Delay-Centric Handover in SCTP

Andrew Kelly, B.Eng

Master of Engineering

School of Electronic Engineering
Dublin City University

Supervised by Dr. John Murphy

September 2004

Acknowledgements

I would like to thank my supervisor Dr. John Murphy for his guidance and commitment to this project. Thanks are also due to Dr. Philip Perry for supporting this work, providing invaluable feedback, and for allowing me to bounce ideas off of, especially on the bus and not look like a raving loon. Thanks must be given to my colleagues for much needed lunchtime discussion/distraction, which kept me sane.

Finally, thanks must go to my mother for her patience and warm meals.

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Master of Engineering in Electronic Engineering is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed: _____ (Candidate) ID No.: 96624141

Andrew Kelly   Date: 10 01 2005

Abstract

The introduction of the Stream Control Transmission Protocol (SCTP) has opened the possibility of a mobile aware transport protocol. The multihoming feature of SCTP negates the need for a solution such as Mobile IP and, as SCTP is a transport layer protocol, it adds no complexity to the network. Utilizing the handover procedure of SCTP, the large bandwidth of WLAN can be exploited whilst in the coverage of a hotspot, and still retain the 3G connection for when the user roams out of the hotspot's range. All this functionality is provided at the transport layer and is transparent to the end user, something that is still important in non-mobile-aware legacy applications.

However, there is one drawback to this scenario - the current handover scheme implemented in SCTP is failure-centric in nature. Handover is only performed in the presence of primary destination address failure. This dissertation proposes a new scheme for performing handover using SCTP. The handover scheme being proposed employs an aggressive polling of all destination addresses within an individual SCTP association in order to determine the round trip delay to each of these addresses. It then performs handover based on these measured path delays. This delay-centric approach does not incur the penalty associated with the current failover-based scheme, namely a number of timeouts before handover is performed. In some cases the proposed scheme can actually preempt the path failure, and perform handover before it occurs. The proposed scheme has been evaluated through simulation, emulation, and within the context of a wireless environment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent times there has been a paradigm shift in the way Internet services are accessed. The image of the static desktop user accessing the internet over fixed, wireline networks such as Ethernet, is being superseded by a mobile, "Internet Anywhere" model. This shift in paradigms can be attributed to the increasing proliferation of portable computing devices. What began in the 1990s with the introduction of digital cellular telephones has continued in this decade with laptop computers and Personal Digital Assistants (PDAs).

As the desire to access services, regardless of location, has grown, service providers have begun to realize the vast potential of this new wireless method of communication. The concepts of composite radio and wireless reconfigurability have allowed operators to deploy their services within a wireless environment most suited to the requirements of the service. Composite radio allows different radio access technologies, e.g. Global System for Mobile Communications (GSM), General Packet Radio Service (GPRS), Universal Mobile Telephony System (UMTS), Wireless Local Area Network (WLAN), to coexist within a heterogeneous wireless access infrastructure. The concept of reconfigurability is what allows terminals, user devices, to adapt to differing wireless networks based on the provider's service, chosen by the user. It is this reconfigurability, coupled

with the increasing computational ability of these portable devices that will drive the use of Internet-based services to new heights.

In order to maintain access to these services within this heterogeneous environment, a mobility scheme must be employed to facilitate user movement. Particularly the concept of vertical handover must be addressed. Vertical handover refers to the procedure of switching from one wireless network to a different wireless network whilst keeping the user's active connection alive.

The decision of where in the protocol stack this vertical handover should be performed is a matter that needs to be considered. To illustrate, consider the OSI model [1] as a representation of the protocol stack. This model is composed of seven distinct layers, which are collected together into application-oriented layers (application, presentation, and session) and network-dependent layers (network, link, and physical) with the transport layer situated in between.

Making the handover decision at the higher end of the stack, towards the application-oriented layers, will incur a certain latency before the handover is performed. The higher up the stack, the greater the latency incurred in performing handover. This may result in inefficient use of the available resources, e.g. remaining longer on a GPRS connection even though a higher bandwidth WLAN connection is available. Making the handover decision lower down the stack, towards the network-dependent layers, would appear to be a better solution. The latency incurred by the higher layers would be reduced resulting in a faster handover being performed. In the case of the GPRS-WLAN handover example, this would result in better utilization of the available bandwidth. However, making the handover decision at any of the network-dependent layers would require reconfiguration of many nodes along the transmission path from the server to the client. In a large scale environment such as the Internet, this is not economically, nor technically feasible.

A viable solution to these problems is to make the handover decision at the trans-

port layer. Employing this solution means that the handover latency incurred at the transport layer is less than that experienced at the application-oriented layers, and as the transport layer is an end-to-end layer it requires no modifications be made to intervening network equipment. However, the two most widely used transport layer protocols, Transmission Control Protocol (TCP) and User Datagram Protocol (UDP), do not support handover natively. A third transport protocol, Stream Control Transmission Protocol (SCTP), was standardized in 2000. SCTP was originally designed to transport telephony signalling over IP networks, and provides support for signalling network characteristics such as path redundancy. This support is via SCTP's multihoming feature. It has been proposed in [2, 3] that this multihoming feature of SCTP could be used to facilitate vertical handover.

1.1   Purpose of the Thesis

Whilst SCTP's multihoming feature provides functionality to facilitate handover, the actual process of performing a handover within SCTP is still designed with wired networks in mind. As such, the handover process defined in the SCTP standard requires a number of timeouts on the currently active transmission path before handover will occur. These timeouts can incur a significant delay in the handover process. This dissertation proposes a new scheme as the basis for handover. Currently, destination addresses in an SCTP association are polled periodically to determine their status. The handover scheme proposed by this dissertation performs more frequent polling of the destination addresses in order to gather Round Trip Time (RTT) measurements for each address. These RTT measurements form the basis for the handover decision.

3

1.2   Structure of the Thesis

Chapter 2 introduces SCTP. It outlines the motivation behind the development of SCTP and contains an overview of some of the concepts of this new transport layer protocol, such as the multihoming feature. Chapter 3 describes the process of handover with regards to cellular systems. It defines the concept of user mobility within IP networks, and describes the Mobile IP mechanism, currently used to achieve this mobility. It also outlines how the Session Initiation Protocol (SIP) could be modified to provide a mobility scheme to users within IP networks. Chapter 4 describes handover within the context of SCTP. It begins with a description of the current handover scheme implemented in SCTP. After detailing the deficiencies of this scheme, the proposed handover scheme is described. Chapter 5 evaluates the proposed handover scheme using the ns-2 simulator. Chapter 6 implements the proposed handover scheme within a wireless network and evaluates its performance. Chapter 7 concludes this dissertation.

Chapter 2

SCTP Overview

2.1 SCTP History

The motivation for the development of the SCTP began with the desire to transport telephony signalling messages over IP networks. Up to this point telephony signalling had been performed over its own separate signalling network, using a protocol called Signalling System Number 7 (SS7)[4] to ensure reliable delivery of signalling messages. It does this by ensuring that there are multiple paths, called links, available between switching entities in the network. This path redundancy is fundamental to switching networks.

Due to the reliability requirements imposed by the nature of telephony signalling, TCP was initially considered as the protocol to transport telephony messages over IP networks. TCP [5] is a transport layer protocol that provides a reliable connection between two endpoints over an IP network. Inherent to TCP's reliability is the strict order of transmission delivery policy that it employs. Under this delivery scheme, should a packet become lost in the network, all subsequent packets in the TCP flow would become blocked until the missing packet was successfully retransmitted. This is referred to as head-of-line blocking and could cause huge delays when transporting

signalling messages over unreliable networks such as IP. The effect of these delays could be the failure of certain types of service, and result in a loss of revenue for the service provider.

This head-of-line blocking was only one of the limitations of TCP that was discovered when applying the protocol to the task of telephony signalling transport over IP networks. Some of the other limitations included a susceptibility to certain Denial of Service (DoS) attacks, no message boundary delineation mechanism, and no support for multihomed hosts [6]. Each of these limitation is discussed briefly.

2.1.1 DoS Susceptibility

TCP's vulnerability to certain types of Distributed Denial of Service (DDoS) attacks, particularly SYN flooding, was one of the reasons it was ruled out of consideration when it came to telephony signalling. A SYN flooding attack involves a malicious host (attacker) attempting to establish many connections to a targeted host (victim). It begins each of these connections by transmitting a TCP SYN packet. The attacker can spoof its IP address making it appear that the SYNs are originating from many different machines. The victim machine, upon reception of each SYN packet , checks to see if the packet is for a port that is accepting connections. If this is so, the victim machine reserves some kernel space for the impending connection and returns a SYN-ACK message to the attacker. The attacker, however does not complete the connection. If the attacker can transmit a large amount of SYN packets in a short space of time, it will deplete the resources on the victim machine and result in the victim machine being unable to service legitimate connection requests. In extreme cases, the attack can completely knock the victim machine out of service.

Certain steps have been taken to help defend against these attacks (ingress filtering [7], ingress access lists, strict reverse path forwarding [8]). However, these measures were not in place when the design of SCTP began, and as such its designers considered

these attacks, and TCP's handling of them, to be of such importance that they did not wish for their signalling application to be directly affected by them.

2.1.2 No Message Delineation

TCP is byte-oriented in design. This means that all data passed down to the transport layer from the application layer is sent as a stream of bytes to the other end. No record markers are inserted by TCP to delineate where one message ends and another begins. This was thought to be an inconvenience to message-based telephony signalling. Applications would have to implement their own method of delineating messages within the byte stream. Practically this would involve marking the beginning and the end of each message, and an extensive use of the push facility to ensure transfer of a complete message within a feasible timeframe.

2.1.3 No Multihoming Support

As stated previously, signalling networks rely on the concept of path redundancy to ensure that signalling messages are transported reliably, with as minimal delay as possible. Since such endpoints in a signalling network would have to be reachable by more than one route, perhaps through several distinct physical networks, this would require the endpoint to be equipped with more than one network interface. Such endpoints are termed multihomed. TCP offers no native support for multihomed endpoints. For this functionality to be implemented it would have to be designed into the application. Due to the mechanisms that have to be handled for each path (congestion control, status, etc), it is a complex matter to design an application that could deal with these sufficiently.

Due to these limitations, TCP was eliminated from consideration for transporting telephony signalling messages. The designers next turned to the UDP. UDP [9] is also a transport layer protocol, but, unlike the connection-oriented TCP, is connectionless

7

by design. This means that it cannot provide a reliable connection between two hosts. UDP cannot guarantee what order messages it has transmitted arrive in, or that they arrive at all. It was this unreliability of the protocol that did not appeal to those wishing to implement a signalling system renowned for its highly available, reliable, data transfer capability.

Another limiting factor of UDP is that it has no explicit congestion control mechanisms. It cannot detect when a network has become congested and as a result of this will not throttle back its transmission rate. This can lead to two undesirable effects [6]:

1. If the network over which the UDP traffic is travelling is congested, the addition of more UDP traffic will not alleviate the congestion; it will, in fact, increase the congestion level present in the network. This will have adverse effects on the traffic of other applications transmitting over the same network.

2. As the congestion increases and queues build up at the intervening routers throughout the network, some packets will inevitably be discarded (due to finite buffer capacity). This makes UDP even more unreliable.

These limitations rule UDP out of providing a means to transport telephony signalling. But UDP is considered a lightweight protocol and as such many of its drawbacks can be overcome by correctly designing an application to sit on top of UDP. UDP is message based and has little overhead. The application that would sit on top of UDP would have to implement mechanisms to overcome UDP's deficiencies, namely

- Some form of congestion control,

- A retransmission scheme to detect and retransmit lost packets,

- A mechanism to correct out of order and duplicated packets.

While it appears as though UDP does provide some ground work upon which to build a signalling application, actually designing and implementing the mechanisms mentioned above is a non-trivial matter.

8

With TCP excluded due to the limitations mentioned previously, any implementation that was to meet the requirements of signalling message transport was designed to work with UDP as the underlying transport protocol. Three consecutive works began, each with the goal of providing reliable signalling over UDP. The culmination of these individual efforts, entitled Multi-network Datagram Transmission Protocol (MDTP), came about in 1997. Work continued on MDTP, producing a working implementation which, along with the general concepts of MDTP, was submitted to the IETF for consideration in 1998.

Around the time of MDTP's submission, the IETF were beginning an initiative to investigate the transport of telephony signalling over IP networks. The outcome of this initiative was the forming of the SIGTRAN working group within the Transport Area of the IETF. SIGTRAN's modular architecture, coupled with the requirements for telephony signalling, integrated well with the concepts of MDTP's design. The merging of the two groups resulted in a host of modifications being made to MDTP. Some of these modifications, or refinements, included:

- Multistream concept

  This was introduced as a defence against the head-of-line blocking that was one of the reasons for rejecting TCP. Central to the multistream concept is the idea that each stream is logically separate from the others. Messages arriving in one stream will not be blocked due to message loss within other streams. All data chunks contain a stream identifier, a stream sequence number, and a transmission sequence number. These terms are explained in Section 2.2.

- Congestion control enhancements & SACK improvements

  The congestion control schemes of MDTP were redesigned to more closely resemble those of TCP, as described in [10]. The Selective Acknowledgement (SACK) mechanism was implemented based on the work done on SACK TCP [11].

- Four way secure handshake

  The original setup procedure involved a three-way handshake, similar to TCP's. This was redesigned to incorporate a security mechanism, an encrypted state cookie, as a defence against DDoS attacks. Another enhancement made to the setup mechanism was to allow user data to be transferred on the third and fourth leg of the handshake. This would reduce the connection setup overhead to two round trip times.

- Message bundling improvements

  The working group introduced a chunk-based message bundling system to replace the previous delivery mechanism. This chunk-based system is covered in more detail in the Section 2.2.

- Path MTU discovery

  This was implemented by the working group as a fundamental feature of the protocol. This would allow MDTP to detect and adapt to differing network conditions.

In order to emphasize the changes made to MDTP the working group decided to rename the protocol to the Stream Control Transmission Protocol. This name change reflected more than just the enhancements made to MDTP's features. It emphasized the expansion of scope and functionality of the protocol. Due to the robustness of the design and the increased capability of the protocol the IETF Transport Area Directorate (TAD) along with the Internet Engineering Steering Group (IESG) made the recommendation that SCTP should run over IP directly, instead of over UDP as it was currently implemented, in December 1999. The IESG and TAD saw a lot of potential in SCTP outside the scope of transporting telephony signalling. Their recommendation of implementing SCTP over IP was an indication of the significance of SCTP as a general purpose transport protocol.

This recommendation was met initially with some trepidation by the working group. It was thought that with SCTP now being a general purpose transport protocol, like TCP and UDP, it would certainly have to be implemented in the operating system kernel code. Until this point SCTP had been implemented in user space. Moving it to the kernel could take a long time to roll-out, as many operating system vendors can take several years to implement a new protocol in their kernel code. Also, due to the fact that the SCTP code would now be implemented within the kernel code of the operating system, some members of the working group felt that they would no longer have strict control over retransmission timers. However, the IESG insisted that SCTP should be defined as a general purpose transport protocol, and suggested that the working group should look outside the scope of transporting telephony signalling to see the benefits SCTP offered to applications with similar requirements. The working group agreed and SCTP was accepted as Request for Comment (RFC) 2960 in October 2000.

2.2 SCTP Terminology

In order for reliable communications to commence in an IP network two nodes must establish a connection, or communications link, between them. Furthermore the nodes must be in a state where they are able to transmit data to, or willing to receive data from, the other node in the link. In SCTP terminology, the two nodes at either end of a communications link are called endpoints. The communications link between the endpoints is termed an association. An association between two multihomed endpoints can be seen in Fig. 2.1.

SCTP is defined as a transport layer protocol in [12]. Endpoints communicate with each other by means of transport addresses. A transport address is composed of an IP address and a SCTP port number. Endpoints use transport addresses to locate a particular application running on a specific machine in a network. The IP address
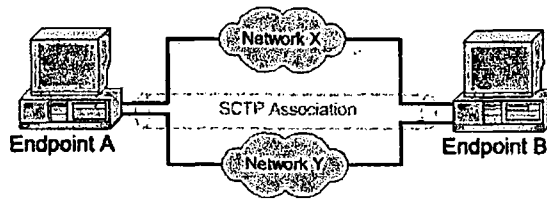
Figure 2.1: An SCTP Association

locates the machine (or more precisely the network interface on the machine) and the port number determines which application running on that machine the arriving data is intended for. This mechanism works similarly for TCP and UDP. SCTP's port space is allocated by the Internet Assigned Numbers Authority (IANA). Both TCP and UDP each have their own port space, also allocated to them by IANA.

2.2.1 Chunks

All communication between two endpoints across an association is conducted through the use of chunks. These are SCTP messages and are divided into two types: control chunks and user chunks. The latter are used to transmit data, passed down from the application, to the peer endpoint. Control chunks are used to establish, monitor, and terminate an association. Different control chunks are used for each of these tasks. Table 2.1 provides an overview of SCTP's control chunks and the functions they perform.

Chunk Type	Chunk Name	Description
0x00	DATA	Used to transmit user data
0x01	INIT	First chunk in SCTP's four way handshake
0x02	INIT-ACK	Response to INIT chunk. Contains State Cookie parameter
0x03	SACK	Acknowledges received user data
0x04	HEARTBEAT	Used to poll idle destination address periodically
0x05	HEARTBEAT-ACK	Sent in response to a HEARTBEAT chunk
0x06	ABORT	Used to terminate an association abruptly
0x07	SHUTDOWN	First chunk in the three-way graceful shutdown procedure
0x08	SHUTDOWN-ACK	Sent in response to a SHUTDOWN chunk
0x09	ERROR	Used to report operational errors to a peer endpoint
0x0a	COOKIE-ECHO	Third chunk in the four way handshake. Returns State Cookie
0x0b	COOKIE-ACK	Response to COOKIE-ECHO. Final chunk in the four way handshake
0x0c	ECNE	Explicit Congestion Notification (ECN) Echo chunk. Reserved
0x0d	CWR	Congestion Window Reduced chunk. Reserved for ECN
0x0e	SHUTDOWN-COMPLETE	Final chunk in three-way graceful shutdown procedure

Table 2.1: SCTP Chunk Types

The chunk types 0x0f - 0xff are reserved by the IETF for future use.

SCTP chunks comprise *type, flags, length* and *data* fields. This is shown in Fig. 2.2. As each chunk is to be completely self-descriptive, the *type* field states what type of chunk it is. A list of the currently defined chunks is given in Table 2.1. The *flags* field contains any special flags that a particular chunk type might need to set. This field will take on a different meaning with each chunk type. The *length* field indicates how long the chunk is, and is stored in bytes. It includes the type and flags fields, as well as itself,
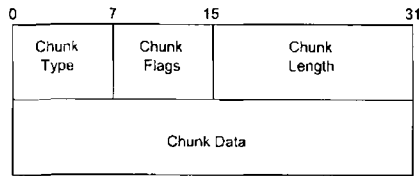
Figure 2.2: SCTP Chunk Structure

when calculating the chunk length, so that if a chunk contains no data, length will be set to 4. The chunk *data* field must end on a 32-bit word boundary. If the information contained in the chunk data field does not end on this boundary it is padded to the next word boundary. The length field does not include any padding that might have been added to the data part of the chunk.
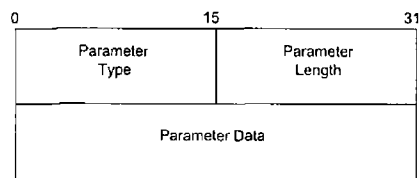
Figure 2.3: TLV Format

The data field is variable in length and contains parameters specific to the chunk type. These parameters are stored in Time-Length-Value (TLV) format. This format is shown in Fig. 2.3. The *type* field indicates what parameter this is. The *length* field contains the length of the parameter in bytes. This includes the type field and itself, so that a parameter that contains no data still has the length field set to 4. The *data* field is variable in length and contains the information carried in the parameter. It is padded to the next 32-bit word boundary if its length is not a multiple of four bytes. The padding is not included in the parameter length field. For more information on the TLV format see [6].

Chunks are transmitted across the network in SCTP packets. An SCTP packet contains the SCTP common header and one or more chunks to be transmitted. The SCTP common header, as shown in Fig. 2.4, is always first in the SCTP packet. It

14

0	15	31
Source Port Number	Destination Port Number	
Verification Tag		
Checksum		
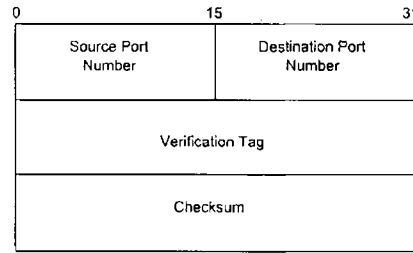
Figure 2.4: SCTP Common Header

contains information such as the source and destination port numbers, a verification tag, and a checksum. The port numbers, coupled with the source and destination addresses in the IP packet header, form the transport addresses described previously. The Verification Tag is to ensure that the packet belongs to the current association and not some previous association between the same two endpoints. It also protects against a blind attacker injecting data into the current association. The checksum ensures that the data within the header has not been changed whilst being transmitted across the network.

2.2.2 Multihoming

One of the features of SCTP that differentiates it from both TCP and UDP is its support of multihoming. If a host may be reached by more than one IP address, it is said to be multihomed, as shown in Fig. 2.1. It may be that the machine has two or more Network Interface Cards (NICs) with an IP address assigned to each interface card, or it has one network card with multiple IP addresses assigned to it. The former is termed simple multihoming in [13] and for the duration of this document is what is meant when referring to hosts that are multihomed. An example of the latter is when, in IPv6 networks, a global address, a site-local address, and a link-local address are assigned to a network interface card.

15

2.2.3 Multistreaming

As stated in Section 2.1, one of the reasons TCP was not used in the transport of signalling messages over IP networks was its strict order of transmission delivery policy. More succinctly it was the head-of-line blocking problem that this policy can incur that led to TCP being rejected. The designers of SCTP introduced the concept of individual message flows within an association as a solution to head-of-line blocking. Messages in one flow, or stream, are unaffected by the loss of a message within another stream. This means that messages within a single stream can be delivered to the application without having to be delayed until the lost message has been retransmitted [14]. Streams are identified by means of a Stream Identifier contained within each data chunk.

