 1xRTT emulation channel. This is because the request is quite small and the compression does not give much gain, and the transmission time reduction gained from the reduced size is smaller than the processing delay introduced by the request compression.

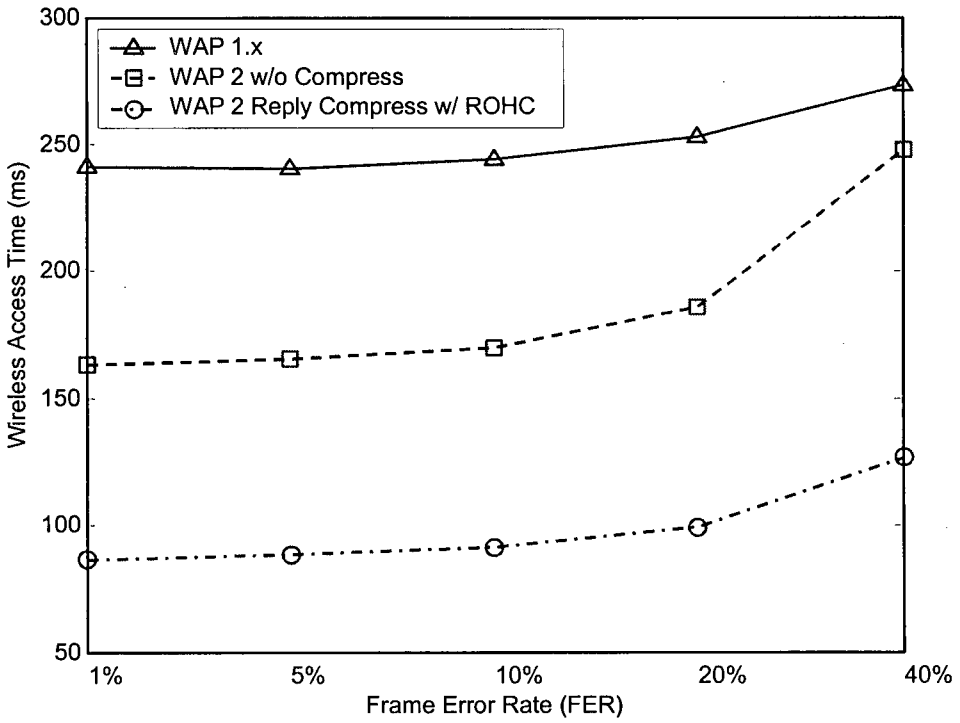


Figure 6.3: WAP Performance in CDMA2000 1xRTT

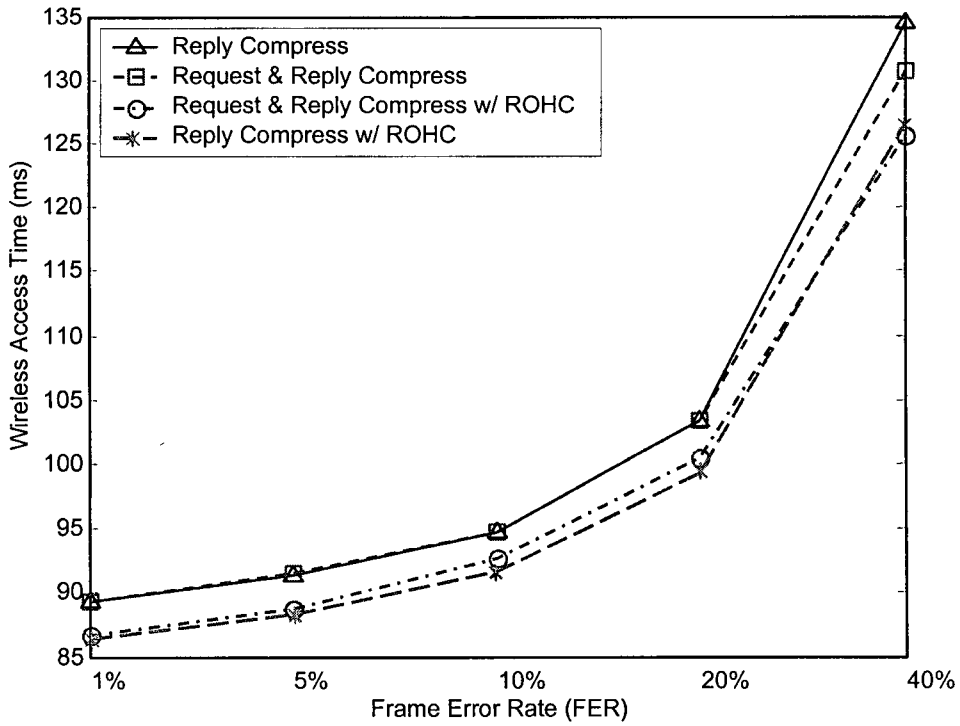


Figure 6.4: Performance of WAP 2.0 with Compression in CDMA2000 1xRTT

The experiment results show that WAP 2.0 is more suitable for high-speed wireless networks, and the compression scheme can yield over 45% improvement in WAT. WAP 1.x stack has a much higher WAT due to long processing delay, which makes it outdated for high-speed wireless networks.

6.3.2 WAP 2.0: Proxy vs. Direct Connection

In WAP 2.0, a direct connection is possible using HTTP/1.1 with end-to-end TCP connectivity. However, using a proxy, which leads to a split-TCP approach, can optimize and enhance the connection between the wireless domain and the Internet domain. Figure 6.5 depicts a WAP device directly accessing a Web Server via the Internet. The wireless IP router is a standard part of an IP network that is used to transfer IP packets from one link layer (e.g., the wireless link) to another (e.g., the wired link). This configuration can apply to cases where bearer-level security (such as IPSec) is utilized.

In case of a direct connection, the optimizations on WP-TCP and WP-HTTP over wireless links may not be available. The traditional TCP's congestion control mechanism adversely affects the

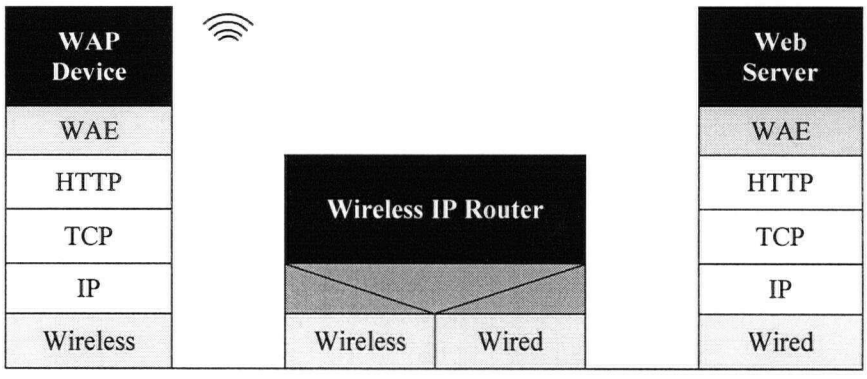


Figure 6.5: WAP 2.0 Direct Connection

TCP performance in a wireless medium. Aside from TCP mechanism problems, as mentioned at WP-TCP in section 2.2.3, two other major factors increase the wireless access time.

First, the split-TCP approach using a proxy shields problems associated with wireless links from the wireline Internet and vice versa. The direct connection causes error propagation between the Internet domain and the wireless domain. This can be proven by a simple calculation.

Given the error/drop rate and transmission time in the Internet and wireless domain as $\varepsilon 1$, $t1$ and $\varepsilon 2$, $t2$, respectively. In a direct connection, the access time (AT) is process delay (PD) plus direct transmission time (t_{direct}):

$$AT_{direct} = PD + t_{direct} = PD + \frac{t_1 + t_2}{(1-\varepsilon_1)(1-\varepsilon_2)} \tag{6.4}$$

If a proxy is present, AT is calculated by (6.5).

$$AT_{proxy} = PD + ITT + WTT = PD + \frac{t_1}{1-\varepsilon_1} + \frac{t_2}{1-\varepsilon_2} = PD + \frac{t_1 + t_2 - (\varepsilon_1 t_2 + \varepsilon_2 t_1)}{(1-\varepsilon_1)(1-\varepsilon_2)} \tag{6.5}$$

It is obvious that $AT_{proxy} < AT_{direct}$. The proxy facilitates independent error recovery over the wireless and Internet domains so that error-free data is always passed from one domain to the next. Thus no retransmission needs to pass through both domains.