Each data chunk transmitted by the sender, regardless of which stream it was sent on, is assigned a unique Transmission Sequence Number (TSN). The sender and receiver use this TSN to determine if all data chunks have been received correctly, or, if there have been losses, which chunks need to be retransmitted. This is analogous to the sequence numbering scheme TCP uses to detect when packets have become lost or reordered. Data chunks also contain a Stream Sequence Number (SSN), which dictates their order within a stream. All messages within a single stream are delivered in the order they were transmitted in, i.e. in increasing SSN. This strict order of delivery within each stream can lead to head-of-line blocking should a message become lost or reordered en route from the sender to the receiver, however, messages within other streams will not be blocked because of this loss.

Currently all streams within an association are of equal priority, that is SCTP does not prioritize one stream over another. However, there have been efforts to adjust this and implement a prioritizing scheme in SCTP [15]. It is believed that by prioritizing a stream, the data contained within a high priority stream would not experience as significant a delay as data contained in a low priority stream during periods where the available bandwidth was low.

16

2.3 SCTP Association Setup

An association is established between two SCTP endpoints by means of a four way message exchange, as shown in Fig. 2.5. This is analogous to TCP's three way handshake mechanism, however SCTP's method incorporates a security measure, known as a Cookie. This security feature provides a defense against the DDoS attacks, such as SYN flooding, that TCP is susceptible to.
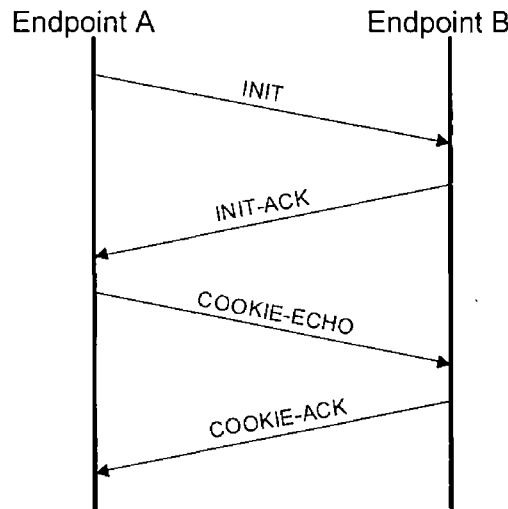
Figure 2.5: SCTP Association Setup Message Exchange

In order to establish an association, an endpoint must either receive an INIT chunk from a peer endpoint, or be explicitly told to initiate an association by the application layer. This latter scenario is achieved when the application layer issues the ASSOCIATE primitive to the endpoint. Once Endpoint A receives the ASSOCIATE primitive from its application layer, it starts an init timer, and sends an INIT chunk to the designated peer endpoint. It then enters the COOKIE_WAIT state. The peer endpoint, Endpoint B in Fig. 2.1, upon reception of the INIT chunk builds a COOKIE, and packs it into an INIT-ACK chunk. It then transmits this INIT-ACK chunk to Endpoint A, and remains in the CLOSED state. Once Endpoint A receives the INIT-ACK, it stops the init timer, copies the COOKIE from the INIT-ACK into a COOKIE-ECHO chunk, starts

a cookie timer and transmits the COOKIE-ECHO chunk to Endpoint B. It then enters the COOKIE_ECHOED state. Endpoint B then receives the COOKIE-ECHO chunk, and upon successful validation of the COOKIE to make sure it was the one it sent to Endpoint A, transmits a COOKIE-ACK chunk to Endpoint A, and enters the ESTABLISHED state. Upon reception of the COOKIE-ACK, Endpoint A stops the cookie timer, and moves into the ESTABLISHED state. The association is now set up and data may be transferred between the two endpoints. This is illustrated by the State diagram shown in Fig. 2.6.
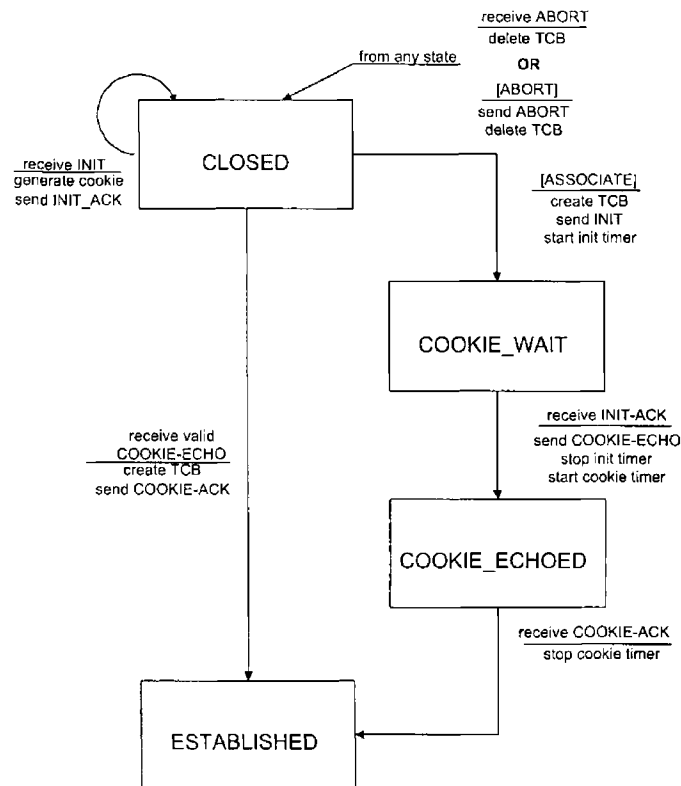
Figure 2.6: SCTP Association Setup State Diagram

Actually user data may be transmitted in the same SCTP packets as the COOKIE-ECHO and COOKIE-ACK chunks, as long as the control chunks are the first chunks in the packet. This was included by the IETF working group to reduce the amount of overhead incurred in establishing an association.

2.4 SCTP Association Termination

There are two ways to close an SCTP association. The first of these, the graceful shutdown, is illustrated in Fig. 2.7. Graceful shutdown involves the exchange of three control chunks. The second method is the abortive shutdown. This involves sending one control chunk, the ABORT chunk, and then moving the association into the CLOSED state. Graceful shutdown is the preferred method of terminating an SCTP association.
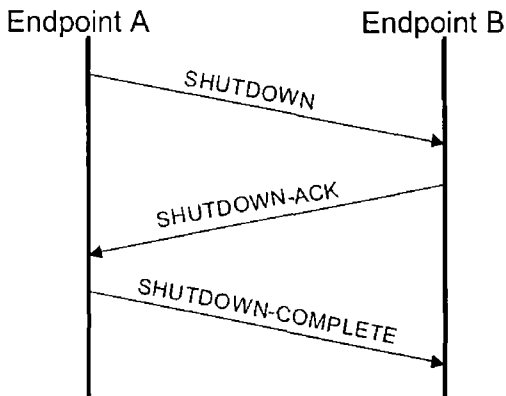
Figure 2.7: Graceful Shutdown Message Exchange

2.4.1 Graceful Shutdown

For an SCTP endpoint to begin the graceful shutdown procedure, it has to be instructed to by the Upper Layer Protocol (ULP). This is the application layer that sits above SCTP. Once the endpoint receives the SHUTDOWN primitive from the ULP it enters the SHUTDOWN_PENDING state. Once in this state the endpoint will no longer accept user data from the ULP. Any data that had been passed down from the ULP prior to the SHUTDOWN primitive, and is currently waiting to be transmitted, is termed pending data. Outstanding data refers to any data that has been transmitted to the peer endpoint but has not been acknowledged yet.

Having received the SHUTDOWN primitive, the SCTP endpoint will transmit any pending data and wait for acknowledgements for all outstanding data before transmit-

ting the SHUTDOWN control chunk to the peer endpoint. After it has transmitted this control chunk, the endpoint starts a shutdown timer and moves the association into the SHUTDOWN_SENT state. This process is illustrated by the State diagram shown in Fig. 2.8.
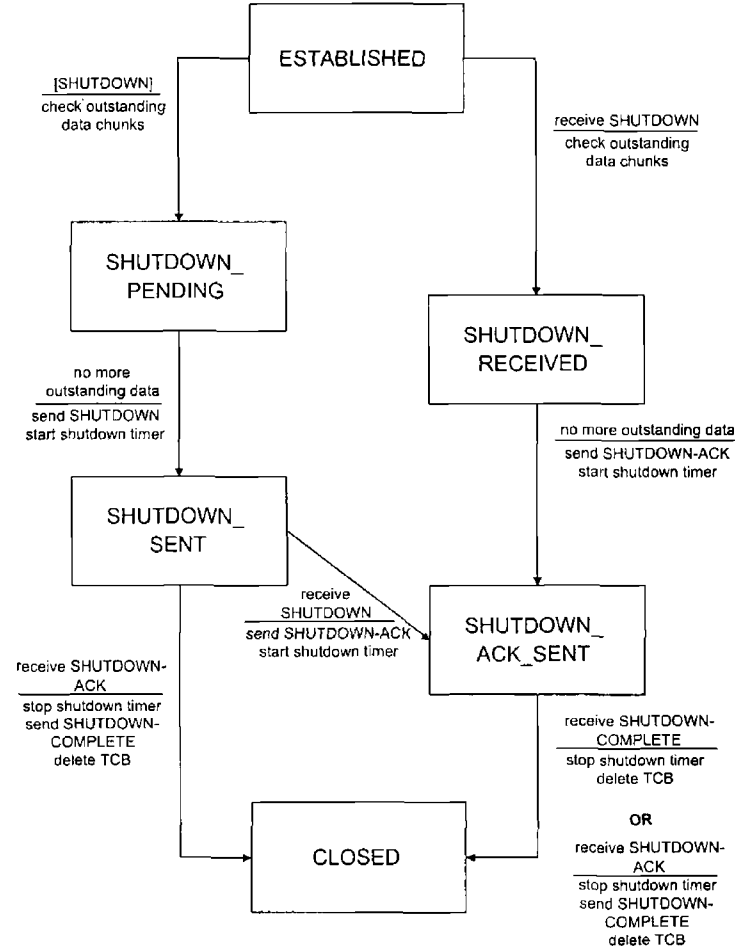
Figure 2.8: SCTP Association Termination State Diagram

The SHUTDOWN chunk contains the chunk header and the cumulative TSN for the association . The peer endpoint uses this to acknowledge any outstanding data chunks it has transmitted. Once the SHUTDOWN chunk is sent any data chunks arriving from the peer endpoint prompt the local endpoint to respond with a SHUTDOWN chunk containing an updated cumulative TSN. Before sending this updated SHUTDOWN chunk, the local endpoint clears any error counter on the association and restarts the shutdown

timer. If the shutdown timer expires, the endpoint should:

1. Retransmit the SHUTDOWN chunk with the cumulative TSN (this may be different from the one included in the previous SHUTDOWN chunk). If the endpoint is multihomed the SCTP specification stipulates that the SHUTDOWN chunk should be transmitted to one of the other destination addresses (if possible).

2. Increment the error counter for the association and the error counter for the destination address. If the destination error counter exceeds a preset maximum set the destination address as **inactive**. If the association error counter exceeds some preset threshold then report the peer as unreachable to the ULP, and move the association into the CLOSED state.

If the endpoint receives a SHUTDOWN-ACK in response to the SHUTDOWN it sent, then the endpoint stops the shutdown timer, transmits a SHUTDOWN-COMPLETE chunk, and moves the association into the CLOSED state.

The above description detailed the graceful shutdown procedure from the perspective of the endpoint that initiated the shutdown. The following describes graceful shutdown from the perspective of the peer endpoint. Upon reception of a SHUTDOWN chunk an endpoint moves the association into the SHUTDOWN_RECEIVED state, as shown in Fig. 2.8. It stops accepting data from its ULP. It continues transmitting any pending data, and waiting for acknowledgement of any outstanding data. Once all data has been acknowledged the endpoint transmits a SHUTDOWN-ACK chunk, starts a shutdown timer, and moves the association into the SHUTDOWN_ACK_SENT state. The procedure for handling the expiration of the shutdown timer in this case is similar to the procedure described for the SHUTDOWN chunk. That is, should the shutdown timer expire, the endpoint should:

1. Retransmit the SHUTDOWN-ACK chunk. As stated previously the SCTP specification stipulates that the retransmission should be to a different destination

address if the endpoint is multihomed.

2. Increment an error counter for the association and one for the destination address. If the destination error counter exceeds a preset threshold, mark the destination address as inactive. If the association error counter exceeds a preset maximum, inform the ULP that the endpoint is unreachable and move the association into the CLOSED state.

If the endpoint receives a SHUTDOWN-COMPLETE chunk whilst in the SHUTDOWN_ACK_SENT state, it should stop the shutdown timer and move the association into the CLOSED state. This completes the graceful shutdown procedure.

2.4.2 Abortive Shutdown

The abortive shutdown is a means of informing a peer endpoint that the association needs to be terminated. It is unreliable and is used in the situation where an application fails, causing an abortive close of the socket, or when an INIT chunk contains invalid mandatory parameters.

To terminate the association by means of the abortive shutdown, the endpoint transmits an ABORT chunk, which may contain error codes. These optional codes contain information about why the endpoint is aborting the association. Once the endpoint transmits the ABORT chunk, it moves the association into the CLOSED state. This can be seen in the State diagram shown in Fig. 2.8. Upon reception of an ABORT chunk, an endpoint first verifies that the ABORT belongs to a current association. To verify the ABORT, the endpoint first checks that it has an association corresponding to the one indicated by the ABORT. Next it extracts the verification tag contained in the SCTP common header of the SCTP packet that delivered the ABORT chunk. It compares this to a valid verification tag. If the two match then the ABORT is valid. The endpoint then moves the association into the CLOSED state. If either of the

above criteria is not met, the ABORT chunk is not valid and is silently discarded by the endpoint.

2.5 SCTP Association Monitoring

When an association is being established, the two endpoints involved inform each other of the IP addresses they can be reached by. These destination IP addresses are exchanged in the INIT and INIT-ACK chunks. In the case of multihomed endpoints, a set of IP addresses defined for the endpoint is conveyed to its peer.

When its peer is multihomed, an SCTP endpoint will choose one of the peer's destination addresses as the primary destination address. All of the peer's other destination addresses are termed secondary and are used to provide path redundancy. Under normal operation, all data to be sent to the peer is transmitted to the primary address. The secondary addresses will not receive user data unless a problem occurs with the primary.

A destination address is considered idle if it has not been communicated with during an interval known as the heartbeat period. This interval is a configurable value, but is usually set to be 30 seconds. Communication consists of transmitting a chunk capable of updating the RTT for the destination address, to the idle destination address. Chunks capable of updating a destination address's RTT are INIT, COOKIE-ECHO, the first-transmission DATA, and HEARTBEAT chunks. The INIT and COOKIE-ECHO chunks are only transmitted once during the lifetime of an association, under normal working conditions. As DATA chunks are only transmitted to the primary destination address of the peer endpoint, the HEARTBEAT chunk is used to update the RTT of all secondary destination addresses. This chunk is transmitted to an individual destination address periodically. Upon reception of a HEARTBEAT chunk, the receiver immediately generates a HEARTBEAT-ACK chunk, copies the chunk data from the HEARTBEAT to

23

the chunk data field of the HEARTBEAT-ACK. The HEARTBEAT chunk data contains a timestamp in TLV format. This HEARTBEAT-ACK is then placed within an SCTP packet and transmitted back to the peer endpoint. According to [12], the HEARTBEAT-ACK is transmitted back to the destination address that the HEARTBEAT originated from, i.e. the destination address contained in the source IP address field of the IP packet that contained the HEARTBEAT chunk.

The timestamp contained in the HEARTBEAT-ACK, that was copied from the received HEARTBEAT chunk, is used to calculate the RTT for the destination address. This heartbeating process is repeated for each of the other secondary destination addresses in turn. The RTT of the primary address is usually updated using DATA chunks.

The heartbeating process is also used to determine whether a destination address has become unavailable. Such destination addresses are termed inactive. The procedure for detecting an inactive destination address is an integral part in the current handover scheme implemented in SCTP. As such, a full explanation of the procedure, including the eventual handover to one of the secondary addresses, is detailed in Section 4.1

The inability of the current transport layer protocols, TCP and UDP, to provide support for telephony signalling over IP networks, led the IETF to develop an alternative solution, entitled SCTP. Originally designed for the sole purpose of transporting telephony signalling, the scope of SCTP was expanded to incorporate transportation of other traffic with similar requirements to telephony signalling. This expansion led to the standarization of SCTP as the third general purpose transport layer protocol, alongside the more established TCP and UDP.

SCTP is message based like UDP but contains congestion control schemes that are based on the schemes used in TCP. SCTP provides a reliable connection-oriented service to applications running on top of it. It has been designed to overcome some of the limitations of TCP and UDP, such as head-of-line blocking and the lack of congestion

24

control. The features of SCTP therefore, can be viewed as enhancements over those of TCP and UDP.

Chapter 3

Handover and IP Mobility

The popularity of cell phones in the 1990s, and the ability to communicate regardless of location, has led to a desire among current users to access services that were previously exclusive to the wired domain. The introduction of IP-based wireless access networks, such as WLAN, has mitigated the deployment of these services to wireless users. However, one of the characteristics of cellular networks is the ability to remain in communication whilst traversing cell boundaries, a practice called roaming. Central to roaming is the concept of handover. In the context of GSM, handover allows for a telephone call in progress to be transferred from the current serving basestation to a more appropriate basestation as the caller moves with minimal disruption to the call. Providing this ability, termed host mobility, to IP-based wireless users has been the mandate of Mobile IP. Recently, it has been proposed that SIP, with minor modifications, could also accommodate host mobility.

3.1   GSM Overview

Due to the high costs of subscriber equipment, and the lack of interoperability between Europe's already established analog cellular networks, the Conference of European Posts and Telecommunications (CEPT) created the GSM committee. The committee was

given the task of developing a pan-European cellular system that would operate in the 900MHz band.

One of the first decisions made by the committee was that the new system would be digital instead of analog. Having established this criterion, the committee began entertaining proposals for this new digital system. Each of the nine proposals submitted [16] were assessed against the following [17]:

- spectrum efficiency

- mobile cost

- base station cost

- subjective voice quality

- hand-portable feasibility

- co-existence with current systems

- the ability to support new services

The final candidate chosen, a proposal by ELAB of Norway [18], was a Time Division Multiple Access (TDMA) system, with a carrier spacing of 200kHz, and employing a Gaussian Minimum Shift Keying (GMSK) modulation scheme.

3.1.1 Cellular Systems

Like the analog systems that it superseded, GSM was designed to utilize a cellular structure. Systems using this structure are composed of cells where a cell is defined as the coverage of one transmitter, called a Base Transceiver Station (BTS). A single cell is illustrated in Fig. 3.1. Though the actual boundary of a cell is irregular in shape, it is often shown as a hexagon for ease of design [19]. One of the advantages of cellular systems is the feature of frequency reuse. This feature allows channels that are
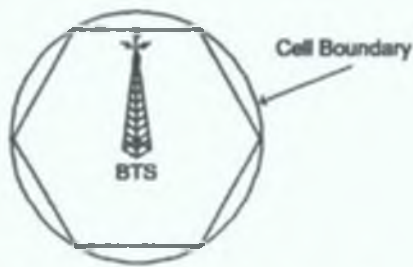
Figure 3.1: A single cell

used in one cell to be reused in other cells. This reuse of the limited frequency band increases the capacity of the system. In the case of GSM, frequency reuse increases the number of simultaneous voice calls that a network can support. However, the reusing of frequencies is contingent on the distance between cochannel cells. These cells cause cochannel interference that can limit the performance of the system.

Cellular planners need to devise a reuse pattern that will incur minimal cochannel interference in any cell in a particular cluster. A typical seven cell repetition pattern is shown in Fig. 3.2. Other repetition patterns may also be used [20]. There are other factors that affect the interference levels within a cell, e.g. adjacent channel interference. See [18, 20] for more information.
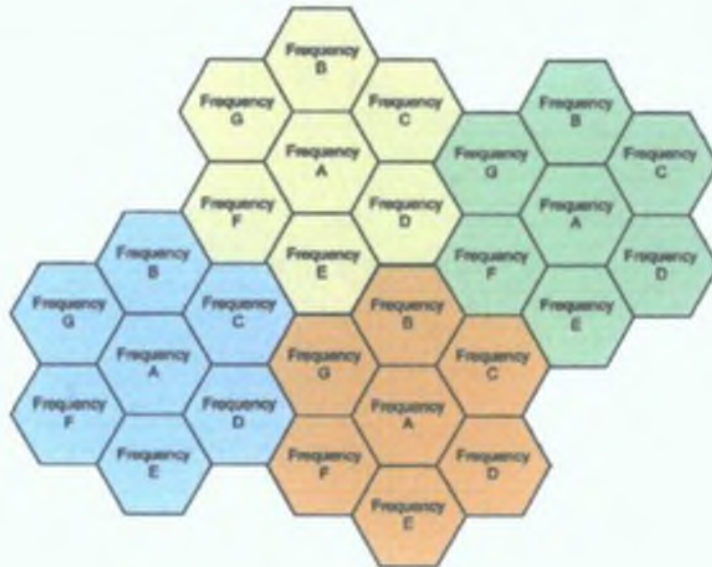
Figure 3.2: Seven Cell Repetition Pattern

28

3.1.2 The GSM System

GSM operates in both the 900MHz and 1800MHz wavebands. Within the 900MHz waveband, two 25MHz-wide frequency bands have been allocated to GSM. The first of these bands is between 890 and 915MHz and is called the uplink in GSM terminology. Uplink communications are from the Mobile Station (MS) to the BTS. The other 25MHz-wide frequency band is the downlink and occupies the spectrum from 935-960MHz. Downlink communications are from the BTS to the MS. The uplink and downlink frequencies for both the 900 and 1800MHz bands are shown in Table 3.1.