Second, TCP is connection oriented. If there is no proxy, for a different WAP server there must be a handshaking for the TCP connection to be established. With a proxy, the mobile client can maintain a long-lived socket with it, thus eliminating extra connection and termination delays.

The wireless session delay (WSD) is used to represent these delays in our experiments. The WSD is the time delay due to TCP connection establishment and termination for different WAP servers in the wireless network.

Results in Figure 6.6 show that, over the low-bandwidth IS-95 channel, the WSD is quite high if a direct connection is used, 234 ms at 1% FER and 483 ms at 40% FER. If content compression is employed, the WSD is reduced 28% at 1% FER and 52% at 40% FER. ROHC in the wireless domain can give a 40% gain over the content compression scheme due to the reduced header size in the handshaking packets.

In CDMA2000 1xRTT channels, WSD is around 70ms at 1% FER and 95ms at 40% FER (Figure 6.7). The WSD is almost the same with and without data compression. This is because the handshaking packets are quite small and they can be transmitted in one physical layer frame in all cases.

Note that the WSDs presented above are obtained over an error-free Internet in our test environment. With the Internet clouds, the WSD will become even higher due to possible retransmissions of handshaking packets caused by the Internet error or drop losses. These results clearly illustrate the performance enhancement by the long-live connection with the proxy, especially when the clients frequently switch applications hosted on different WAP servers.

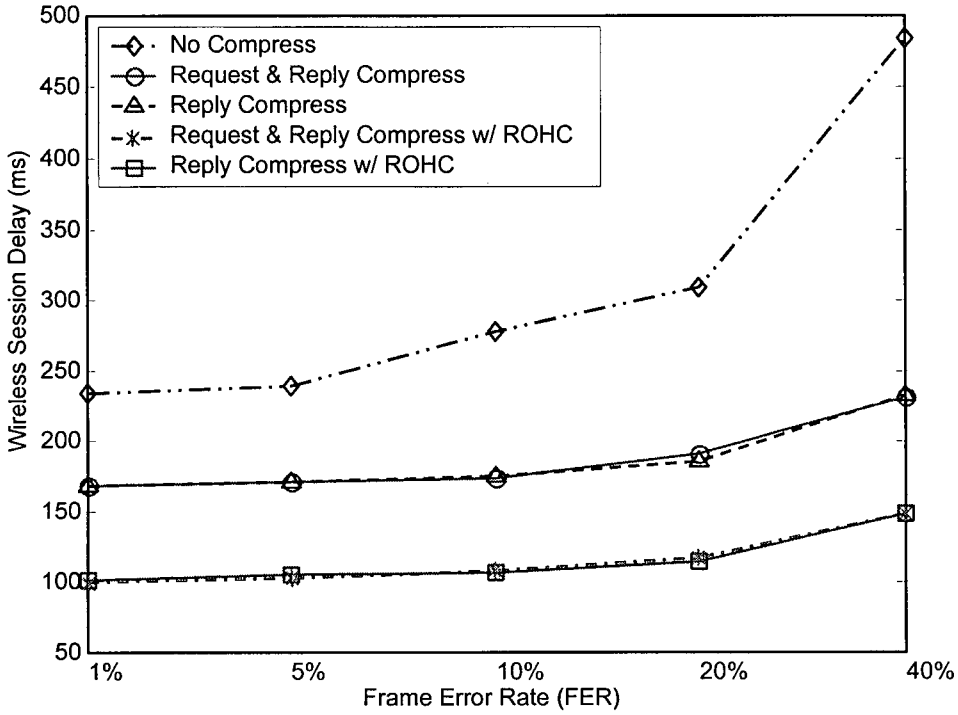


Figure 6.6: Direct Connection Wireless Session Delay in IS-95

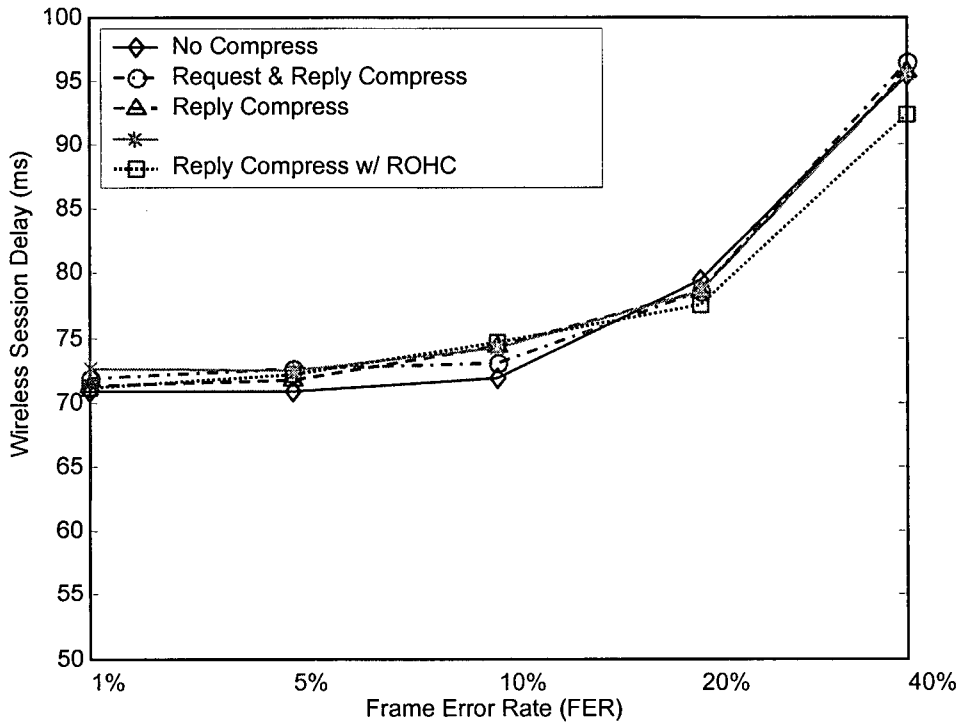


Figure 6.7: Direct Connection Wireless Session Delay in CDMA2000 1xRTT

6.4 Discussions and Optimization

In our experiments, the compression scheme is only applied to WAP 2.0, although it is possible to apply compression on UDP-based WAP 1.x WML bytecodes [49]. It is foreseeable that with the compression in WAP 1.x bytecode, the WAP 1.x protocol stack will only perform better only in low-bandwidth wireless channels. In the wide-band systems the process delay is the major source of wireless access delay in WAP 1.x, compression will not give much gain, since the bytecode is already compressed by WSP encoding and is quite small in size.

In our compression proxy, only the deflate algorithm is deployed, other lossless compression algorithms can be evaluated to reveal the best compression results for different types of content. More testing can be done in consideration of Internet conditions and queuing at both server and proxy. This can be modeled by several M/D/1 systems, and some theoretical results can be derived and included in our results. More optimizations on WP-HTTP WP-TCP can be deployed in the test bed. More modifications for new services such as MPEG-4 video streaming over wireless networks need to be developed and tested.

Chapter 7: Summary and Conclusions

In this thesis, a wireless emulation channel is implemented to simulate the IS-95 and CDMA2000 1xRTT wireless channels and to evaluate performances of different WAP stack configurations in terms of Wireless Access Time (WAT). A novel proxy architecture with compression scheme is introduced into the WAP 2.0 protocol network, which employs advanced data compression and significantly improves the access time performance of WAP 2.0. It also ensures strict end-to-end secured applications with TLS tunneling, thus overcoming WAP 1.x gateway security concerns.

The advanced data compression scheme consists of transport layer content compression and robust header compression (ROHC). Most of the WAT improvement is due to reply content compression, while ROHC can offer further improvement. Request compression is useful for low-bandwidth networks such as IS-95, but gives no benefit in high-speed wireless networks such as CDMA2000 1xRTT, due to extra processing time.