	900MHz	1800MHz
Uplink Frequencies	890-915 MHz	1805 - 1880 MHz
Downlink Frequencies	935-960 MHz	1710 - 1785 MHz

Table 3.1: Uplink and downlink frequency bands for GSM

GSM uses a combination of Frequency Division Multiple Access (FDMA) and TDMA to divide up these bands. Under this scheme, each of the 25MHz frequency bands allocated to GSM is divided into 124 frequency channels, each with a separation of 200KHz between them. Each of these channels is further divided in time using TDMA. A GSM subscriber transmits in periodic bursts called timeslots. A timeslot is 0.577ms in duration. Eight timeslots constitute a TDMA frame.

Each TDMA frame is assigned a frame number. This frame number wraps around every 3h 28m 53.76s. This period is referred to as a hyperframe. The hyperframe is composed of 2048 superframes. Each superframe is itself composed of 51 26-multiframes or 26 51-multiframes. The 26-multiframe is composed of 26 TDMA frames, each of 120ms duration. These frames contain Traffic Channels (TCHs) aswell as the associated control channels. TCHs carry speech or user data up to 9.6kbps. Control channels are used for network management and channel maintenance tasks. Some of the common control channels are listed in Table 3.2. The 51-multiframe is composed 51 TDMA frames, each of 235.8ms duration. Each of these frames contain only signalling data.

29

Channel		Description
Broadcast Control Channel	BCCH	Provides the MS with the parameters it needs to identify and access the network
Synchronous Channel	SCH	Gives MS the training sequence needed to demodulate BTS transmissions
Stand-alone Dedicated Control Channel	SDCCH	Used to setup calls
Frequency Correction Channel	FCCH	Supplies the MS with the frequency of the system to synchronize with the network
Fast Associated Control Channel	FACCH	Used in handover
Slow Associated Control Channel	SACCH	Carries measurement parameters needed to maintain link between MS and BTS
Paging Channel	PCH	Used to alert a MS to an incoming call

Table 3.2: GSM Control Channels

The GSM system consists of three separate sections, as seen in Fig 3.3. The first of these section is the Mobile Station.

Mobile Station

The MS comprises of the Terminal Equipment (the handset), and the Subscriber Identity Module (SIM) card. The SIM card contains, amongst other information, an International Mobile Subscriber Identity (IMSI) number and a secret authentication key. These security measures allow a subscriber to be validated onto a network, and is a preventative measure against cellphone cloning. The SIM card may also be protected by a Personal Identity Number (PIN) to ensure no unauthorized access can occur.
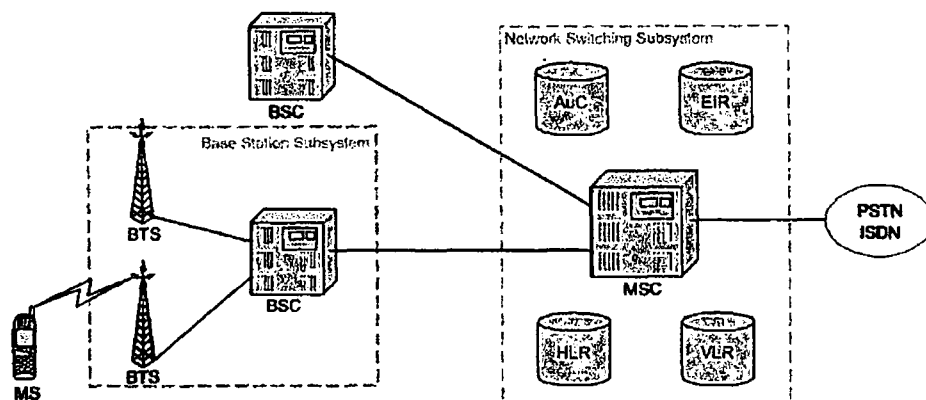
Figure 3.3: GSM System

Base Station Subsystem

The Base Station Subsystem (BSS) is composed of one Base Station Controller (BSC) and one or more BTS. The BTS communicates with the MS over the U_m interface. This interface is also called the air interface or the radio link. The BTS communicates with the BSC over the A_{bis} interface. The BSC controls the radio resources (radio channel setup, frequency hopping, handover) for all the BTSs it is responsible for. The BSS provides the conduit between the MS and the Mobile services Switching Centre (MSC).

Network Subsystem

The network subsystem provides the switching capability of the network, similar to a switching node in the Public Switched Telephony Network (PSTN), or Integrated Services Digital Network (ISDN). Additionally it has all the functionality required to accommodate a mobile subscriber. This includes registration, authentication, location updating, handover, and call routing to a roaming subscriber. The central entity in the network subsystem is the MSC. One of the MSC's tasks involves providing an interface to the fixed networks (PSTN, ISDN), allowing a mobile subscriber to call a person on a fixed line, and vice versa.

The signalling between the functional entities in the network subsystem uses the

31

SS7. SS7 is also used in the PSTN, and for trunk signalling in ISDN.

Call routing and the ability to roam, central concepts in a mobile network, are provided by the Home Location Register (HLR) and the Visitor Location Register (VLR) in conjunction with the MSC. The HLR is a database that contains all administrative information belonging to a subscriber in a GSM network. Amongst this information is the current location of the subscriber. This is typically the SS7 signalling address of the VLR servicing the area that the subscriber is currently located in. There is one HLR per GSM network, though it may be a distributed database. The VLR contains a subset of the subscriber information contained in the HLR. This information is used to provide call control and access to the subscribed services for each mobile located in the VLR's service area. The VLR is usually co-located with the MSC. This simplifies the signalling between the two entities.

The other two registers are used for security and authentication purposes. The Equipment Identity Register (EIR) contains a list of all valid subscriber equipment on the network[1]. The Authentication Centre (AuC) is a database that contains a copy of the secret authentication key stored on the subscribers SIM card. This is used to validate a subscriber onto the network.

3.1.3   GSM Handover

In GSM, handover can be described as the process of switching a call from one cell to another cell without interrupting the call in progress. The reasons to perform handover could involve:

1. The quality of the radio link between the MS and the serving BTS has decreased due to increased interference in the cell

2. The call in progress is causing interference to other users in the cell

---

[1] Each MS handset is identified by its International Mobile Equipment Identity (IMEI) number.

3. The subscriber is moving out of range of its serving BTS

4. The effects of multipath fading has reduced the quality of the call

In the case of point 4, performing handover to a different channel on another frequency in the same cell could alleviate the problem. This is known as Intra-cell handover.

The possible types of handover include:

- Intra-cell handover

- Handover between cells controlled by the same BSC

- Handover between cells controlled by different BSCs, but the same MSC

- Handover between cells controlled by different MSCs

In order to illustrate the handover procedure in GSM, only handover between cells controlled by the same BSC will be described.

In order to determine if a handover should occur, and to which BTS, several measurements need to be gathered [18]. These measurements are performed by both the MS and the BTS. The decision to perform a handover is also based on other network characteristics, such as network load.

The measurements gathered by the MS include:

- The downlink receive level from the serving BTS

- The quality, or Bit Error Rate (BER), of the downlink signal

- The downlink receive levels from neighbouring BTSs

The measurements obtained by the serving BTS include:

- The uplink receive level from the MS

33

- The BER of the uplink signal

- The distance between the MS and the BTS, based on the adaptive timing advance parameter

- The interference level in unallocated timeslots

Once these measurements have been determined, a decision on whether to perform handover or not is made. In the case where handover is to occur the BTS to handover to has already been resolved based on the receive levels of neighbouring BTSs gathered by the MS.

Adopting the naming convention used in [21] this new BTS, BTS-new, is requested by the BSC to reserve a channel for the imminent handover. The BSC makes this request by issuing an "RSM Channel Activation" message to BTS-new [19]. BTS-new reserves the channel, if available, and returns an "RSM Channel Activation Acknowledge" message. The BSC then sends an "RIL3-RR Handover Command" to the MS on the FAACH via the BTS, BTS-old, that is currently serving the MS. This message assigns the new channel and contains information regarding the new channel's characteristics, what power level to use, and whether asynchronous or synchronous handover is to be used. The message also assigns a new SACCH. Once the MS has received this message, it releases the old channel and tunes to the new channel.

If synchronous handover is used, the MS transmits four "RIL3-RR Handover Access" messages to BTS-new over the FAACH. BTS-new, upon reception of these messages activates the channel in both directions, and may send an "RSM Handover Detection" message to the BSC.

In the case where the handover is asynchronous, i.e. BTS-old and BTS-new are not synchronized, the MS sends a continuous stream of "RIL3-RR Handover Access" messages to BTS-new until it receives an "RIL3-RR Physical Information" message. This message contains the timing advance to apply.

Once the timing issues have been resolved, the MS transmits an "RIL3-RR Handover Complete" message to the BSC over the new FAACH. This procedure can be seen in Fig. 3.4.
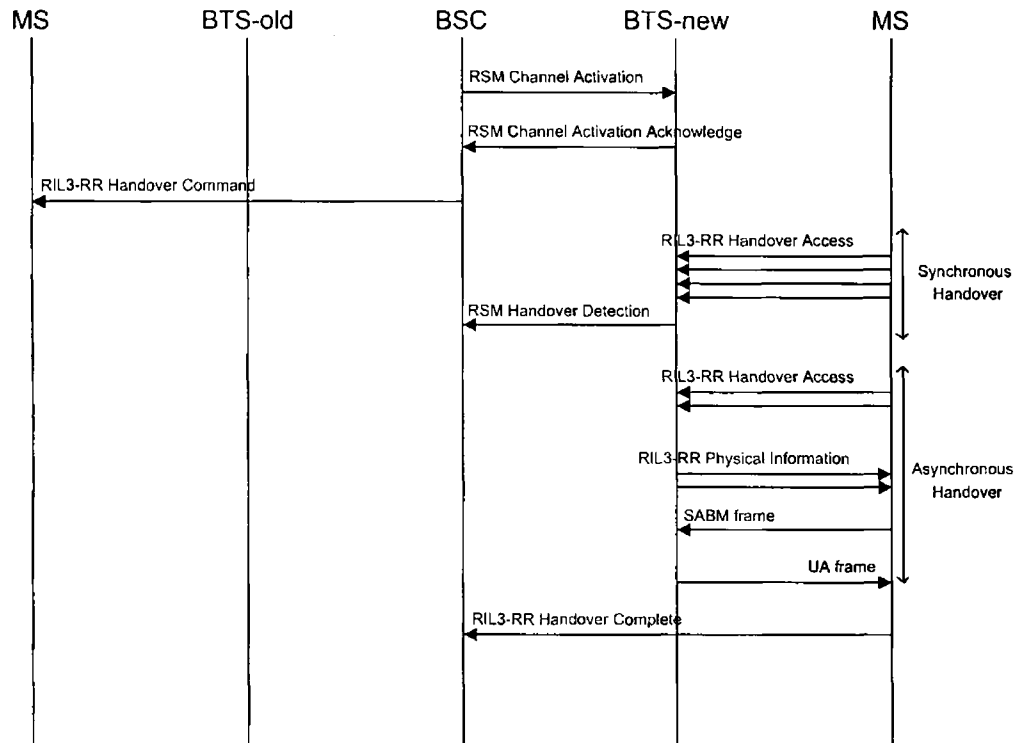
Figure 3.4: Reproduced from [19]. Handover between cells controlled by the same BSC

3.1.4 General Packet Radio Service (GPRS)

The increased interest among mobile subscribers for services similar to those of packet-switched networks led to the deployment of such services within networks such as GSM. The data rates these 2nd Generation, or 2G, networks offer is limited when compared to wired networks. To address this deficiency development began on higher rate, next generation networks such as UMTS. These networks, titled 3G, or third generation systems, require a new access infrastructure to be in place before they can be deployed. The prohibitive cost of this infrastructure has delayed the deployment of 3G systems. In the interim, solutions such as GPRS have been developed. These solutions, termed 2.5G,

are essentially overlay networks, deployed to enhance the data provisioning capability of 2G networks before 3G networks are rolled out.

GPRS provides a packet oriented data service to circuit-switched GSM networks, allowing GSM subscribers access to services that were previously the exclusive domain of data networks such as IP (Internet). The data rates offered by GPRS range from 9.6kbps to 114Kbps. GPRS functionality can be added to existing GSM networks through the introduction of two supporting nodes, the Serving GPRS Support Node (SGSN), and the Gateway GPRS Support Node (GGSN). The SGSN translates traffic between the IP and radio network protocols. It also tracks the movement of the user so as to determine where to deliver incoming packets to. The GGSN is the gateway that connects the GPRS network to other networks such as the Internet.

The higher data rates of GPRS are achieved by reserving one or more timeslots when a subscriber has data to send. These resources are released when the data has been transmitted. This is in contrast to circuit-switched GSM, where a timeslot is assigned to a subscriber for the duration of the call. This provides more efficient use of the scarce radio resources. As timeslots are only reserved when a subscriber has data to send, many subscribers can utilize the same timeslots provided they don't all wish to transmit at the same time. This forms the basis for GPRS's "always on" service.

3.2 WLAN Overview

Data services provided by 2G and 2.5G systems are quite limited in bandwidth they offer. Third generation (3G) cellular systems, such as UMTS, provide higher data rates than current systems, such as GSM and GPRS. These 3G systems form the backbone of an emerging wireless access infrastructure. This infrastructure is composed of wide-range, low-rate mobile systems such as UMTS and higher bandwidth, short range access technologies. Among the latter is IEEE 802.11, the standard for WLANs.

IEEE 802.11 operates in the Industrial, Scientific, and Medical (ISM) transmission band. The standard has several high speed variants, the most popular of which is IEEE 802.11b, known commercially as WiFi, which supports transmission rates up to 11Mbps. Its popularity can be attributed to these high transmission rates, the relative low cost and ease of installation of WiFi hardware, and the increasing proliferation of "hotspots". Hotspots are WiFi coverage areas, and are currently being deployed in popular consumer areas such as coffee shops, fast food restaurants, airports, and hotels.

802.11 supports two modes of operation. The first of these two modes, Distributed Coordination Function (DCF) does not employ any central control. The second mode, called Point Coordination Function (PCF) uses the base station to coordinate all activity in the cell.

3.2.1   Distributed Coordination Function

The medium access mechanism used in Ethernet Local Area Networks (LANs) is entitled Carrier Sense Multiple Access with Collision Detection (CSMA/CD). Under this scheme, a station that has begun transmitting data continually monitors the medium to determine if a collision has occurred, and if a collision is detected, stops transmitting data and waits a random back-off period. However, the half-duplex radio transceivers used in WLAN equipment cannot simultaneously transmit data and listen for collisions on a single frequency. Due to this limitation of the WLAN transceivers, coupled with the hidden station and exposed station problems (see Section 3.2.4), mean that 802.11 can not use CSMA/CD as Ethernet does. This led to the development of 802.11's own medium access method, entitled Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). This mechanism is a variation of the CSMA/CD mechanism used in Ethernet.

DCF employs CSMA/CA as its medium access method. Two methods of operation are supported by DCF. The first of these is the simplest case whereby a station upon

37

sensing the medium to be idle, begins transmitting an entire frame of data. If the channel is busy the station defers its transmission to a later time. If a collision occurs, the colliding stations wait a random backoff period before transmitting again. This backoff period is determined using the Ethernet binary exponential backoff algorithm [22].

The second method of operation involves both physical channel sensing, as in the first method, and virtual channel sensing. A station wishing to transmit first senses the channel to determine if it is idle. If so, it waits a random backoff period before sensing the medium again. This is to minimize the chance of colliding with another station who also has data to transmit and has sensed the channel to be idle. If the channel is still idle after the second check, then the station wishing to transmit sends a short control frame, a Request To Send (RTS), to the destination station that it wishes to transmit its data to. If the destination station is willing to receive the data, it responds with the Clear To Send (CTS) control frame. Upon reception of the CTS the first station begins to transmit its data. The destination station acknowledges receipt of the data with an ACK control frame. This is sent once the data frame is received and free from errors. In the case where the frame has been corrupted, the destination station will transmit a NAK control frame back to the data sender informing it of the corruption of the frame.

All other stations in the cell, upon receiving either the RTS or CTS (or both), are informed of the impending transmission. They defer from attempting to transmit any of their data until the impending transmission is completed. Information contained in the RTS and CTS allow each of the other stations to estimate how long the impending transmission will last. This value is stored in a Network Allocation Vector (NAV) at each of the stations.

Employing either of these methods does not make 802.11 in any way reliable. In fact due to the nature of radio transmission, and the noise inherent in this form of communication, 802.11 is even more unreliable than Ethernet. However, by employing

the above methods the chances of collisions occuring are reduced.

3.2.2 Point Coordination Function

In contrast to DCF's lack of central control, all transmissions in PCF mode are controlled by the base station. Stations are polled periodically to determine if they have data to send. Central to this scheme is the idea of the beacon frame. This is a control frame that is broadcast by the base station 10-100 times a second. The beacon frame contains system information such as clock synchronization. The 802.11 standard specifies the polling mechanism (beacon frame) however, it does not specify polling frequency, polling order, or whether equal service to all stations needs to be guaranteed.

3.2.3 Coexistence

It is possible to mix both modes of 802.11 in a single cell. That is DCF and PCF can coexist. The 802.11 standard makes provisions to accommodate this. After a frame has been sent a period of dead time is required before a station can transmit a frame. Within this period, four different time intervals are specified, as shown in Fig. 3.5. Each time interval is defined for a specific purpose. The shortest of these intervals is the Short
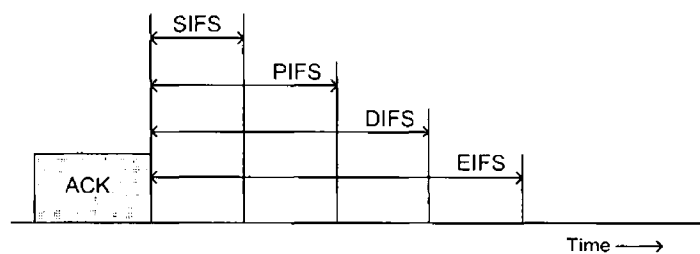

Figure 3.5: 802.11 Interframe Spacing

InterFrame Spacing (SIFS). It is used to give two stations, already in communication with each other, an "advantage" over the other stations in the cell. This advantage can involve letting the receiver transmit a CTS in response to a RTS, letting the receiver

send an ACK to a data frame, and letting the sender of a fragment burst[1] send the next fragment without having to send an RTS first.

There is exactly one station permitted to transmit after a SIFS interval. If it does not avail of its opportunity and a PCF InterFrame Spacing (PIFS) interval elapses then the base station may send a beacon frame or a poll frame. The PIFS provides a means for letting a station sending a fragment sequence to finish sending its data without other stations intervening. It is also a means by which the base station can seize control of the channel after a station has finished transmission without having to contend with the other stations in the cell.

After the next interval, DCF InterFrame Spacing (DIFS), has elapsed, if the base station has nothing it wishes to broadcast, then the channel is considered idle in the DCF sense, i.e. stations can attempt to seize control of the channel using CSMA/CA.

The final interval, the Extended InterFrame Spacing (EIFS) is used by a station to report a bad or unknown frame it has just received. It is given the lowest priority so as not to interfere with an existing communication between two stations.

3.2.4 Hidden/Exposed Station Problem
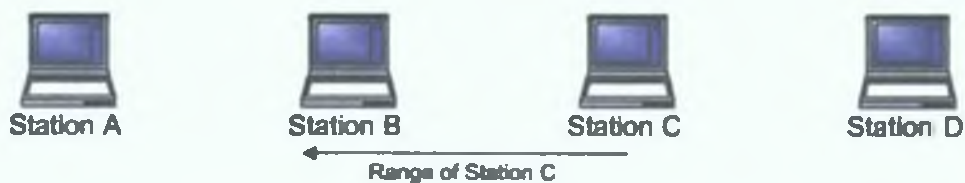
Figure 3.6: Hidden/Exposed Station Problem

In Fig. 3.6, Station A and Station B are within range of one another and so can potentially interfere with each other. Station C can also potentially interfere with Station B and Station D but not Station A, as it is out of Station C's range.

---

[1]802.11 allows for data frames to be divided into individually numbered fragment blocks to counteract noisy radio channels. A collection of fragments is termed a fragment burst.

As Station A is not within the transmission range of Station C any transmission from Station A to Station B will not be detected by Station C. Station C will then determine (falsely) that the channel is idle and will begin transmitting to Station B. At Station B, the transmission from Station C will interfere with the transmission from Station A. This is termed the hidden station problem.

The inverse of this problem is termed the exposed station problem. In this scenario, Station B upon sensing the channel to be idle begins transmitting to Station A. Station C, upon sensing the channel, determines that Station B is transmitting and defers its transmission to Station D. However, Station C's transmission would only cause interference in the region between Stations B and C, which are not the intended receivers of any current transmissions.

3.3 Mobile IP

The Internet is comprised of internetworks connected together via a core backbone network. Each of these internetworks is managed independant of the other internetworks, and as such is termed an Autonomous System (AS). Each AS contains interior gateways for routing of packets between hosts located within the same AS, and exterior gateways to route packets across the backbone network to hosts located in other internetworks.

Typically in an IP network, host computers are identified by their IP addresses. These addresses are composed of a network ID and a host ID. IP packets destined for a host in a particular AS are routed using the network ID part of the address, through intervening exterior gateways, to the exterior gateway on the host's home network. From here the IP packets are routed to an interior gateway which locates the physical machine that corresponds to the host ID part of the IP address using a solution such as Address Resolution Protocol (ARP) [23]. It then forwards the packets on to this machine.

This routing scheme reduces the size of the routing tables at the exterior and interior gateways, and thus the latency in finding the corresponding route, as each gateway only needs to lookup how to reach the network specified by the network ID in its routing tables to deliver packets to all hosts on that network. The routing tables for interior gateways are updated periodically using an interior gateway protocol, such as Routing Information Protocol (RIP) or Open Shortest Path First (OSPF). Similarly, the exterior gateways routing tables are updated periodically using an exterior gateway protocol, such as Border Gateway Protocol (BGP).

This scheme works well for static hosts. However, with the increasing adoption of wireless computing devices, there has been a growing desire among users to access the services of wired networks while on the move. This has led to a problem with regards routing. If a host were to move out of its home network to a new network then IP packets destined for it would still be routed to the home network and could not be delivered to the host. A solution would be to issue the Mobile Host (MH) with a new IP address upon entering the new network, allowing packets to be routed to this network instead of the previous one. However, this would require informing large numbers of people, programs and databases of the IP address change. This is not feasible on a network the size of the Internet.

The IETF set up a working group to address the problem of mobility in IP networks. The working group's mandate included:

1. Mobile Host's home address must be portable (i.e. used anywhere)

2. The solution should involve making no changes to the fixed hosts

3. The solution should involve making no changes to routers and routing tables

4. No detours for packets destined for mobile hosts

5. There should be no overhead incurred for a mobile host on its home network.

42

The solution the working group devised was entitled IP Mobility Support for IPv4 [24]. This solution is known more commonly as Mobile IP. It introduces two new network entities. The first of these is the Home Agent (HA). This is a router that resides in the Mobile Host's home network. It intercepts packets destined for the MH when the MH is on another network. It forwards these packets to the MH at the new network. It also informs any Corresponding Nodes (CNs), hosts that wish to communicate with the MH, of the new location of the MH. The second agent is the Foreign Agent (FA). This is a router located in the network that the MH has moved to. It informs the HA of an address, the Care-of Address (CoA), that it can use to reach the MH while it is present in the foreign network. Fig. 3.7 shows the components of a Mobile IP system.
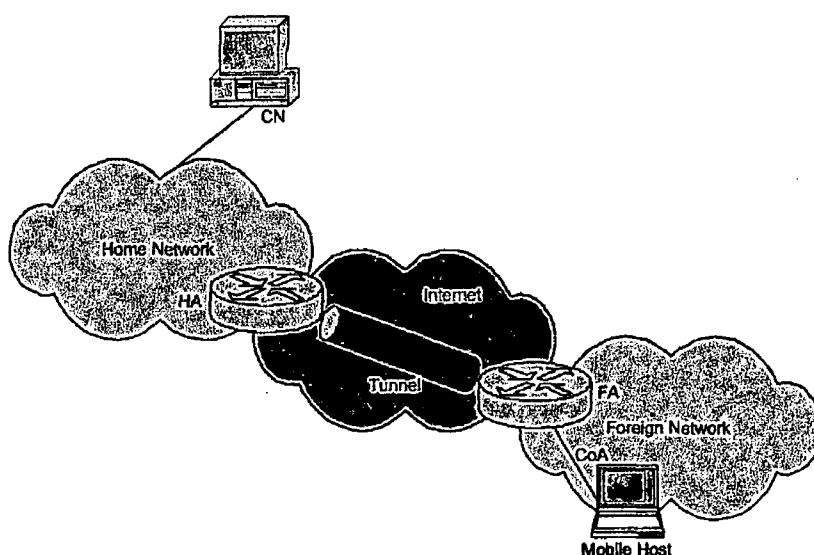

Figure 3.7: Mobile IP Components

3.3.1 How Mobile IP works

When a Mobile Host enters a foreign network, it listens for advertisements being broadcast by the FA. If it does not receive any advertisements the MH may broadcast a message inquiring if there are any FAs present. Once a FA has been found, the MH

43

will register with it. After registration has taken place, the FA contacts the HA and informs it of a CoA that the HA may use to contact the MH while it resides on the HA's network. The CoA is usually the IP address of the Foreign Agent.

When a CN wishes to communicate with a Mobile Host it transmits packets as normal. Its packets are routed, according to normal IP routing practices, to the MH's home network. Here the packets are intercepted by the HA. The HA encapsulates the packets and tunnels them to the CoA. Upon reception of the packets from the HA the FA decapsulates the packets and delivers the original packets to the MH. Subsequently the HA informs the CN of the MH's CoA so that it may route any subsequent packets directly to the foreign network.

Before the MH leaves the foreign network it de-registers with the FA. A problem exists if a MH does not perform this de-registration. The problem is similar to the original problem of routing to a host that no longer resides on a particular network. If the MH leaves the foreign network without de-registering the FA will be unable to deliver any packets that were destined for the MH. The working group devised a solution to this problem. Their solution involves making registration valid only for a certain period of time. After this period if the registration is not refreshed, it times out and the entry for the MH is cleared from the FA's routing table.

While Mobile IP provides a means for IP networks to support user mobility, there are some disadvantages with implementing the solution [2]. These include:

- There is a high handover delay associated with Mobile IP. A Mobile Host must determine a new CoA and register with its HA before the HA can forward the packets to the MH [25].

- When a MH moves from one network to another, it must register its new CoA with the HA. However, in the time it takes for the MH to perform this registration with the HA, packets sent to the MH will be lost, as the MH is no longer at its old CoA and the HA does not yet know the MH's new CoA [26]

- In the base specification of Mobile IP, a lot of resources are wasted due to the tunneling involved in transmitting packets from the HA to the MH, via the FA. As the number of Mobile Hosts increases, the load on the agents will also increase. This could lead to failure of an agent, which could result in many MHs not receiving forwarded packets [25].

- Mobile IP does not work well when the HA is behind a firewall, or when the foreign network employs ingress filtering [7]. There are solutions to these problems, such as firewall traversal [27] and reverse tunnelling, where the HA's IP address is used as the exit point of the tunnel [2].

3.3.2 Cellular IP

Cellular IP [28] aims to reduce the effect of some of these disadvantages, namely the high handover latency and the excessive traffic generated by the tunnelling of packets. It proposes to do this by dividing user mobility into local-area and wide-area mobility. Mobile IP can facilitate wide-area mobility, also known as macro-mobility. Cellular IP manages the local-area mobility, or micro-mobility. Under this scheme, Cellular IP handles all handovers within a certain wireless access network. If the MH moves out of the current access network into a different wireless access network, e.g the MH moves into the coverage of a wireless network controlled by a different WISP, Mobile IP handles the handover to this new network. This solution has the advantage of reducing the load on the both the HA and the FA. This will be of benefit as the number of mobile users increases.

Route mappings are used to deliver packets to a mobile host in a Cellular IP system. No node in a Cellular IP network has a full definition of the topology of the network. Each node merely forwards all packets destined for a particular MH on to the next node along the path. Route mappings are updated by packets emanating from the MH

45

towards the Gateway Router[1]. All nodes along the path update their routing caches upon reception of the packets. In order to reduce the overhead required to locate a MH, and so as not to overload the network, two caches are maintained at each node. The Routing Cache is used to monitor the position of active MHs. Idle MHs are located using the Paging Cache maintained by each node.

Like Mobile IP, there are drawbacks to employing Cellular IP. Through the separation of wide-area and local-area mobility Cellular IP reduces the load on the HA and FA, and also decreases the amount of traffic that needs to be transmitted between the entities. However, Cellular IP still relies on Mobile IP for movement between different wireless access networks, thus all the problems associated with Mobile IP are also associated with Cellular IP (triangular routing, high handover latency between foreign networks) though these problems are reduced compared to employing Mobile IP exclusively. The usefulness of Cellular IP is also limited by the size of the wireless access network that it is employed in. Small access networks will require frequent Mobile IP-assisted handovers. Large wireless access networks will require larger caches, and more frequent paging to maintain these caches.

3.4 SIP Overview

Mobile IP provides a means for a mobile host to remain connected to subscribed services whilst roaming. These services may consist of multimedia traffic which have strict real-time constraints which are unknown to the network layer, the layer at which Mobile IP resides. SIP, an application layer protocol, could provide a means of route optimization, which could improve the performance of these real-time services.

SIP, as defined in [29], was designed to allow voice calls to be conducted over IP networks, such as the Internet. The transmission of voice packets over the Internet, a process known as Voice over Internet Protocol (VoIP), was the almost exclusive domain

---

[1]The Gateway Router can be co-located with the FA of Mobile IP

of the ITU's H.323 protocol. However, the H.323 protocol suite was considered by some people in the Internet community to be a complex and inflexible product [22]. As the H.323 protocol is more of an "umbrella" standard, incorporating many task-specific protocols, support of newer services would require a revision of the suite to be developed and deployed. It was this growing dissatisfaction with H.323's inflexibility that led the IETF to approach the issue of VoIP in a more modular way. The result of this approach was SIP.

SIP is a single module that interworks with already established protocols. One of its design mandates was to reuse existing protocols wherever possible. As such SIP is only responsible for the setting up, management, and termination of sessions. SIP does not know the details of the session. This is left to the protocol handling the data transfer for the session (e.g RTP [30]).

SIP is text-based. It is modelled on HTTP [31] and represents telephone numbers as URIs, e.g. sip:alice@example.com. In this example, the SIP user alice is located at the domain specified by the DNS name example.com. SIP URIs can also be composed of IPv4 addresses, IPv6 addresses, or actual telephone numbers. SIP messages, described by Session Description Protocol (SDP), contain a SIP method on the first line and additional parameters, such as the endpoint's multimedia capabilities and supported format types. The six methods defined in [29] are shown in Table 3.3.

Method	Description
INVITE	Request session initiation
ACK	Confirmation of session initiation
BYE	Request session termination
OPTIONS	Inquire about a host's capabilities
CANCEL	Cancel pending request
REGISTER	Update a redirection server about a user's (new) location
INFO	Sent mid-session. Does not alter session state

Table 3.3: SIP Methods

The replies from SIP servers are coded response messages. The codes used by SIP

are of the same type as those used by HTTP (e.g 500 - Internal Server Error). The codes and their meanings are given in Table 3.4 [22].

Code	Meaning	Example
1xx	Information	180 - ringing
2xx	Success	200 - ok
3xx	Redirection	301 - moved
4xx	Client Error	404 - not found
5xx	Server Error	500 - internal server error
6xx	Global Error	600 - busy everywhere

Table 3.4: SIP Response Codes

3.4.1 SIP System

A SIP system is composed of SIP User Agents, SIP Proxy Server, SIP Redirect Server, and SIP Registrar Server. A SIP system is illustrated in Fig. 3.8.
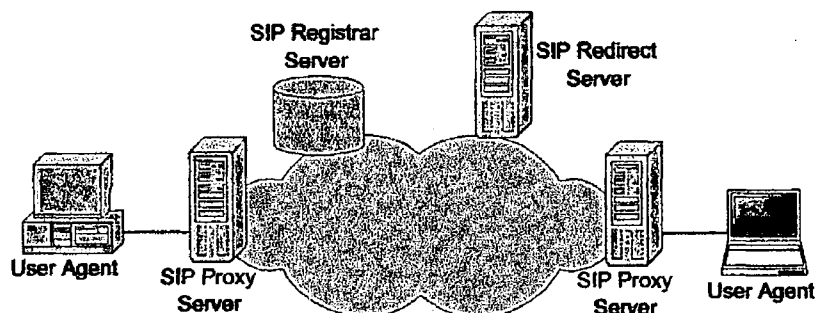

Figure 3.8: SIP System

- User Agent (UA)

  These are clients (UAC) and servers (UAS). A UAC issues requests to a UAS. The UAS replies with coded response messages.

- Proxy Server

  These accept session requests from UA and query the SIP Registrar to locate the recipient UA. The proxy server then forwards the request on to the recipient UA.

48

- Redirect Server

  These allow SIP Proxy Servers to direct SIP requests to other domains.

- Registrar Server

  These are databases containing location of all UAs in a domain.

SIP allows clients to negotiate the services they wish the session to have. SIP also provides call management facilities such as adding, dropping, or transferring calling parties. SIP can be used to change a session's features while the session is ongoing. All other functionality is provided by other protocols.

3.4.2 SIP Device Mobility

Mobile IP allows for the change of IP address due to host mobility to be hidden from higher protocol layers. This transparent mobility prevents active TCP connections from failing as the mobile host moves from network to network. Services with real-time constraints more commonly use RTP with UDP, instead of TCP, as the underlying transport layer protocol. Within the context of wireless networks, the factors affecting the QoS of this type of traffic include fast handoff, low latency and high bandwidth utilization. The problems associated with Mobile IP, namely inefficient routing, high handoff delay, and that an agent must be present in the network being visited by the mobile host [2], are not conducive to mobile users employing these services.

To overcome these disadvantages, [32] proposes to introduce a *mobility awareness* into the application layer, using SIP as the application layer protocol. SIP already supports personal mobility [29]. A UA registers with the Registrar server each time it obtains a new IP address. The Registrar server will return an IP address for a particular username when requested. This feature of SIP provides a means of contacting a user regardless of their network location.

The changes proposed in [33] would allow SIP to accommodate device mobility also.

49

The two methods used to provide this mobility are third call party control, and the REFER mechanism. The first of these, third party call control [34], allows a UA to transfer a currently established session to a new device. This process is as follows: In
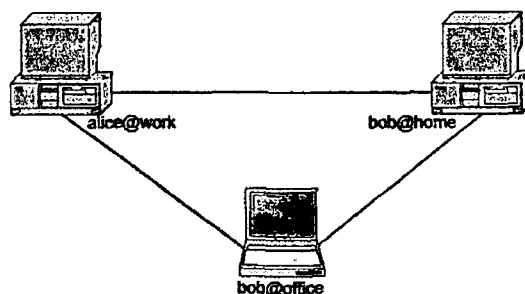

Figure 3.9: SIP Session Transfer

Fig. 3.9 bob@home is currently communicating with alice@work. bob@home wishes to transfer the session to his second device, bob@office. To accomplish this bob@home sends an INVITE to bob@office containing a description of all the session's characteristics. bob@home also sends alice@work the session description generated by bob@office. alice@work transfers session transmission to bob@office, instead of bob@home. This approach has the disadvantage of requiring the continual presence of bob@home so that it may be contacted to change or terminate the session.

The second method involves using the REFER mechanism [35]. Consider the scenario described previously, where bob@home wishes to transfer the session to bob@office (Fig. 3.9). Using this method, bob@home sends alice@work the REFER request, instructing alice@work to contact bob@office. alice@work then negotiates a session with bob@office using a regular INVITE request. This method does not require the presence of bob@home once the session has been established with bob@office.

There are some disadvantages with using SIP to handle user mobility. As with Mobile IP, the SIP device mobility solution must obtain a topologically correct IP address upon entering the new network domain. This new IP address is normally supplied by a DHCP server located in the new network. However, there is a latency

involved in using DHCP to obtain an IP address. According to [36] this delay can be between 10 and 30 seconds. As the envisaged device mobility strategy for SIP doesn't employ soft handover this DHCP-incurred latency could have dire consequences to a multimedia application with strict real-time constraints. It has been proposed by [37] that this latency can be reduced by employing an experimental scheme entitled Dynamic Rapid Configuration Protocol (DRCP).

The SIP device mobility scheme provides a scheme whereby multimedia sessions can be continued across device interchange and network traversal. However, the employment of this solution will sever any existing TCP connections once handover has been performed. Integrating the SIP scheme with Mobile IP would provide a means to perform mobile aware handover and still maintain existing TCP connections, however, this solution would require extensive changes to both the protocol stack at the client and the introduction of several new entities into currently existing networks.

3.4.3 Other Solutions for IP Mobility

The solutions described to accommodate user mobility in IP networks, namely Mobile IP and device mobility in SIP, are not the only ones that exist. Two further solutions that can facilitate IP mobility include Migrate and Host Identity Payload (HIP).

Migrate [38] involves shifting an existing TCP connection from one IP address to another by using a modified TCP SYN message and subsequent TCP ACK message. Migrate proposes that the hostname of a particular Mobile Host remain static and its IP address change as it moves from network to network. Each time a new IP address is obtained the Mobile Host informs the DNS [39] within its home domain of the change. New IP addresses are supplied by the DHCP [40]

HIP [41] involves separating the hostname from the IP address. HIP proposes a globally unique identifier for any host with an IP stack. This name is cryptographic by design and can be used to authenticate transactions. Furthermore HIP proposes to

interpose a protocol layer between the network (IP) layer and the transport layer. This will effectively decouple transport connections from IP addresses.

A further explanation of each of these mobile solutions and a comparison with Mobile IP can be found at [42].

The handover capability of GSM, which is central to providing mobility to GSM subscribers, is handled hierarchically, i.e. depending on the mobile subscriber's location the handover procedure is controlled by the currently-serving BSC, the target BSC, the MSC, etc. This structural handover is not possible within IP networks. Mobility within IP networks is currently the domain of Mobile IP. However, there are disadvantages to employing Mobile IP. These disadvantages include high handover latency and the problem of triangular routing. The delays incurred by Mobile IP can be detrimental to multimedia traffic. To overcome these drawbacks, it has been proposed that SIP could be used to provide mobility to multimedia enabled devices. The SIP approach also has its drawbacks, namely no support for existing TCP connections when performing handover, and high latency in obtaining a new IP address (via DHCP) upon entering a new network domain.

Chapter 4

Proposed Handover Scheme

One of the features of SCTP, its multihoming ability, allows an endpoint to exploit more than one of the IP addresses it has available to it when establishing an association. This functionality, which is not present in TCP or UDP, was designed into SCTP to provide path redundancy to telephony applications. It has been proposed that SCTP multihoming could be used to facilitate mobility at the transport layer, providing a mechanism to handle handover within a single transport connection. However, the current handover scheme in SCTP is based on a static, wireline model of communications. Handover is only performed in the event that the primary path of the association fails. The time taken for handover to occur would prohibit the mechanism being deployed in a wireless context. A more "wireless friendly" handover scheme is needed if SCTP is to be deployed within wireless networks.

4.1   Current SCTP Handover Scheme

In SCTP, for reliable communications to exist between two multihomed endpoints, each endpoint must be aware of the status of each of its peer's destination addresses. If a problem should occur along the primary path of the association, an endpoint needs to be able to detect this and respond appropriately. The responsibility for detecting and

responding to problems lies with SCTP's fault detection and management scheme. In the event that this scheme determines the peer's primary destination address is unreachable, handover to one of the secondary destination addresses needs to be performed.

4.1.1 SCTP Fault Detection

In an SCTP association between two multihomed endpoints all data transferred between the endpoints is transmitted along the primary path. The primary path is the communications link between the primary destination addresses of the endpoints. All the endpoints other destination addresses are marked as secondary addresses. They are used in the event that a fault occurs along the primary path, making the peer's destination address temporarily unavailable. This provides the path redundancy needed by high availability systems.

However, for this high availability to succeed, an endpoint must be aware of the status of all of its peer's destination addresses. As such an endpoint maintains a state, a RTT, a Retransmission Time-Out (RTO) determined from the RTT, and an error counter for each destination address of its peer endpoint. The state of a destination address is either **active**, where the address is reachable, or **inactive** where an endpoint can no longer reach the destination address. This may be due to transient network conditions, such as congestion, or due to some hardware fault along the communications path to the destination address.

In order to maintain the status of its peer's destination addresses, an endpoint periodically polls each of the destination addresses. This polling is performed using SCTP's heartbeat mechanism, which is as follows:

1. Upon sensing that one of its peer's destination addresses has become *idle*, the endpoint generates a HEARTBEAT chunk. A destination address is considered idle if it has not been polled within an interval known as the heartbeat period. This heartbeat period is an implementation set parameter and is usually set to be 30

54

seconds.

2. Within the Heartbeat Sender Specific Information field of the HEARTBEAT chunk, the endpoint places a timestamp in local time, and the destination address that the chunk is to be sent to.

3. The endpoint transmits the HEARTBEAT chunk to the idle destination address and starts a heartbeat timer.

4. Upon reception of the packet containing the HEARTBEAT chunk, the peer endpoint generates a HEARTBEAT-ACK chunk.

5. The peer endpoint copies the Heartbeat Sender Specific Information parameter from the HEARTBEAT into the HEARTBEAT-ACK chunk. This HEARTBEAT-ACK is then transmitted to the destination address contained in the source field of the IP packet that contained the HEARTBEAT chunk.

6. Upon reception of the HEARTBEAT-ACK, the endpoint stops the heartbeat timer, and clears the error counter for the destination address, and the error counter for the association. The destination address is marked as active.

If the heartbeat timer expires, the error counter for the destination address is increased incrementally. If the error counter exceeds the threshold *Path.Max.Retrans* the destination address is marked as inactive and reported to the upper layer. The association error counter is also increased upon expiration of the heartbeat timer. If the association error counter exceeds the threshold *Association.Max.Retrans* then the association is moved into the CLOSED state, and the upper layer is informed of the unreachable state of the peer endpoint.

The heartbeat mechanism outlined above is used to determine the state of a destination address that has become idle. This mechanism forms a part of SCTP's fault

detection strategy. The fault detection algorithm used by SCTP to determine a destination address' reachable state is described as follows:

1. Upon association establishment, set all error counters to be zero, and set state of all destination addresses to be active.

2. Whenever a timer expires on a DATA transmission that was sent to a destination address in the active state, increment the error counter for that destination address. If the error counter exceeds *Path.Max.Retrans*, mark the destination address as inactive and inform the upper layer. Also increment the association error counter. If the association error counter exceeds *Association.Max.Retrans* then move the association into the CLOSED state and inform the upper layer that the peer endpoint has become unreachable.

3. Whenever a HEARTBEAT chunk is sent to an idle destination address, increment the error counter for that destination address. If the error counter exceeds *Path.Max.Retrans*, mark the destination address as inactive and inform the upper layer. Also increment the association error counter. If the association error counter exceeds *Association.Max.Retrans* then move the association into the CLOSED state and inform the upper layer that the peer endpoint has become unreachable.

4. Upon reception of a SACK chunk acknowledging DATA sent previously to a destination address, the error counter for that destination address is cleared. If the destination address is not already set to active, it is marked as such.

5. Upon reception of a HEARTBEAT-ACK in response to a HEARTBEAT chunk sent previously to a destination address that was considered idle, the error counter for the destination address is cleared. The destination address is marked as active.

The error thresholds, *Path.Max.Retrans* and *Association.Max.Retrans*, are configurable parameters and are independent of each other. However some care must be taken

when setting these values as careless configuration can lead to the association entering the dormant state. In this state it is possible for all destination addresses of an endpoint to become inactive and yet the association will still be in the ESTABLISHED state, even though no data may be transmitted. Further explanation of the SCTP's dormant state can be found in [6].