Although WAP 1.x protocol stacks are optimized for low-speed, high latency wireless networks, in IS-95 networks, the proposed request and reply compression with ROHC can reduce the WAP 2.0 access time to the same level as WAP 1.x. In CDMA2000 1xRTT networks, even the uncompressed WAP 2.0 outperforms WAP 1.x, and the introduced advanced data compression scheme can achieve over 46% less access time. Emulation results show the data compression scheme significantly improves WAP 2.0 performance in all cases. Our results enable appropriate configuration of the WAP 2.0 protocol stack for various bearer services.

Although a proxy is optional in WAP 2.0 network configurations, it can optimize the communication process by isolating the wireless from the wired domain, which gives a performance benefit as well as feature service enhancement. The proxy not only prevents the error propagation between wired and wireless domains, but also eliminates the wireless session delays (WSD) due to a TCP connection establishment by long-live connections between the proxy and wireless terminals. In WAP 1.x networks, the proxy, also known as the gateway, must be present to handle the protocol conversion and data encoding functions.

With the deployment of IP-enabled high-speed 3G networks, WAP 2.0 will facilitate further convergence with Internet technologies. However, WAP 1.x will still exist for narrow-band bearers and backward compatibility.

Bibliography

[1] WAP Forum, WAP 2.0 Technical Specifications http://www.wapforum.org/what/technical.htm.

[2] WAP Forum, "WAP architecture specification", version 12-July-2001, July 2001.

[3] WAP Forum, "WAP 2.0 technical white paper", Jan. 2002.

[4] K.K. Tan, C.Y. Soh, "Internet home control system using bluetooth over WAP" Engineering Science and Education Journal, vol. 11, Issue 4, pp. 126-132, Aug. 2002.

[5] N.M. Sgouros, S. Gerogiannakis, "Integrating WAP-based wireless devices in robot teleoperation environments", in Proc. IEEE ICRA '02, Washington D.C., vol. 2, pp. 1191- 1196, May 2002.

[6] P. d'Angelo, P. Corke, "Using a WAP phone as robot interface", in Proc. IEEE ICRA '02, Washington D.C., vol. 2, pp. 1173-1178, May 2002.

[7] M. Hagleitner, T.A. Mueck, "WAP-G: a case study in mobile entertainment", in Proc. of HICSS, Wailea Maui, Hawaii, pp. 1115-1124, Jan. 2001.

[8] T. Lilja; "Mobile energy supervision", Telecommunications Energy Conference, in Proc. INTELEC 2000, Phoenix, Arizona, pp. 707-712, Sept. 2000.

[9] J. Parkka, M. Van Gils, T. Tuomisto, R. Lappalainen, I. Korhonen; "A wireless wellness monitor for personal weight management", Information Technology Applications in Biomedicine, in Proc. IEEE EMBS 2000, Hangzhou, China, pp. 83-88; Sept. 2000.

[10] K. Hung, Y.T. Zhang; "On the feasibility of the usage of WAP devices in telemedicine", Information Technology Applications in Biomedicine, in Proc. IEEE EMBS 2000, Hangzhou, China, pp. 28-31, Sept. 2000.

[11] P. Stuckmann, C. Hoymann., "Performance evaluation of WAP-based applications over GPRS", in Proc. IEEE/ICC 2002, New York, pp. 3356-3360, May 2002.

[12] Andreadis, G. Benelli, G. Giambene, B. Marzucchi, "Performance analysis of the WAP protocol over GSM-SMS", in Proc. IEEE/ICC 2001, St.-Petersburg, Russia, vol. 2, pp. 467-471, June 2001.

[13] Sungwon Lee; Nah-Oak Song, "Experimental WAP traffic modeling on CDMA based mobile wireless network", in Proc. IEEE/VTC 2001, Rhodes Island, Greece, vol. 4, pp. 2206–2210, May 2001.

[14] S. Sheroan and V.C.M. Leung, "Evaluation of WAP network configuration supporting enhanced security" in Proc. IEEE/ICCE'02, Los Angeles, CA, pp. 78 –79, June 2002.

[15] P. Ashley, H. Hinton, M. Vandenwauver, "Wired versus wireless security: the internet, WAP and imode for e-commerce", in Proc. ACSAC 2001, New Orleans, LA, pp. 296 – 306, Dec. 2001.

[16] E.-K. Kwon; Y.-G. Cho and K.-J. Chae, "Integrated transport layer security: end-to-end security model between WTLS and TLS", in Proc. IEEE/ICOIN-15, Beppu City, Japan, pp. 65 –71, Jan. 2001.