4.1.2 SCTP's Current Handover Procedure

Currently handover in SCTP is said to be failure-centric. Using the methods described in Section 4.1.1, data will only be transmitted to one of the secondary destination addresses if the original primary address becomes inactive. Some failure along the path between the two primary destination addresses has to occur. The process of marking the primary destination address as inactive and performing handover to one of the secondary destination addresses is detailed below.
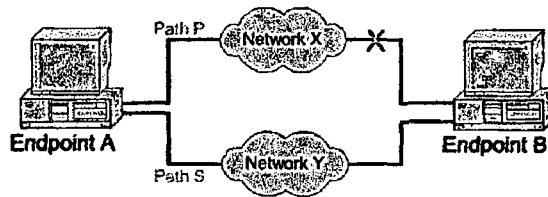

Figure 4.1: An SCTP Association to illustrate SCTP handover

To illustrate handover in SCTP, consider the setup shown in Fig. 4.1. The primary destination address for each endpoint is set by its peer at association establishment. The path between these primary destination addresses is termed the primary path and is denoted as Path P in Fig. 4.1. Under normal operation all data transmitted between the endpoints will travel along Path P.

In order to clearly illustrate the current handover scheme in SCTP, a simplified transmission scheme between the endpoints is assumed. Under this scheme Endpoint A only has data to send. Endpoint B just acknowledges the data it receives from Endpoint A.
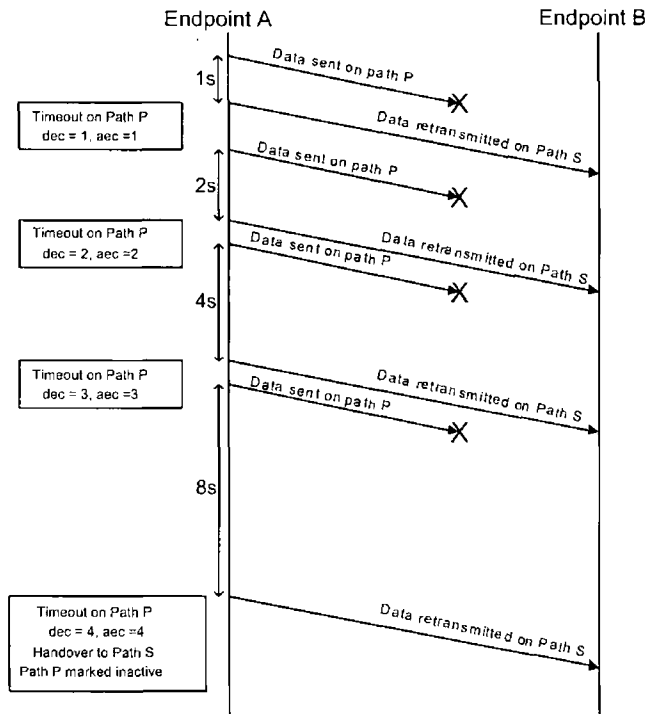
Figure 4.2: Detecting failure of the Primary Path

At some time during the lifetime of the association, Path P fails. The failure of Path P means that the data Endpoint A transmits, at time T_0 in Fig. 4.2, will not reach Endpoint B. This will result in a timeout at Endpoint A. Once this timeout has occurred, Endpoint A will increment an error counter for the destination address. This is shown as *dec* in Fig. 4.2. The error counter for the association, *aec* in Fig. 4.2, is also increased. The retransmission timer, RTO, for the destination address is then doubled and the data is retransmitted along the secondary Path S. The next time that Endpoint A has data to send to Endpoint B it will transmit the data again along Path P. This will result in another timeout on the acknowledgement from Endpoint A's perspective. However, as the RTO has been doubled after the previous timeout, it will take twice the original RTO before the error counters are increased. Once both error counters have been increased, the RTO timer for the destination address is doubled once again, making it four times its original value. The data is retransmitted along Path

S. This continues until an acknowledgement arrives along Path P (in which case all error counters are reset) or the retransmit limit, *Path.Max.Retrans*, is reached. Once this parameter is reached the destination address is marked as inactive, and a new destination address is set to be the primary address. This process is referred to as handover. The new primary destination address is only set temporarily. The polling of all destination addresses continues and if the original primary destination address becomes active again, SCTP will hand back to it.

It can be seen that the setting of *Path.Max.Retrans* is of vital importance when deciding when handover should occur. If *Path.Max.Retrans* is set to too high a value, it will take a long time before handover is performed. This is unacceptable to applications that have strict real-time requirements. If *Path.Max.Retrans* is set too low then the destination address may be marked inactive when that may not be the case, e.g. acknowledgements were delayed due to temporary network congestion. Therefore a compromise must be reached when deciding what value *Path.Max.Retrans* should be set to. The reference implementation of SCTP, supplied with [6], sets this parameter to 4. This would seem a reasonable trade-off.

By the time that this value of *Path.Max.Retrans* is reached and the original primary destination address is marked inactive, the RTO for Path P will have reached eight times its original value. This can be seen in Fig. 4.2. The total amount of time, F_t before a (failed) destination address is marked as inactive is given by:

$$F_t = RTO + 2 \times RTO + 4 \times RTO + 8 \times RTO \tag{4.1}$$

If the initial value of the RTO is 1 second[1] then according to Eq. 4.1 it will take 15 seconds before the failure of Path P is detected and handover to one of the secondary paths, Path S in Fig. 4.1, occurs. This is unacceptable in the case of real time applications, e.g. VoIP. It should be noted that if *Path.Max.Retrans* was set to five, then it

---

[1]The reference implementation of SCTP lower bounds this calculation to be 1 second.

would take 31 seconds[2] before the (failed) primary destination address was marked as inactive and handover to one of the secondary destination addresses occurred.

4.2 New Handover Scheme for SCTP

In a multihomed association, each endpoint must maintain information regarding the status of each of the destination addresses of its peer endpoint. The endpoint obtains this status information through the use of two special control chunks, the HEARTBEAT chunk and its response chunk the HEARTBEAT-ACK. These chunks are sent periodically by an endpoint to poll all destination addresses in the association. The status of the primary destination address is usually obtained using DATA chunks, however, if no data has been transferred between the endpoints for a significant period then the HEARTBEAT and HEARTBEAT-ACK chunks are used to update its status.

The SCTP specification [12] also provides functionality for users to initiate the heartbeating of any of the destination addresses in the current association. SCTP can differentiate between "user" HEARTBEATs, and "system" HEARTBEATs. User HEART-BEATs are sent by the user as part of this on-demand heartbeating mechanism, while system HEARTBEATs have been scheduled previously in order to update status and RTT measurements for a particular destination address. The behaviour of both types is the same, particularly the immediate response of the peer host with a HEARTBEAT-ACK. These user HEARTBEATs form the basis upon which the proposed delay-centric handover scheme is built.

The handover scheme being proposed in this dissertation is based on the scheme developed in [43]. Central to the proposed scheme is the principle that the path with the least amount of delay (ie smallest RTT) should be the primary path. The RTT was

---

[2]If *Path.Max.Retrans* was set to 5,

$$F_t = 1 + 2 + 4 + 8 + 16 = 31$$

chosen as it is well known that it is one of the factors upon which the throughput of a TCP connection is dependent [44]. As SCTP was designed to use very similar congestion control schemes as TCP it can be inferred that SCTP's throughput is similar to TCP's. There is still research being carried out in this area [45].

It should be noted that there are other factors that affect the QoS of multimedia traffic, eg jitter, path loss, etc., and that delay is not the only factor in and of itself. However, the scheme proposes a simplified, yet functional, implementation of a handover scheme using delay as its handover criterion. A more complex scheme could possibly implement some form of path prioritizing based on several factors, including, but not limited to, delay, jitter, cost, etc. The proposed scheme determines which path has the least delay by heartbeating all the destination addresses of the association and computing the corresponding RTTs using the timestamps contained in the HEARTBEAT-ACK chunks. It then compares all the recently computed RTTs and determines the smallest one. It sets the destination address that corresponds to this RTT to be the primary path.

4.3   Reference Implementation Code

The reference guide to SCTP [6] is accompanied by a user space implementation of the new protocol. This implementation is open source. The code is written in the C programming language and will compile on any UNIX-like operating system (Linux, FreeBSD, NetBSD, Lynx O/S, Solaris).

To illustrate the organization of the reference code, the details of establishing an association are described with reference made to the relevant functions that are used. For information on any of SCTP's other functions, such as association termination, see [6]. The reference implementation code is also well commented.

4.3.1 Association Setup

The process of establishing an association between two SCTP endpoints was described in Section 2.3, and involves a four way message exchange. An endpoint, under the instruction of its ULP, will begin this exchange by transmitting the first of these messages, the INIT chunk. An association may also be established, if an endpoint receives a valid INIT chunk from a peer endpoint. As this section aims to illustrate the functionality of the reference implementation code, only the former of these establishment scenarios will be described, i.e. the endpoint being described is the sender of the INIT chunk.

Sending the INIT

The user first sets the peer destination address and port number, via the *sethost* and *setport* upper layer commands. Next the user will issue the *assoc* command to begin establishing the association. This upper layer command is a wrapper to the *sctpAS-SOCIATE()* command. An association may also be established by the user calling the upper layer command *send* with correct arguments. The *send* command is a wrapper to the *sctpSEND()* command.

In the first of these scenarios, after the *sctpASSOCIATE()* function is called, it will call *SCTPstartInit()*. This module allocates an association Transmission Control Block (TCB) with a call to *SCTPalocAssociation()*. It then generates and transmits the INIT (by calling the *SCTPsendInitiate()* function). Next it sets up the TCB's internal state, and starts a timer for the INIT chunk via a call to *timerWork()*. Finally it returns the newly created association.

If the ULP calls the *sctpSEND()* instead of *sctpASSOCIATE()*, the process of allocating the TCB and transmitting the INIT chunk is the same as above. However, before this process begins, a check is made that the data to be sent once the association is established, is being transmitted on Stream 0. Stream 0 is the only stream guaranteed to be present once the association is established. If the ULP is trying to transmit to

any stream other than Stream 0, *sctpSEND()* will return an error. If the ULP is transmitting on Stream 0, a call to *sendToStream0()* is made. This function will try to find the association, via the *SCTPfindAssociation()* function. If the association is not found then *SCTPstartInit()* is called and the process is the same as described above for the *sctpASSOCIATE()* case.

The user data that *sctpSEND()* was called with is queued by the association TCB returned by *SCTPstartInit()*. This ensures that the data is piggybacked with the COOKIE-ECHO, which is transmitted after a valid INIT-ACK has been successfully received.

Receiving the INIT-ACK

After the INIT has been transmitted, the endpoint will be in the COOKIE_WAIT state. In this state one of two things will happen. Either the timer it started after sending the INIT chunk will expire, or the endpoint will receive an INIT-ACK from its peer. In the case of timer expiration, the *sctptimerExpires()* function is called. This function determines what type of timer has expired and subsequently calls the relevant function to handle this. In this instance it is an INIT timer that has expired so *SCTPhandleInitTimerUp()* is called. *SCTPhandleInitTimerUp()* checks if the maximum number of retransmissions has been reached. If it has been reached then the association TCB is removed (by a call to the *SCTPfreeAssociation()* function) and the ULP is informed of the situation.

If the maximum number of retransmissions has not been reached, then *SCTPhandleInitTimerUp()* calls *SCTPsendInitiate()* again. This will retransmit another INIT chunk to the peer. As before, after the INIT has been sent a timer is started. The endpoint remains in the COOKIE_WAIT state, waiting for an INIT-ACK from its peer, or the INIT timer to expire.

If the endpoint does receive an INIT-ACK it will be read by the *sctpfdEvent()* function. This function then calls *SCTPprocessInbound()* which processes all incoming SCTP packets. This is the normal procedure for receiving SCTP packets. Once

63

detected by *sctpfdEvent()*, *SCTPprocessInbound()* is called. This function then calls *SCTP_handleControlPortion()* to handle any control chunks located in the packet. As control chunks are always placed at the beginning of SCTP packets, *SCTP_handleControlPortion()* will continue processing chunks until one of the following occurs:

1. it reaches the end of the SCTP packet

2. it encounters a chunk that is required to be transmitted in its own packet (e.g. INIT, INIT-ACK, HEARTBEAT, etc.)

3. it encounters a DATA chunk

In this case the SCTP packet should only contain the INIT-ACK chunk. If any other chunks have been placed in the packet with the INIT-ACK they will be ignored according to 2 above. Once the *SCTP_handleControlPortion()* encounters the INIT-ACK it will call *SCTP_handleInitiateAck()*. The *SCTP_handleInitiateAck()* function will then try to find the association the INIT-ACK belongs to, if it was not found by *SCTP_handleControlPortion()* previously. It does this by calling *SCTPspecialFindAssociation()*. This function looks inside the INIT-ACK to try to determine what association the INIT-ACK is a part of.[3]

SCTPspecialFindAssociation() performs an association lookup based on each destination address it finds in the INIT-ACK. If no association is found the function returns a NULL. If an association is found, a series of checks on the packet are performed next. These checks involve validating the Verification Tag, checking the peer's Initial Tag is not zero, making sure the peer has met the minimum *a_rwnd* requirement, that there is at least one stream allowed by the peer's Maximum Inbound Streams (MIS) value, and verifying that the endpoint is in the correct state to receive an INIT-ACK (the

---

[3]It should be noted that neither endpoint is in the ESTABLISHED state at this stage and the association is not fully setup. The association that these functions are trying to locate is the one currently being established.

COOKIE-WAIT state). If all these checks are correct, then the INIT-ACK is adopted. *SCTPadoptThisInitAck()* performs this adoption, and updates the association TCB values with those found in the INIT-ACK. One of the first things *SCTPadoptThisInitAck()* does when called is to extract the Cookie from the INIT-ACK. This Cookie is then placed in a special cookie pointer. This can be accessed by *SCTPsendAnyWeCan()* when *SCTPprocessInbound()* returns. *SCTPsendAnyWeCan()* will transmit the Cookie, in a COOKIE-ECHO chunk, back to the peer endpoint. *SCTPadoptThisInitAck()* calls *SCTPadoptThisInit()* to build all appropriate structures within the association TCB.

Sending the COOKIE-ECHO

As stated previously the Cookie is extracted from the INIT-ACK so it may be placed in a COOKIE-ECHO chunk once the processing of the INIT-ACK has completed. Once the COOKIE-ECHO is transmitted the endpoint starts a timer, via a call to *timerWork()*. If this timer expires the procedure for handling its expiration is similar to the procedure for handling the init timer expiration described previously, with the exception of *SCTPhandleCookieTimerUp()* being called in place of *SCTPhandleInitTimerUp()*. *SCTPhandleCookieTimerUp()* increments the overall error counter, checks to see if the maximum retransmission threshold has been exceeded. If it has, then the ULP is notified and the association is removed. If the maximum retransmission threshold has not been exceeded, then *SCTPsendRetransmits()* is called to retransmit the COOKIE-ECHO. The COOKIE-ECHO is retransmitted in a packet on its own. It was a design decision not to retransmit any user data that may have been piggybacked on the previous COOKIE-ECHO transmission.

Receiving the COOKIE-ACK

If the endpoint receives the COOKIE-ACK, it will be read in the same way as the INIT-ACK, leading to the packet being processed by the *SCTP_handleControlPortion()* function. Once the SCTP packet's Verification Tag has been validated and the COOKIE-ACK processed the association is moved into the ESTABLISHED state and the ULP is notified. The association is now completely set up and the transfer of user data can begin.

4.4 Proposed Handover Scheme Implementation

Central to the proposed handover algorithm outlined in Section 4.2, is the ability to perform on-demand heartbeating. That is, the application layer can transmit a HEARTBEAT to any destination address in the current association at intervals other than that specified by the heartbeat period.

The reference implementation of SCTP, supplied with [6], allows for on-demand heartbeating by differentiating between "system" HEARTBEATs and "user" HEARTBEATs, or more precisely the implementation can distinguish between a response to a HEARTBEAT transmitted after a heartbeat period has expired, and the HEARTBEAT-ACK in response to a HEARTBEAT sent under instruction of the ULP. The proposed scheme incorporates this differentiating ability of the reference implementation in providing a more efficient handover mechanism.

As well as the addition of several new functions one of the reference implementation's current functions had to be altered. In the standard reference implementation code, once a HEARTBEAT-ACK has been detected by *sctpfdEvent()*, and processed by *SCTPprocessInbound()* and *SCTP_handleControlPortion()*, it is passed to the function *SCTPProcHBResp()*. This function then determines if the HEARTBEAT-ACK is in re-

sponse to a user HEARTBEAT or a system HEARTBEAT. In the former case, the ULP is notified and the HEARTBEAT processing continues as normal, updating the RTO for the destination address based on the timestamp contained in the HEARTBEAT-ACK.

In order to facilitate the correct operation of the proposed handover scheme it was necessary to adjust the handling of user HEARTBEATs so that the proposed functionality will be used. This was achieved by making a small modification to the *SCTP-ProcHBResp()* function. Instead of simply informing the ULP that a user HEARTBEAT has arrived, the code was modified to call a function, *andyRTT()* as part of the proposed handover scheme.

The new handover process is started from within the user interface. This is a program that acts as a ULP. It allows a user, amongst other things, to specify the peer destination address to which the INIT chunk should be sent. This is then set as the initial primary destination address. The heartbeating begins when the user calls the *assoc* command within the user interface. This command begins the normal association procedure, as described in Section 4.3.1, and then calls the *andysendHB()* and a user HEARTBEAT is sent. The peer endpoint will respond with a HEARTBEAT-ACK.

Once it has been determined that the HEARTBEAT-ACK that has been received is in response to a user HEARTBEAT, *andyRTT()* is called with the timestamp, extracted from the HEARTBEAT-ACK, and a reference to the destination address the HEARTBEAT-ACK was transmitted from. The *andyRTT()* function first determines how many destination addresses there are in the association. Next it calculates the RTT for the destination address that the HEARTBEAT-ACK was received from, via a call to *andyCALCRTT()*, passing to it the timestamp contained within the HEARTBEAT-ACK. This RTT value is stored in an integer array. The array is defined using the **static** keyword so that the RTT value will not be lost when *andyRTT()* returns.

Once the RTT has been calculated the *andyRTT()* function next checks if all destination addresses in the association have been heartbeated. If they have not then a

HEARTBEAT is sent to the next destination address by calling *andysendHB()* with the index to the relevant destination address. If all destination addresses have been polled then a handover decision is to be made. This is accomplished by first determining the smallest of the calculated RTTs, using a simple *for* loop. Once this has been found, the corresponding destination address is compared to the current primary destination address, obtained by calling *sctpGETPRIMARY()*, to determine if the current primary path is the one with the smallest RTT. If this is the case no handover needs to occur, and *andyRTT()* returns. If the destination address associated with the smallest RTT is not the current primary destination address then it is set to be the new primary destination address via a call to *sctpSETPRIMARY()*. Finally, a timer is invoked to begin the heartbeating process again 1 second later.

The current handover scheme employed by SCTP provides a mechanism for multihomed endpoints to continue communicating even in the presence of primary path failure. This feature of SCTP is not available in either TCP or UDP, and provides path redundancy to applications availing of it. The algorithm used to detect the failure of the primary path has been designed to be deployed within a wired environment. As such, the handover mechanism used in SCTP whilst being functional is quite conservative, taking of the order of fifteen seconds to handover once the primary path has failed.

The delay in performing handover imposed by the current handover scheme would limit its effectiveness when deployed within a wireless environment such as WLAN. For SCTP applications that are to be implemented within such networks, a new strategy for performing handover is needed. The handover scheme proposed by this dissertation involves more frequent polling of the destination addresses in the association to gather information regarding RTTs to each of these destination addresses. Having gathered all these RTTs the scheme performs handover to the destination address with the smallest RTT. This proposed scheme does not incur the four timeout penalty associated with

the current handover scheme in SCTP.

Chapter 5

Simulation

In order to evaluate the handover scheme proposed by this dissertation, the scheme was implemented within the simulation environment. Within this context, the stated advantage of reduced handover latency was investigated. The tool chosen to perform this evaluation was the Network Simulator, ns-2. This tool was designed to specifically simulate networking scenarios. Its modular design allows for the introduction of new protocols, such as SCTP.

The proposed scheme was initially evaluated in isolation. An association was established between two multihomed SCTP endpoints. The delay on one of the links between the endpoints was varied and the behaviour of the traffic transferred between the endpoints was observed. The scheme was next evaluated within the presence of competing background traffic. The effect of the contention delay on the SCTP packets due to the background traffic was examined.

5.1 ns-2 Overview

Using the REAL simulator as a basis, the development of the ns simulator began in 1989, and progressed through to the current version (Version 2) thanks to a DARPA research initiative, and the contributions of fellow ns-2 users. ns-2 is a discrete event

simulator. It provides support for various networking technologies, such as various *flavours* of TCP (Reno, Tahoe, SACK) [11], UDP, and numerous unicast and multicast routing protocols. The scope of ns-2 ranges from the provision of a testbed for the evaluation of adaptive streaming schemes for Video On Demand (VOD) [46] to the examination of QoS strategies in IEEE 802.11 wireless systems [47].

ns-2 was designed to be object-oriented. It was written in both C++ and OTcl. Two languages were chosen due to speed and configurability constraints. The code written in C++ comprised the protocol implementations, such as the class definition for TCP, UDP, etc. C++ was chosen as it was more efficient at manipulating bytes, and running algorithms over large sets of data. It also provided faster execution time compared to OTcl. However, changes made to the C++ configuration required recompilation of the code. This greatly increased the time taken to run simulations, especially where only a small number of parameters needed to be changed from one simulation run to the next. For this purpose the simulation scripts were written in OTcl. OTcl, the object-oriented version of the Tool Command Language, Tcl, was slower in execution than C++. However, scripts written in OTcl could be reconfigured quickly without needing to be recompiled.

Due to being implemented in two languages, the concept of linkage was central to ns-2. Linkage provided a direct correspondence between C++ classes and classes defined within the OTcl environment. Variables declared in C++ classes could be accessed from OTcl scripts by means of a binding function. This allowed for OTcl scripts to set object parameters, such as the number of streams that an SCTP endpoint advertises when trying to establish an association.

5.2 SCTP Module for ns-2

The SCTP code used in ns-2 was developed by the Protocol Engineering Laboratory at the University of Delaware. It was implemented in the form of an ns-2 module[1]. This module contained most of the functionality of SCTP as defined in [12]. This included the ability to establish an association by means of the four way handshaking mechanism, though no Cookie was generated. The functionality the module provided also included the transmission and acknowledgement of DATA chunks, and support for multihomed endpoints. SCTP's multistreaming concept was also supported via the provision of Stream Identifiers and SSNs. Endpoint failure may also be detected by the module. The module also supported heartbeating, though it didn't support on-demand heartbeating. This functionality was added in order to evaluate the proposed handover scheme through simulation.

The SCTP module for ns-2 also provided a sample application. This application exploited some of the SCTP module's features, such as multihoming. The sample application was modified to suit the needs of the simulation models used in the evaluation of the proposed handover scheme.

Much of the research into SCTP was conducted through simulation, utilizing ns-2 and the SCTP module developed by the University of Delaware. Significant research has been conducted in the area of implementing SIP using SCTP as the underlying transport layer [48, 49, 50]. Other areas of SCTP research include using ns-2 to explain the multistreaming feature of SCTP[51] and the development of a scheme for prioritizing streams within an SCTP association [15]. The effects of SCTP's congestion control on traffic transmitted over satellite links [52], and in a multi-hop wireless environment [53] were also determined using the ns-2 simulator.

The scheme outlined in [54] described how to perform handover from a UMTS connection to a WLAN connection. It outlined the architecture that should be used in

---

[1]available from http://pel.cis.udel.edu/

what the scheme termed Mobile SCTP (M-SCTP). Central to this scheme was the exploitation of SCTP's multihoming feature coupled with the Dynamic Address Reconfiguration (DAR) extension defined in [55] (see Section 7.1.1). The scheme negated the need for a solution such as Mobile IP, and avoided the triangular routing problem associated with such a solution. The scheme did not specify the criteria upon which the handover was based, merely that handover occurred once the presence of a WLAN cell was detected.

5.3 A Simple Model

In order to evaluate the proposed handover scheme it was necessary to make modifications to the ns-2 SCTP module that was developed by the Protocol Engineering Laboratory. The modifications included the addition of an on-demand heartbeating process, and the introduction of "user HEARTBEATs". These adjustments were tested using the model shown in Fig. 5.1. In the model, a multihomed endpoint was represented by a core node (Nodes 0 and 3 in Fig. 5.1) and several interface nodes (Nodes 1 and 2 and Nodes 4 and 5).
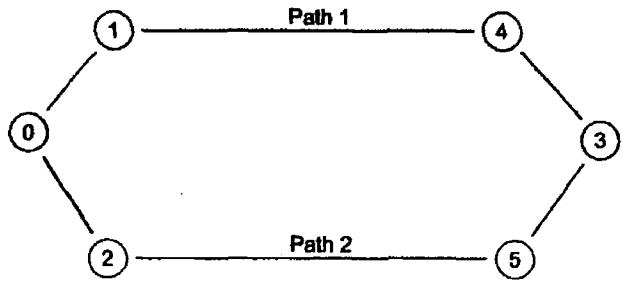
Figure 5.1: SCTP multihomed simulation model

Once the association between the two multihomed nodes was established, the periodic heartbeating process was initiated. After 4 seconds simulation time the delay on Path 1, the initial primary path, was increased from its initial level of 15ms to a delay of 50ms for a duration of 3 seconds, after which it returned to its previous delay level. The delay on Path 2, the secondary path, was set to be a constant 30ms for the duration of the simulation. These delay patterns can be seen in Fig. 5.2. Fig. 5.3 shows a trace of the SCTP packets (TSNs) exchanged between the two endpoints during the simulation. Correlating this trace with the delays shown in Fig. 5.2, it can be seen that the on-demand heartbeating process is functioning as prescribed by the handover scheme being proposed.

Upon closer examination of both Fig. 5.2 and Fig. 5.3, there appears to be a slight lag between the path delay on the initial primary path increasing and SCTP performing handover. This lag is approximately 1 second. The reason for the lag is due to the nature of the heartbeating process, as each destination address is polled periodically, not continuously, so that changes in network conditions take time to be determined and processed.
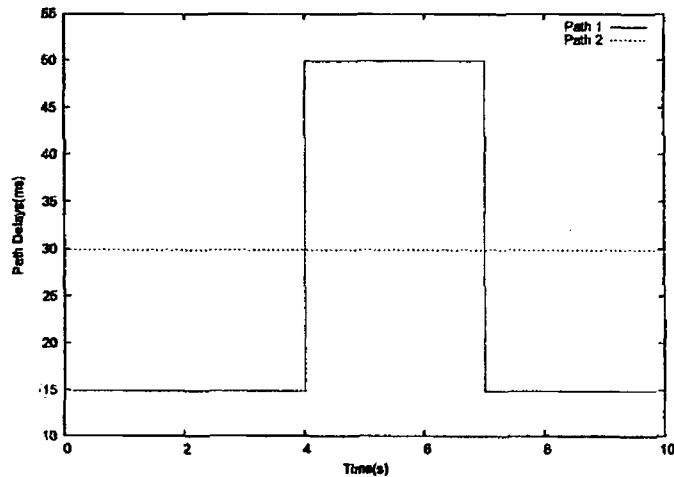

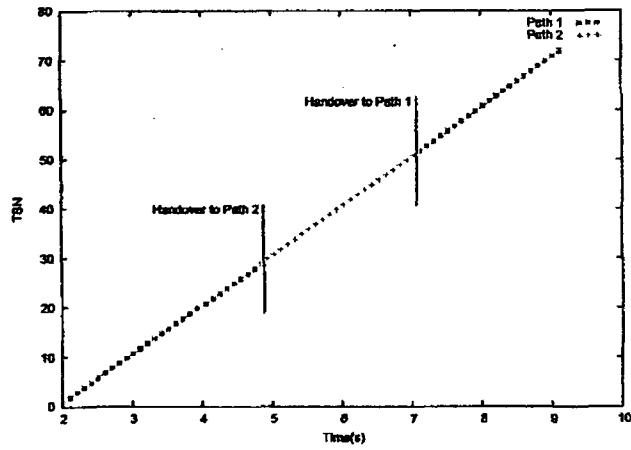Figure 5.2: Delays implemented in simple SCTP handover model

Figure 5.3: TSN trace of simple SCTP handover model

5.4 Further Simulation Scenarios

5.4.1 SCTP and TCP

The model described in Section 5.3 examined the performance of the proposed handover scheme in the presence of fluctuating path delays. This simple model was devised to determine that the handover scheme was functioning correctly before more complex scenarios were simulated.
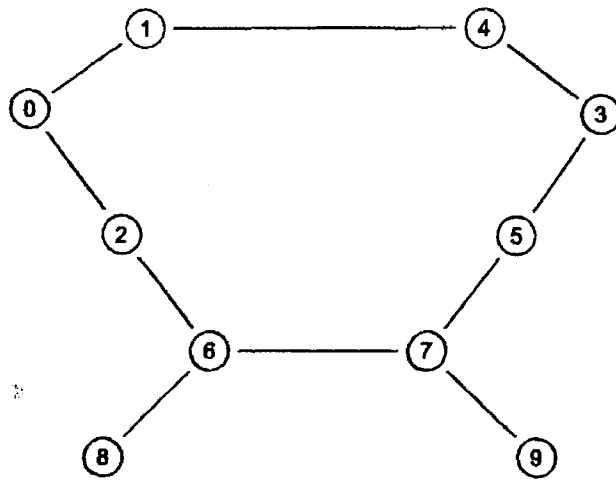

Figure 5.4: Network Topology

The first of these scenarios involved observing the effect that a competing TCP flow

had on the delay experienced by SCTP packets over a shared link. The previous model was altered to accommodate the introduction of two TCP nodes and the shared link. This is shown in Fig. 5.4. Node 8 generated FTP traffic which it transmitted to Node 9 via the link between Nodes 6 and 7. The TCP traffic used was FTP traffic with a packet size of 1000 bytes. The SCTP traffic pattern employed was similar to that used in [56].

Each link had a bandwidth of 10Mb and an associated delay of 10ms except the link between Nodes 1 and 4 and the link between Nodes 6 and 7. The link between Nodes 1 and 4 was the initial secondary path, and had a bandwidth of 10Mb and an associated delay of 90ms. The bandwidth of the link between Nodes 6 and 7 was 0.7Mb, the bottleneck link, with an associated delay of 60ms. The initial primary path, Path 1, was between Nodes 2 and 5, incorporating the bottleneck link.

The SCTP association was established after 1 second and the heartbeating process was started after 1.5 seconds. The TCP traffic was introduced 3 seconds into the simulation run and was maintained for a duration of 4 seconds. The delays recorded for each path are shown in Fig. 5.5.

It can be seen in Fig. 5.5 that the TCP traffic caused an increase in the round trip delay experienced by SCTP packets travelling along the bottleneck link. This increased delay, measured at 4.6 seconds exceeded the round trip delay experienced by the HEARTBEAT packets on the secondary path, Path 2. Handover to Path 2 was performed at this point. This is indicated in the SCTP packet trace shown in Fig. 5.6. Once the TCP-induced congestion had passed, after 7.3 seconds, the measured round trip delay of Path 1 was lower than that of Path 2 and handover was performed once again, this time back to Path 1. This is also indicated in Fig. 5.6.
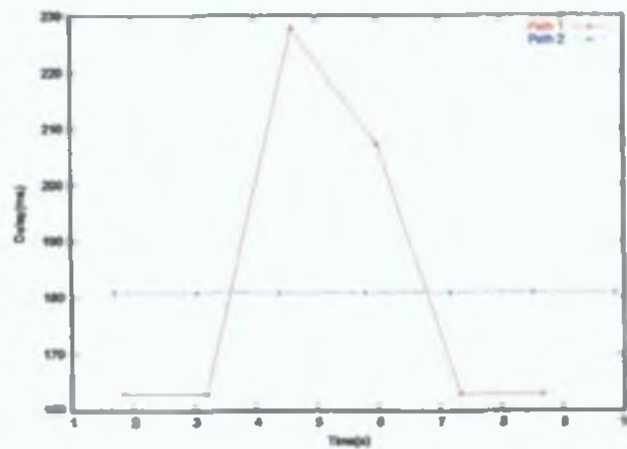
Figure 5.5: Delays experienced by SCTP HEARTBEATs in the presence of TCP traffic
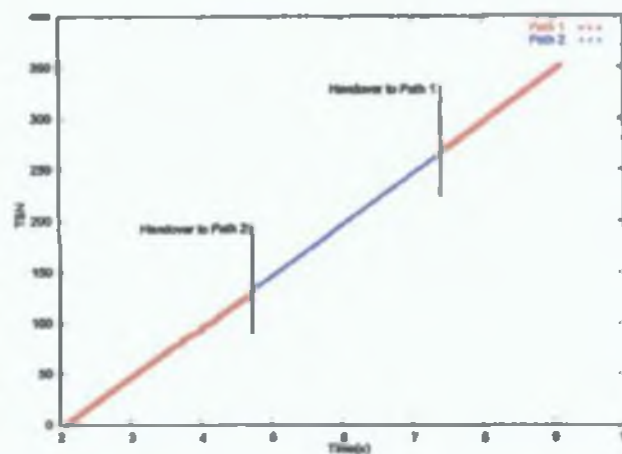

Figure 5.6: TSN trace of SCTP DATA chunks (TCP background traffic)

5.4.2 SCTP and UDP

This scenario involved observing how constant bitrate UDP traffic affected the delays experienced by SCTP packets over a shared link. The topology used was the same as shown in Fig. 5.4. The UDP traffic was CBR type with a packet size of 1000 Bytes and a transmission rate of 680kb/s and was transferred from Node 8 to Node 9 via the bottleneck link. The various link bandwidths and associated delays were the same as those for the TCP scenario.

The SCTP association was established after 1 second. The heartbeating process was started after 1.5 seconds. Node 8 began transmitting UDP traffic after 3 seconds for a duration of 4 seconds. The delays recorded by the SCTP HEARTBEATs are illustrated in Fig. 5.7. The delay on Path 1 began to increase after 3.2 seconds. This was due to the introduction of the UDP traffic. After 4.6 seconds the round trip delay experienced by SCTP packets on Path 1 had exceeded the round trip delay of Path 2. Handover to Path 2 was performed at this point. This is illustrated in the SCTP packet trace shown in Fig. 5.8. Once the congestion incurred as a result of the UDP traffic had passed, at 7.3 seconds in Fig. 5.7, handover back to Path 1 was performed. This is also shown in the TSN trace of Fig. 5.8.
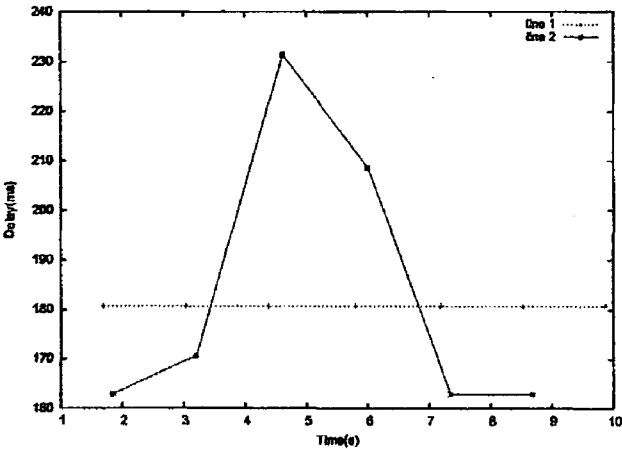

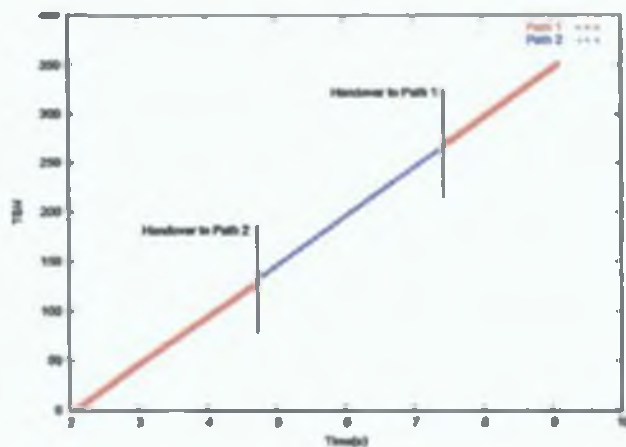Figure 5.7: Delays experienced by SCTP HEARTBEATs in the presence of UDP traffic

Figure 5.8: TSN trace of SCTP DATA chunks (UDP background traffic)

5.5 Summary

The proposed handover scheme, designed to address the deficiencies of the current
handover strategy employed in SCTP, was evaluated within a simulation environment.
The network simulator, ns-2, was selected as a module that contained most of the
SCTP functionality described in [12] had already been developed. This module pro-
vided mechanisms for establishing and monitoring an association, however, it did not
provide support for on-demand heartbeating. This process is central to the proposed
handover scheme and had to be developed so that the proposed scheme could be imple-
mented. The result of introducing this functionality to the existing SCTP simulation
code has been examined through the use of a simple model consisting of two multi-
homed SCTP endpoints. These endpoints were connected by two links. To determine
that the on-demand heartbeating, and thus the proposed handover scheme, was func-
tioning correctly the delay on one of the links was increased and the behaviour of the
association was observed. As expected handover was performed from the link where
delay had been increased to the second link where delay had not been altered. The
delay of the first link was returned to its original delay value and the association was
observed to perform handover once again, back to the first link, demonstrating that the
proposed scheme was indeed functioning correctly.

Next the effects competing traffic had on the delays recorded by the proposed scheme was modelled. First, a TCP flow was introduced. This flow shared a common link with the SCTP traffic. It was observed that as the TCP traffic was transferred across the shared link, the delay experienced by SCTP HEARTBEATs along the same link increased. This, obviously, was to be expected. The increase in the delay experienced by the SCTP packets caused handover to be performed from this congested link to the secondary path of fixed delay. Once the TCP traffic had finished being transmitted handover was once again performed. This time back to the original, shared link. A similar approach was taken to examine the effects of UDP traffic on the round trip delay recorded by the SCTP HEARTBEATs. In this scenario the delay began increasing more quickly than in the scenario involving TCP traffic. This was to be expected as the UDP traffic was constant bit rate and the TCP connection began in "slow start" phase. Handover was performed when the delay on the shared link increased above the delay on the secondary path, due to the presence of the UDP traffic. Once the UDP traffic had abated, handover was once again performed back to the original path, the shared link. Both of these scenarios illustrate that the proposed handover scheme functioned correctly, performing handover in the presence of high round trip delay, and handing back once the congestion that induced the original handover had passed.

Chapter 6

Experimental Results

The SCTP handover scheme proposed by this dissertation was initially evaluated within a simulation environment. The ns-2 simulator was used to evaluate the efficacy of the proposed scheme. However, the simulation environment is a somewhat artificial context in which to evaluate modifications to a network protocol such as SCTP. The next stage in the evaluation of the proposed handover scheme, therefore, was to observe how the scheme behaved when deployed within a "real" environment.

6.1   Network Emulation

The first step was to deploy the scheme within a "controlled" real network. This term is somewhat contradictory, as a real network environment is practically impossible to control. In order to achieve some level of control the NIST Net emulation tool was employed. Following on from emulation the proposed scheme's behaviour was determined within a wireless context. In this instance, an SCTP association was established across two WLAN cells, and the amount of background traffic in one cell was increased. This had the effect of increasing the congestion, and subsequently the delay, experienced by the SCTP packets in that cell and caused the proposed scheme to perform handover. Finally the effect of mobility of one endpoint on the proposed handover scheme is exam-

ined. In particular how the proposed handover scheme contends with lost HEARTBEATs on the primary path.

6.1.1 NIST Net Overview

Due to the cost associated with deploying services within an environment such as the Internet, service providers needed a guarantee that the services when deployed would function as envisioned. In order to accomplish this extensive testing was necessary prior to deployment. However, services designed to be deployed across the Internet were complex in design. Their behaviour could not be extrapolated from an examination of their codebase. Factors such as how the service would cooperate with already deployed services, how the service would react to varying network conditions, such as congestion and bandwidth limitations, needed to be assessed. Only an approximation of the effect these factors, as well as the effects of jitter, delay, and loss, had on the QoS of the service could be determined from simulation.

The NIST Net tool[1] was developed to provide an environment that could emulate the characteristics of a "live" network. NIST Net's designers termed this environment "semi-synthetic" as it provided real-world network characteristics which were user-controlled and were easily reproduced. NIST Net can be installed on any PC running the GNU/Linux operating system. It is implemented as a kernel module[2] and a user interface. This user interface is provided as both a Graphical User Interface (GUI) and a command line interface. The latter is more useful when setting NIST Net entries via shell scripts. The NIST Net GUI is shown in Fig. 6.1.

NIST Net operates on packets at the IP layer according to user-specified network characteristics. The set of characteristics that the NIST Net user has available for manipulation include packet delay, jitter, random and congestion-dependant packet drop-

---

[1] available from National Institute of Standards and Technology (NIST), http://snad.ncsl.nist.gov/itg/nistnet/

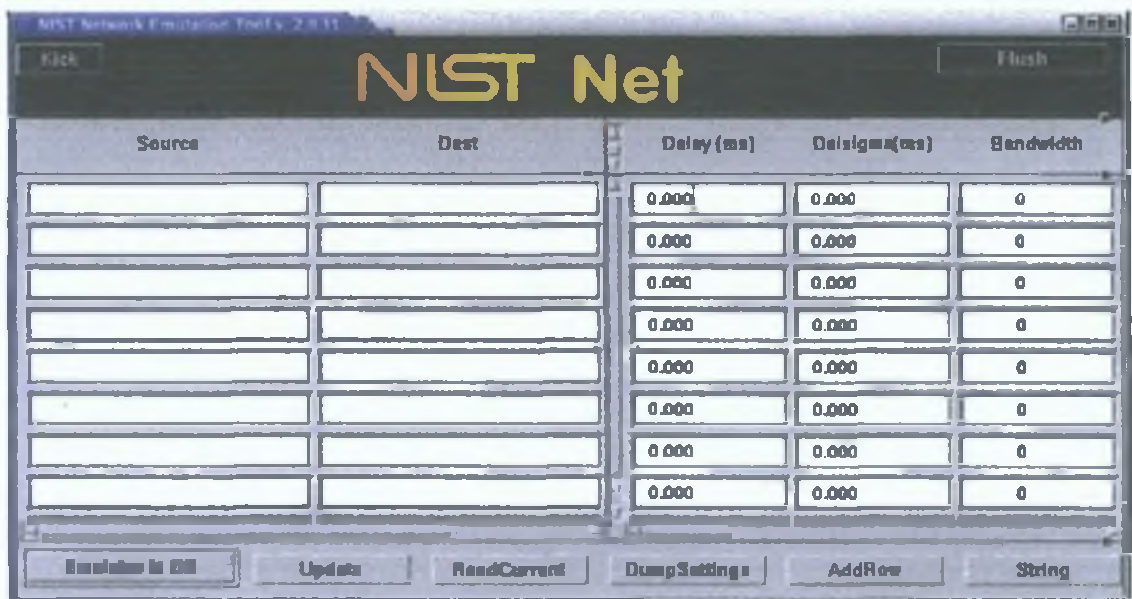[2] The NIST Net module will only work with Linux kernel 2.0.x and above

Figure 6.1: NIST Net Graphical User Interface

ping, bandwidth limitations, packet reordering, and packet duplication. The network characteristics specified by the user are stored in NIST Net emulator entries. These entries are composed of three parts:

1. a set of specifications that are used to match packets that correspond to this particular entry (e.g Source/Destination IP address)

2. a set of specifications that are to be applied to matched packets (e.g. a delay of 300ms)

3. a set of statistics regarding packets that have been matched against a particular entry

Each emulator entry is completely separate from all other entries. NIST Net can support thousands of these entries. Entries can be added or changed manually, or via shell scripts, during the course of NIST Net operation[3].