[17] Liang Jin; Shi Ren; Liang Feng; Gao Zheng Hua; "Research on WAP clients supports set payment protocol"; IEEE Wireless Commun., vol. 9, Issue 1 , pp. 90-95, Feb. 2002.

[18] M. Soriano, D. Ponce, "A security and usability proposal for mobile electronic commerce"; IEEE Commun. Magazine , vol. 40 Issue: 8 , pp. 62 –67; Aug. 2002.

[19] WAP Forum, "WAP wireless session protocol", Approved Version 5-July-2001.

[20] WAP Forum, "Wireless transaction protocol", version 10-July-2001.

[21] WAP Forum, "Wireless transport layer security", Version 06-Apr-2001.

[22] WAP Forum, "Wireless datagram protocol", Version 14-Jun-2001.

[23] WAP Forum, "Wireless profiled HTTP", version 29-March-2001.

[24] WAP Forum, "WAP TLS profile and tunneling", version 11-April-2001.

[25] WAP Forum, "Wireless profiled TCP", version 31-March-2001.

[26] http://www.isi.edu/nsnam/ns

[27] http://forum.nokia.com/wapforum/main/toolkit

[28] http://www.kannel.org/

[29] http://httpd.apache.org/

[30] http://www.apachesoftware.com/

[31] P. Deutsch, "DEFLATE compressed data format specification version 1.3", RFC1951, May 1996

[32] P. Deutsch, "ZLIB compressed data format specification version 3.3", RFC 1950, May 1996

[33] http://www.gzip.org/zlib/

[34] Bormann etc., "Robust header compression (ROHC)", RFC3095, July 2001.

[35] M. Degermark Ed., "Requirements for robust IP/UDP/RTP header compression"; RFC 3096, June 2001.

[36] M. Degermark, H. Hannu,, E. Jonsson, K. Svanbro, "Evaluation of CRTP performance over cellular radio networks", IEEE Pers. Comms, vol.7, no.4, Aug. 2000.

[37] TIA/EIA Interim Standard-95, "Mobile station – base station compatibility standard for dual-mode wideband spread spectrum cellular system", July 1993.

[38] http://www.3gpp2.org/

[39] 3GPP2 C.S0001-C, Version 1.0, "Introduction to cdma2000 standards for spread spectrum systems, Release C", May 28, 2002

[40] 3GPP2 C.S0001-C, Version 1.0, "Physical layer standard for cdma2000 spread spectrum systems, Release C", May 28, 2002

[41] John J. Lemmon, "NTIA Report 02-394, Wireless link statistical bit error model", June 2002

[42] J. Pérez-Romero, L.G. Alonso, R. Agustí, "Average block error probability in the reverse link of a packet DS/CDMA system under Rayleigh fading channel conditions", IEEE Commun. Letters, vol. 4, No. 4, pp. 116-118, April 2000.

[43] C.C. Tan, N.C. Beaulieu, "On first-order Markov modeling for the Rayleigh fading channel", IEEE Trans. on Commun., vol. 48, No. 12, pp. 2032-2040, Dec. 2000.

[44] M.R. Hueda, "On first-order Markov modeling for block errors on fading channels", IEEE VTC Spring 2002. Birmingham, AL, vol. 3, pp. 1336-1339, May 2002.

[45] M.R. Hueda, "On the Markovian approximation for block-errors in DS-CDMA transmissions over slow fading channels with multicarrier transmit diversity", IEEE ICC 2002, N.Y., USA, vol. 2, pp. 737-741, May 2002.

[46] M.R. Hueda, "On the equivalence of channel-state and block-error Markov models for the analysis of transmission over slow fading channels", IEEE VTC 2001, Rhodes, Greece, vol. 2, pp. 1043-1047, May 2001.

[47] K. Fall and K. Varadhan Ed., The VINT project, "The ns manual", http://www.isi.edu/nsnam/ns/ns-documentation.html

[48] Eged, T. Dezso, F. Egedi, "Server side round-trip delay measurements in WAP environments", in Proc. IEEE/IMTC 2001, Budapest, Hungary, vol. 1, pp. 525 –529, May 2001.