[3]The operation of NIST Net is discussed in further detail in "NIST Net - A Linux-based Network Emulation Tool", available from http://snad.ncsl.nist.gov/itg/nistnet/nistnet.pdf

6.1.2 NIST Net Results

The testbed shown in Fig. 6.2 was used to evaluate the proposed SCTP handover scheme. The testbed consisted of two multihomed SCTP endpoints connected through a PC running the NIST Net emulation tool. One of the SCTP endpoints, Endpoint A, was modified to include the functionality described in the proposed scheme. Endpoint B was unchanged. The delays that were implemented in NIST Net are shown in Fig. 6.3.
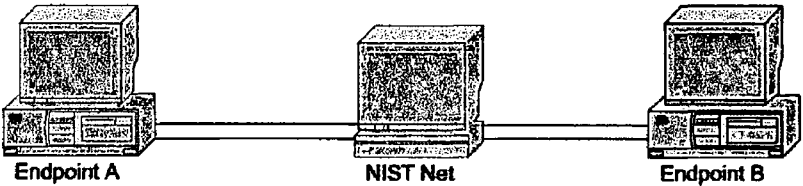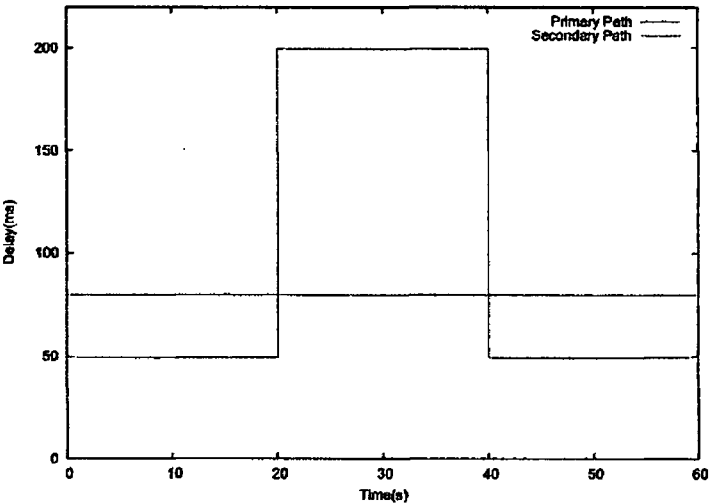

Figure 6.2: NIST Net testbed

Figure 6.3: Delays implemented in NIST Net

After 20 seconds the delay experienced by packets travelling on the initial primary path ("Primary Path" in Fig. 6.3) was increased to 200ms. This level of delay on the initial primary path was sustained for a further 20 seconds before decreasing to its former value of 50ms. The effect this increase in path delay had on the SCTP traffic being transmitted from Endpoint A to Endpoint B can be seen in the trace of TSNs shown in Fig. 6.4.
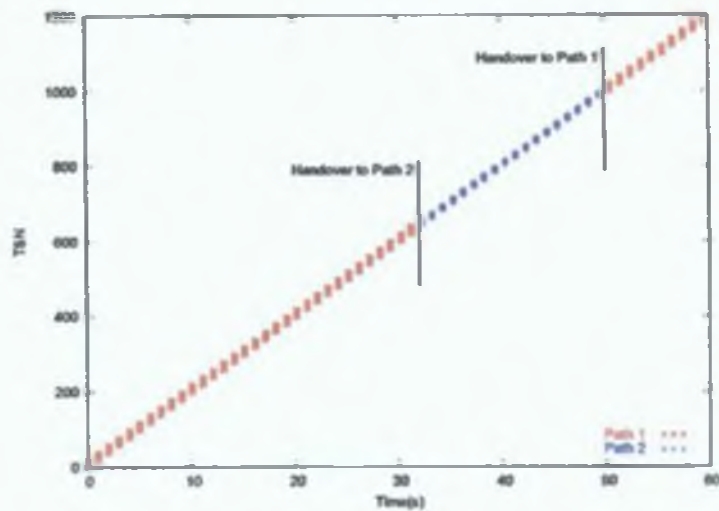
Figure 6.4: SCTP TSN trace using NIST Net

It can be seen from Fig. 6.4 that handover occurs from the initial primary path to the other path of the association ("Secondary Path" in Fig. 6.3) after approximately 32 seconds. It can also be seen that handover occurs once again, back to the initial primary path, after approximately 50 seconds. The reason that there is a 12 second lag between the change in path delay and the first handover occurring is that the handover decision is not based on just one set of measurements. Several sets of RTTs are gathered, and from these an average RTT for each path is determined. It is these average values that are used as the basis of the handover decision. The lag was large in this case because a handover decision had been made just prior to the primary path delay increasing. The explanation is similar for the lag in handing back to the initial primary path after 50 seconds. The delays recorded by the HEARTBEATs can be seen in Fig. 6.5.
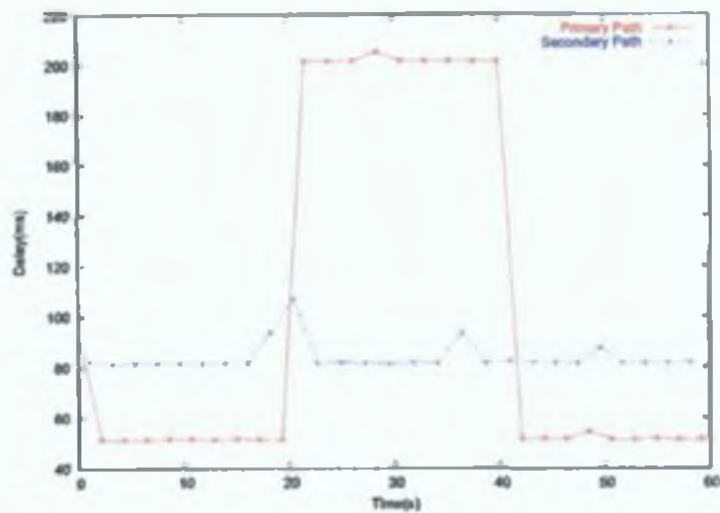
Figure 6.5: Delays recorded by HEARTBEATS

6.2 WLAN Results
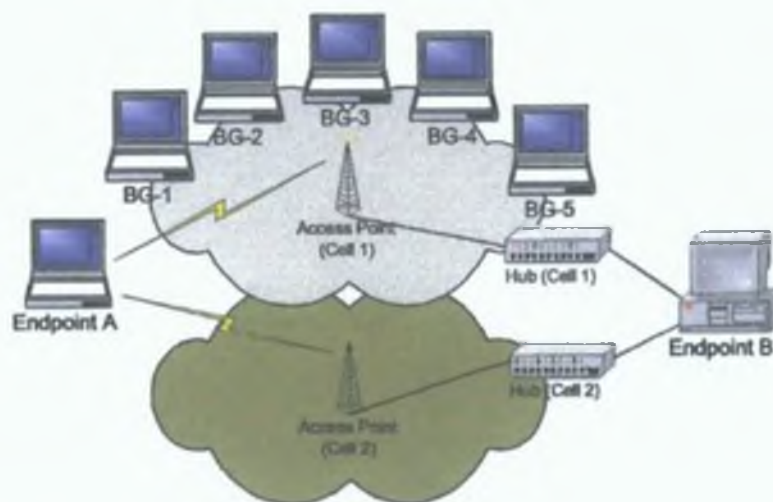
6.2.1 Effects of Congestion

Figure 6.6: WLAN Equipment Setup

The setup used to evaluate the proposed handover scheme is shown in Fig. 6.6. The setup included two multihomed SCTP endpoints running the GNU/Linux operating system. The implementation of SCTP installed on each endpoint was the reference implementation of SCTP available from [G]. Modifications were made to the implemen-

tation installed on Endpoint A to accommodate the proposed handover scheme. The association between the two endpoints incorporated the links through both Cell 1 and Cell 2. The initial primary path, Path 1, was set to be the path through Cell 1. The secondary path, Path 2, was the path through Cell 2.

As the proposed scheme was to be tested within a wireless environment, such as WLAN, some modifications were made to the scheme to address the deficiencies of such an environment. These deficiencies include sporadic delay spikes, and the "ping-pong" problem. In order to reduce the effect delay spikes had on handover, an average of the delay measurements was used when making the handover decision, as mentioned in Section 6.1.2. The average delay measurement was determined from three values of round trip delay for each path. The value of three was chosen as a trade-off between minimizing the effect of delay spikes and not introducing too great a latency in performing the actual handover.

Within cellular networks, there exists the problem of excessive handoffs when a user is located between the boundaries of two cells. The continual handing off between the two basestations is known as the "ping-pong" effect. As each handoff incurs an amount of network overhead, this problem results in inefficient use of network resources. The introduction of a hysteresis margin reduces the effects of the ping-pong problem on the handover decision. A hysteresis margin of 20ms was chosen for the proposed handover scheme. Again this was a trade-off between the accuracy of the handover and the speed at which it was performed.

The setup also included several wireless stations (BG 1 - 5 in Fig. 6.6) used to generate background traffic. The introduction of more traffic into a WLAN cell increased the congestion level in that cell. This congestion resulted in more collisions occurring, and had the effect of increasing the delay experienced by packets in that particular cell. When this occurred the proposed scheme performed handover from the path through the congested network to the secondary path, through Cell 2 which was less congested.

The SCTP association was established and traffic was transmitted for a duration of 60 seconds. The setup utilized the same simplified transmission scheme as was employed in Section 5.4 and Section 6.1.2, namely that Endpoint B had no data to send and only transmitted acknowledgements of the data that it received from Endpoint A. The SCTP traffic used was designed to emulate voice traffic according to the G.723.1 [57] recommendation. According to this recommendation an 80 byte message was sent every 20 milliseconds [56]. The background traffic consisted of 1500 byte UDP packets, generated by the Iperf package[4].
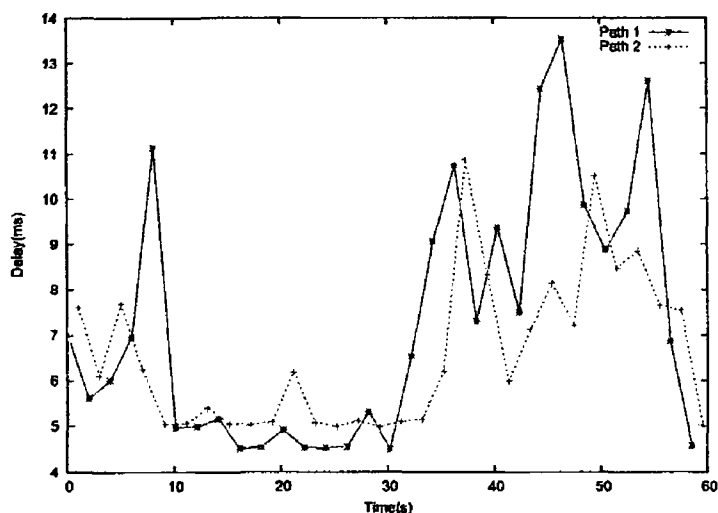

Figure 6.7: Delays Recorded With 3 Background Stations (80 bytes)

Background traffic was introduced 20 seconds after the association had been established. Initially, 3 background traffic sources were used. However, this did not incur sufficient congestion to induce handover of the SCTP association. The delays recorded due to 3 stations generating background traffic are shown in Fig. 6.7. It can be seen from this graph that the difference in delay between the two paths never exceeded the implemented hysteresis margin of 20ms, and therefore handover never occurred.

The experiment was repeated with 4 background stations generating traffic. The delays experienced by SCTP for this scenario were recorded and can be seen in Fig. 6.8.

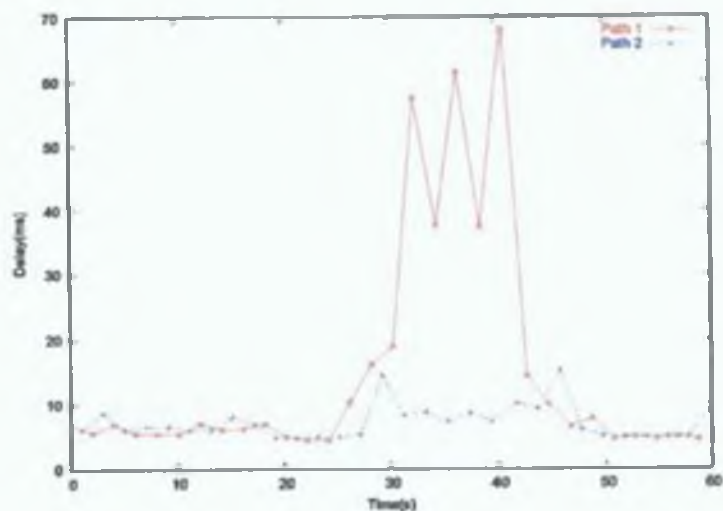[4]available from http://dast.nlanr.net/Projects/Iperf/

Figure 6.8: Delays Recorded With 4 Background Stations (80 bytes)

From this graph it can be observed that the delay on the initial primary path, the path through Cell 1, exceeded the delay on the secondary path by more than the hysteresis margin, and as result of this handover was performed to the path through Cell 2. Once the background stations finish transmitting the delay in Cell 1 returned to the level of an uncongested cell, similar to Cell 2. However, since a hysteresis margin had been implemented, handover did not occur from Cell 2 back to Cell 1. The handover from Cell 1 to Cell 2, due to the presence of background UDP traffic is illustrated in Fig. 6.9.
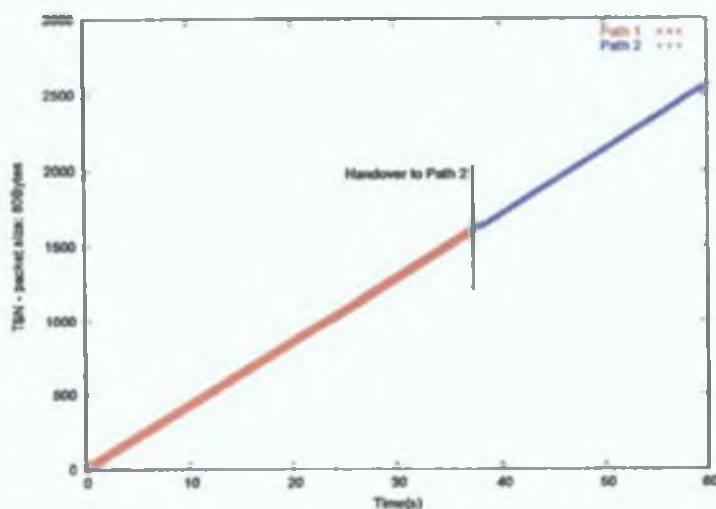

Figure 6.9: WLAN Handover With 4 Background Stations (80 bytes)

89

The experiment was repeated once again, with the number of background stations increased to 5. The results were similar to those for 4 background stations. The recorded delays are shown in Fig. 6.10 and the handover is illustrated in Fig. 6.11.
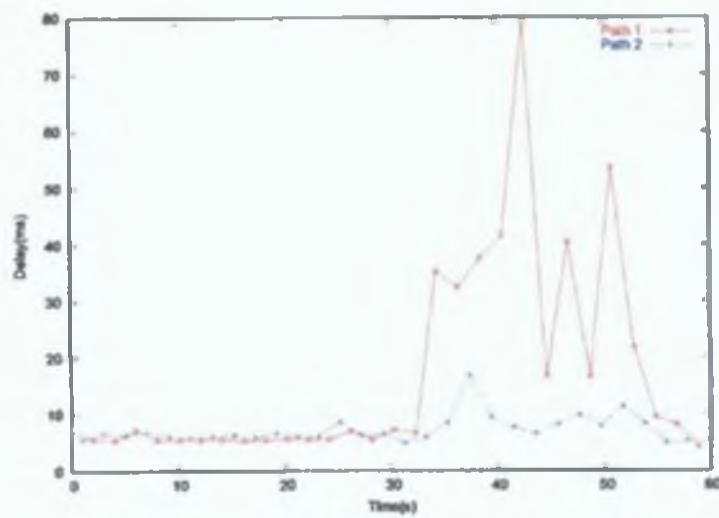

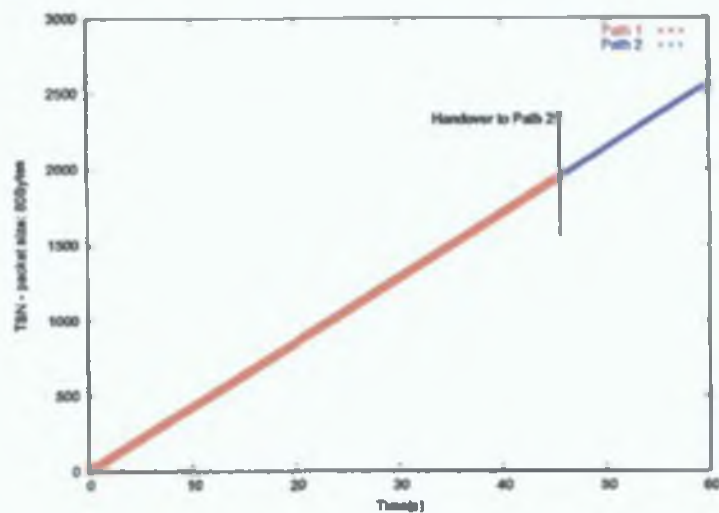Figure 6.10: Delays Recorded With 5 Background Stations (80 bytes)

Figure 6.11: WLAN Handover With 5 Background Stations (80 bytes)

90

The SCTP packet size was increased to 480 bytes. This represented the average packet size of WWW traffic over a GPRS/UMTS connection [58, 59]. The above experiments were repeated. As in the previous tests, the congestion introduced due to 3 background stations was not enough to cause handover to occur. This can be seen in Fig. 6.12. The number of background stations was once again increased to 4 and the experiment repeated. The delays recorded by the handover scheme can be seen in Fig. 6.13. From these it can be seen that the delay on the path through Cell 1 increased above the delay in Cell 2 at 26 seconds. Correlating this with the SCTP packet trace shown in Fig. 6.14, it can be seen that at approximately this time handover to Path 2 occurs. Due to the inclusion of a hysteresis margin handover was not performed back to Path 1 when the congestion had passed. The experiment was repeated once again with 5 background stations. The delays for this scenario are shown in Fig. 6.15 and the handover is illustrated in Fig. 6.16.
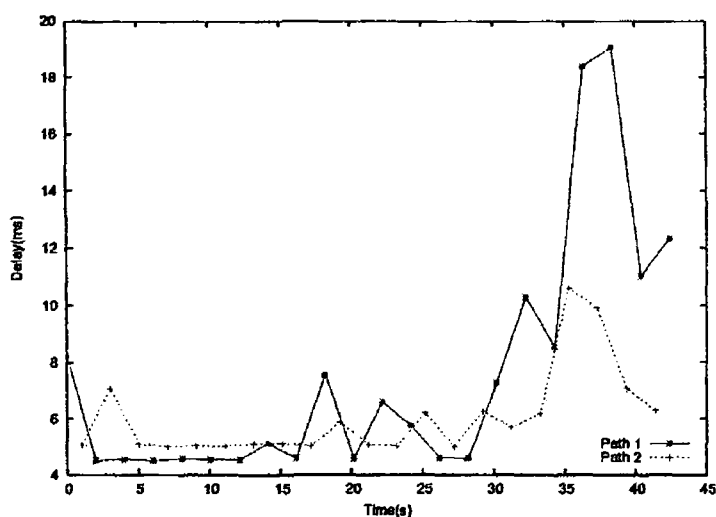

Figure 6.12: Delays Recorded With 3 Background Stations (480 bytes)
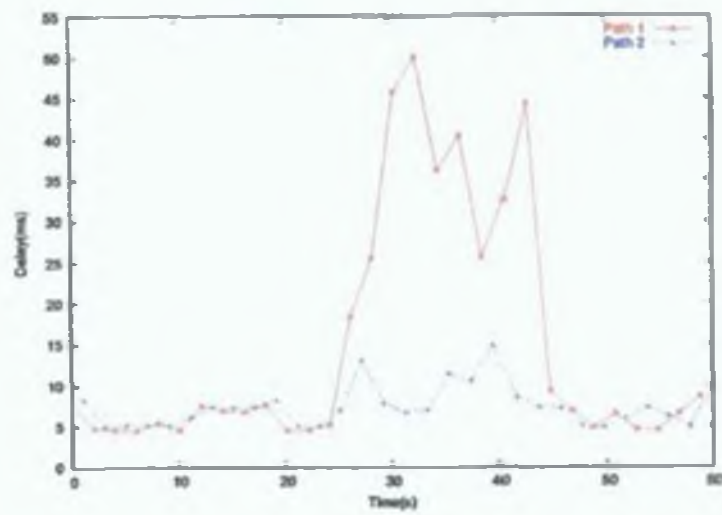
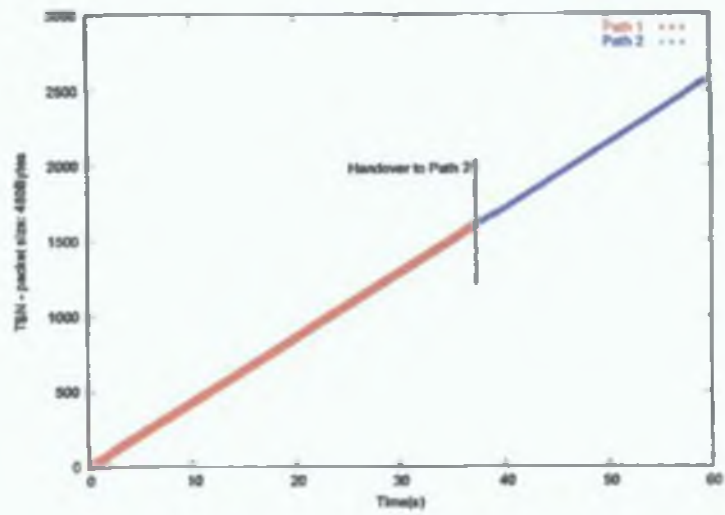Figure 6.13: Delays Recorded With 4 Background Stations (480 bytes)