[49] Ojanen, J. Veijalainen, "Compressibility of WML and WMLScript byte code: initial results", in Proc. IEEE/RIDE 2000, San Diego, pp. 55 –62, Feb. 2000.

[50] V. Jacobson, R. Braden, D.Borman, " TCP extensions for high performance", RFC1323, May 1992.

[51] M. Allman, V. Paxson, W. Stevens, "TCP congestion control", RFC2581, April 1999.

[52] M. Allman, S. Floyd, C. Patridge, "Increasing TCP's initial window", RFC2414, October 1996

[53] M. Mathis, J. Madhavi, S. Floyd, A Romanow, "TCP Selective Acknowledgement Options", RFC2018, October 1996

[54] J. Mogul, S. Deering, "Path MTU Discovery", RFC1191, November 1990.

[55] K. Ramakrishnan, S. Floyd, "A Proposal to add Explicit Congestion Notification (ECN) to IP", RFC2481 January 1999.

[56] J. Salim, U. Ahmed, "Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks", RFC2884, July 2000.

[57] V. Jacobson, "Compressing TCP/IP headers for low-speed serial links", RFC1144, February 1990.

[58] S. Casner, V.Jacobson, "Compressing IP/UDP/RTP headers for low-speed serial links", RFC2508, February 1999.

[59] A. Feldspar, "An explanation of the deflate algorithm", http://www.gzip.org/zlib/feldspar.html, August 23, 1997.

Appendix: List of Abbreviations and Acronyms

2G	2nd Generation cellular networks
3G	3rd Generation cellular networks
3GPP2	Third Generation Partnership Project 2
ARQ	Automatic Repeat Request
BDP	Bandwidth Delay Product
BER	Bit Error Rate
BLER	Block Error Rate
BPF	Berkeley Packet Filter
CDMA	Code Division Multiple Access
CGI	Common Gateway Interface
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CSS	Cascading Style Sheets
DNS	Domain Name System
DS/CDMA	Direct Sequence CDMA
ECN	Explicit Congestion Notification
EDGE	Enhanced Data Rates for Global Evolution
EFI	External Functionality Interface
FDMA	Frequency Division Multiple Access
FEC	Forward Error Correction
FER	Frame Error Rate
GIF	Graphics Interchange Format
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HSCSD	High-Speed Circuit-Switched Data

HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IETF	Internet Engineering Task Force
IMT-2000	International Mobile Telecommunications -2000
IP	Internet Protocol
IS-95	Interim Standard 95
ISI	Inter-Symbol Interference
LAC	Link Access Control
LAN	Local Area Network
LL	Link Layer
LZ77	Lempel-Ziv 77 Compression Algorithm
LZW	Lempel-Ziv-Welch Compression Algorithm
MAC	Medium Access Control
MTU	Maximum Transmission Unit
NIC	Network Interface Card
NS-2	Network Simulator version 2
PILC	Performance Implications of Link Characteristics
PKI	Public-Key Infrastructure
QoS	Quality of Service
RFC	Request For Comments
RLC	Radio Link Control
RLP	Radio Link Protocol
ROHC	Robust Header Compression
RTP	Real-Time Transport Protocol
RTT	Radio Transmission Technologies
SACK	Selective Acknowledgement
SCH	Supplemental Channel

SNR	Signal to Noise Ratio
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
TLS	Transport Layer Security
TIA	Telecommunication Industry Association
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WAE	Wireless Application Environment
WAP	Wireless Application Protocol
WAT	Wireless Access Time
WBMP	Wireless Bitmap
WCDMA	Wideband Code Division Multiple Access
WDP	Wireless Datagram Protocol
WML	Wireless Markup Language
WP-HTTP	Wireless Profiled HTTP
WP-TCP	Wireless Profiled TCP
WSD	Wireless Session Delay
WSP	Wireless Session Protocol
WTA	Wireless Telephony Application
WTLS	Wireless Transport Layer Security
WTP	Wireless Transaction Protocol
WWW	World Wide Web
XHTML	Extensible Hypertext Markup Language
XHTMLMP	XHTML Mobile Profile
XML	Extensible Markup Language