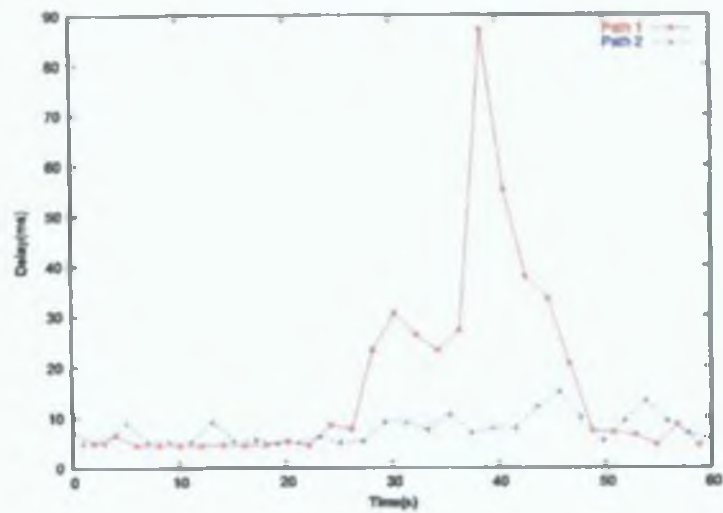Figure 6.14: WLAN Handover With 4 Background Stations (480 bytes)

92

Figure 6.15: Delays Recorded With 5 Background Stations (480 bytes)
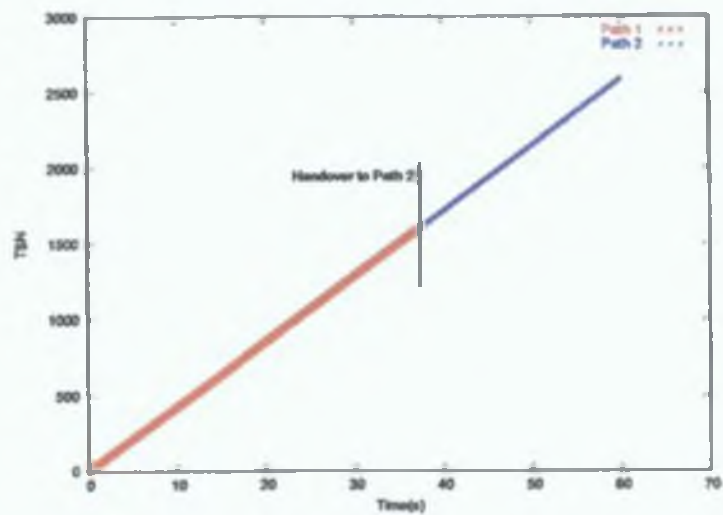

Figure 6.16: WLAN Handover With 5 Background Stations (480 bytes)

As described in Section 6.1.2, there was a lag between detecting that a cell had become congested, as seen in the graphs of recorded delays, and responding to this congestion by performing handover, as seen in the handover graphs. This lag was due to several factors, the most significant of which was the use of an average of the delay values for each path in making the handover decision. The average path delay was chosen as a means of minimizing the effect delay spikes had on handover. Making a handover decision based on one set of measurements, which could include a delay spike, could result in many unnecessary handovers which would incur high handover overhead. As the proposed handover scheme performs seamless handover, the performance decrease did not result in lost packets.

6.2.2 Effects of Mobility
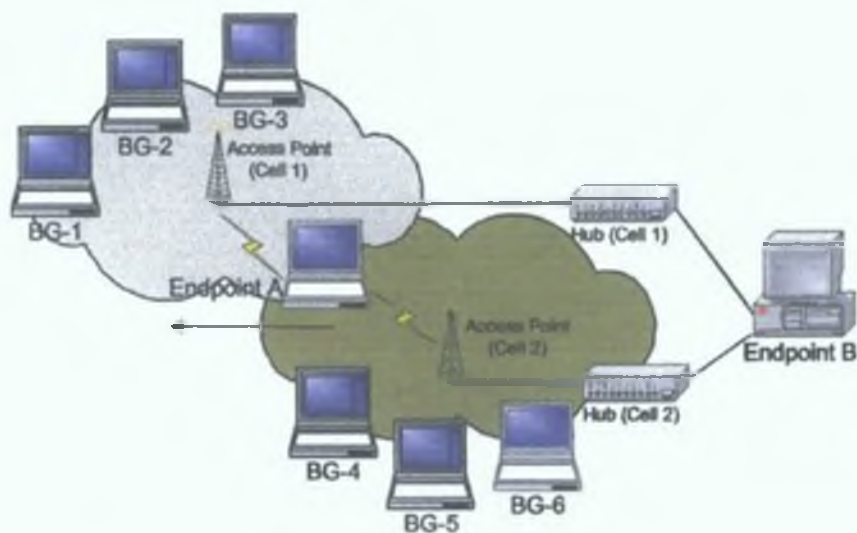

Figure 6.17: WLAN Equipment Setup for Mobility Tests

The setup shown in Fig. 6.17 was used to evaluate the effects of mobility on the proposed handover scheme. The setup included two multihomed SCTP endpoints, Endpoint A (mobile) and Endpoint B (fixed). Endpoint A was located initially in a region where the coverage of the APs overlapped. Each cell contained three wireless

94

stations used to generate background traffic.

All background stations were set to transmit the same amount of traffic, providing comparable levels of congestion in each cell. Endpoint A moved from the coverage of Cell 2 towards AP1 in Cell 1. The RTTs recorded during this transition failed to increase to a sufficient level to induce handover. These delays are shown in Fig. 6.18.
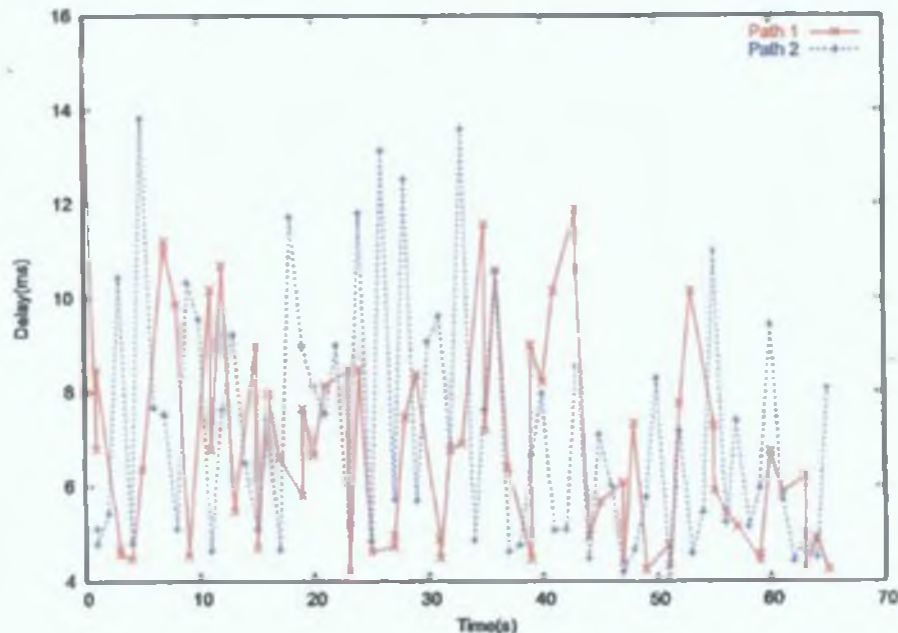
Figure 6.18: WLAN Path Delays Due To Mobility Of Endpoint A

As can be seen in Fig. 6.18 movement within a WLAN cell did not cause a significant increase in the delay experienced by traffic within the cell. Furthermore, as shown in [60], the path delay in a WLAN cell only increases dramatically once the border of the cell has been reached. In the case of the primary path of an association this will result in the loss of data packets. The data will be retransmitted on the secondary path however, there is an incurred latency associated with retransmission. The handover scheme being proposed in this dissertation aims to minimize this latency by performing handover immediately upon the detection of a missing HEARTBEAT-ACK on the primary path. Missing HEARTBEAT-ACKs on the secondary paths are detected using current SCTP methods, as described in [12]. This scheme only incurs the latency due to one

95

timeout, the timeout to detect the missing HEARTBEAT-ACK on the primary path.

The setup used to examine the effects of packet loss on the primary path is shown in Fig. 6.17. The wireless stations were set as before, to generate comparable levels of congestion within each cell. The trace of TSNs obtained can be seen in Fig. 6.20. As can be seen in Fig. 6.20, handover was performed after approximately 30 seconds. This correlates with Fig. 6.19 which shows when the HEARTBEAT was lost on the primary path.[5]
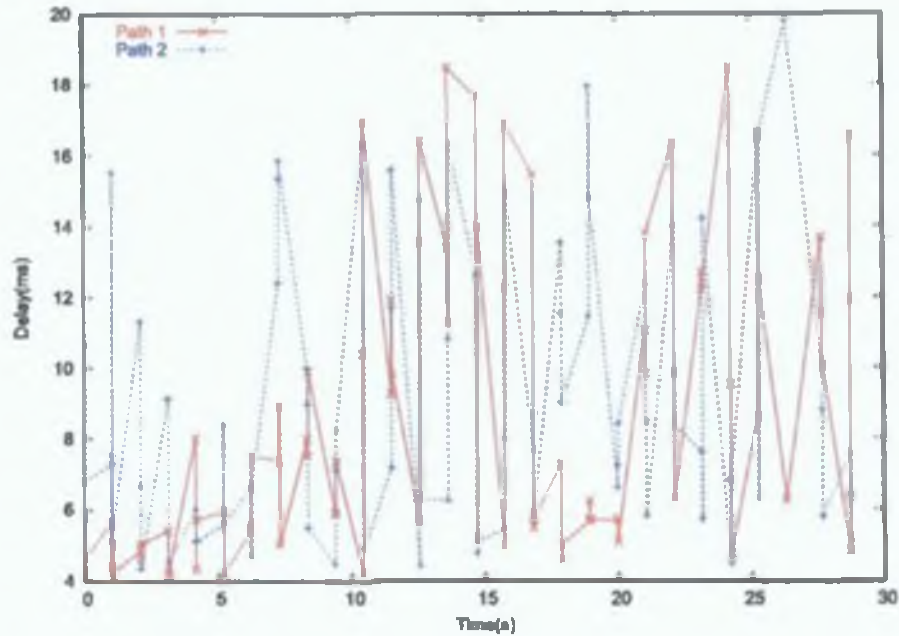
Figure 6.19: WLAN Path Delays As Endpoint A Moves Out Of Coverage Of AP2

As illustrated in Fig. 6.20 the handover that performed was not as seamless as the handovers performed due to congestion effects. The reason for this was due to the packet loss and timeout needed to induce handover to the secondary path.

---

[5]As there are only two paths in the association once the HEARTBEAT is lost on the primary path the heartbeating process is postponed until the primary destination address is reachable again. This does not occur during the experiment.
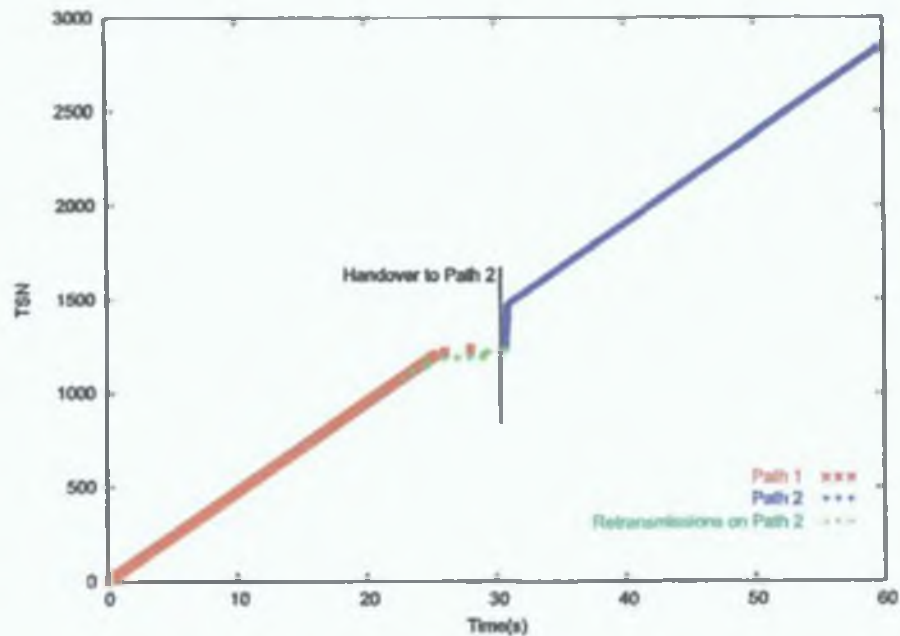
Figure 6.20: WLAN Handover TSN Trace As Endpoint A Moves Out Of Coverage Of AP2

6.3 Summary

The proposed handover scheme was implemented using the reference implementation of SCTP as a base. The proposed scheme was tested within the confines of an emulated environment using the NIST Net package. This tool provided a guide to the kind of behaviour that could be expected when deploying the scheme within a wired environment. It also allowed for troubleshooting of the implemented code. Once it had been established that the scheme was functioning as envisaged, it was next deployed in a wireless access network. IEEE 802.11b WLAN was chosen as the wireless network to use due to the popularity of the medium and the availability of sufficient hardware. The results of deploying the scheme within such an environment showed that a threshold of 4 background stations was required to increase the congestion within a WLAN cell to a level where the scheme would perform handover. It was also discovered that the delay experienced in an uncongested cell only increased when the cell boundary was reached.

The evaluation of the handover scheme proposed by this dissertation showed that

although it is of limited accuracy, it does provide the ability to perform vertical handover from a congested link to a less congested link. This simple scheme offers the benefit of avoiding excessive timeouts before performing handover, as is the case with the current SCTP handover scheme. The scheme can also perform handover due to packet loss on the primary path. While this also has the benefit of minimizing the number of timeouts needed to induce handover, it does not perform handover completely seamlessly.

The results of Section 6.2 were obtained by performing handover within a homogeneous WLAN domain. In a heterogeneous wireless environment it is more likely that handover will be performed from networks of vastly different round trip delays, e.g. from a GPRS connection to a WLAN link [61]. In this context, the proposed scheme could perform handover to the higher bandwidth WLAN connection for the duration of the user's residence in the WLAN cell and hand back to the GPRS connection when the user had reached the coverage limit of the WLAN basestation (Access Point).

Chapter 7

Further Work and Conclusions

7.1　Further Work

The limitations of Mobile IP such as long handover delays and additional complexity in the form of Agents has led to a desire to perform handover higher up the protocol stack. The first layer above Mobile IP is the transport layer, the first end-to-end layer in the stack, it is also the layer at which SCTP resides. Current research in SCTP involves investigating whether its multihoming feature could provide a solution for performing handover. It has been shown using the simple mechanism proposed by this dissertation that SCTP can facilitate handover. However, this handover ability of SCTP is limited to the destination addresses negotiated by the endpoints at association establishment. This means that the mobility of the user would need to be known in advance so that an address for each network that the user will visit can be included in the four-way message exchange at the establishment of the association. This is often not feasible. The designers of SCTP have proposed the DAR extension to SCTP. This extension, though still at Internet Draft stage, provides a mechanism by which addresses acquired by an endpoint after the association has been setup can be added to the association without requiring a restart of the association. This extension, coupled with the location

management capabilities of SIP, would provide a mobility solution for IP-based hosts that would negate the need for Mobile IP.

7.1.1 SCTP Dynamic Address Reconfiguration Extension

It is believed that the DAR extension [55] could be used to provide support for host mobility using SCTP [2]. Using SCTP to support IP mobility would negate the need for a complex solution like Mobile IP, and the additional overhead that the Mobile IP solution includes (high handover latency, introduction of network agents, etc.)

The DAR extension defines two new chunk types, six new parameter types, and several new error causes. The two new chunk types are listed in Table 7.1. The ASCONF chunk is used by an endpoint requesting any of the features described by the extension. Its structure is similar to the chunks described in Section 2.2.1. The new features, such as the addition of a new IP address to an established association, or setting an alternate address as the primary destination address for the association, are defined in the six new parameter types.

The ASCONF-ACK chunk is sent in response to ASCONF chunks. The ASCONF-ACK chunk contains a parameter response or an error code, informing the requesting endpoint of the status of the request it has made.

Chunk Type	Chunk Name	Description
0xC1	ASCONF	Address Configuration Change Chunk
0x80	ASCONF-ACK	Address Configuration Acknowledgement

Table 7.1: SCTP Chunk Types

Combining the proposed handover scheme with the functionality provided by the DAR extension would enable SCTP to fully support user mobility without employing a solution such as Mobile IP. The "mobile-aware" SCTP would also offer the advantage of facilitating seamless handover. This is a process of establishing a new connection

before breaking the current connection, resulting in less packet loss due to handover. The mobility scheme using SIP does not currently offer this functionality.

7.1.2 Primary Path Failure

The current SCTP handover scheme, as described in [6], performs handover only after a number of timeouts have occurred on the primary path. The handover scheme proposed by this dissertation aims to mitigate these timeouts by performing handover based on RTT measurements made on each path in an association. These measurements are gathered periodically using SCTP's HEARTBEAT chunks. However, the scheme can only make a handover decision once all path delays have been gathered. If a HEARTBEAT or HEARTBEAT-ACK chunk is lost on a particular path, then no RTT can be obtained for that path. The easiest solution to this problem is to simply exclude the corresponding destination address from the handover decision, and continue by sending a HEARTBEAT to the next destination address in the association. However, the question of what should occur in the event of primary path failure needs to be explored. The proposed handover scheme currently uses a timer to detect any loss of HEARTBEAT chunks on the primary path and perform handover to the secondary path once the loss has been detected. This method has the advantage of only incurring a delay of one timeout, however, this timeout value would require further tuning to make the handover as seamless as possible.

A second solution to the problem of sudden failure of the primary path is to use DATA chunks to update the RTT of the primary path. Currently this is how SCTP updates the primary path RTT, and subsequent RTO. Handover could be performed due to several DATA retransmissions instead of a timeout. This could provide a more seamless handover than the current implementation of the proposed scheme. Both of these solutions also have drawbacks associated with them, e.g. needless handover due to a large delay spike causing a number of timeouts at the sending endpoint. These delay

spikes are a common occurrence in WLAN networks. Assessing fully the implications of each of these and other scenarios and determining the correct strategy to employ will require further research.

7.2   Conclusions

The increasing demand of mobile users for access to services formerly exclusive to wired networks has led to the introduction of these services within a wireless environment. However, the characteristics of such environments are not identical to their wired counterparts and issues such as host mobility need to be addressed. Host mobility requires that a host remain connected to all services that it is currently subscribed to whilst being allowed to move freely between cells of the same network, or even between other networks.

This introduces the concept of handover. Traditionally handover was between cells on the same network and, as such, was controlled by the network itself, e.g. GSM. However, the emerging mobile environments are heterogeneous, with several competing access technologies providing a range of bandwidth and coverage. The host chooses which wireless network to use on the basis of the demands of the application that it wishes to run. The process of performing handover from one wireless access network to a different access network is termed vertical handover, and is central to host mobility within a heterogeneous wireless environment.

Mobility of IP-based hosts is provided through the use of Mobile IP. However, Mobile IP is not an efficient solution to IP-based host mobility. It has been proposed that SCTP could provide a solution to IP-based hosts in a mobile environment. The multi-homing feature of SCTP allows an endpoint to connect through several networks when establishing an association. However, the process of handover in SCTP currently is only performed on a failure basis. Switching data transmission to one of the other, sec-

ondary, destination addresses is only performed when the current primary destination address is found to be unreachable. The process of determining that the current primary destination address involves experiencing a number of timeouts. These timeouts have associated delays resulting in a large latency between when the primary destination address fails and when handover to one of the secondary paths is performed.

This dissertation has outlined the disadvantages of this failure-oriented scheme and proposed a new delay-centric scheme that uses the round trip delay to each destination address as the criterion upon which to make the handover decision. This new scheme employs a more aggressive polling strategy in order to determine these round trip delays, and performs handover based on these delays. This scheme, though simple in nature, has been shown to offer a more efficient method of handover than the currently implemented mechanism, namely by avoiding the excessive timeouts associated with the current scheme. Both the results of simulation and emulation verify this assertion.

It is believed that coupling this simplified scheme with the functionality of dynamically adding destination addresses could accommodate IP-based host mobility at the transport layer, negating the need for a complex solution such as Mobile IP.

Acronyms

ARP Address Resolution Protocol

AS Autonomous System

AuC Authentication Centre

BER Bit Error Rate

BGP Border Gateway Protocol

BSC Base Station Controller

BSS Base Station Subsystem

BTS Base Transceiver Station

CBR Constant Bit Rate

CEPT Conference of European Posts and Telecommunications

CN Corresponding Node

CoA Care-of Address

CSMA/CD Carrier Sense Multiple Access with Collision Detection

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance

CTS Clear To Send

DAR Dynamic Address Reconfiguration

DCF Distributed Coordination Function

DDoS Distributed Denial of Service

DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

DoS Denial of Service

DRCP Dynamic Rapid Configuration Protocol

EIR Equipment Identity Register

FA Foreign Agent

FDMA Frequency Division Multiple Access

FTP File Transfer Protocol

GGSN Gateway GPRS Support Node

GMSK Gaussian Minimum Shift Keying

GPRS General Packet Radio Service

GSM Global System for Mobile Communications

GUI Graphical User Interface

HA Home Agent

HIP Host Identity Payload

HLR Home Location Register

HTTP Hyper Text Transfer Protocol

IANA Internet Assigned Numbers Authority

IEEE Institute of Electrical and Electronics Engineers

IESG Internet Engineering Steering Group

IETF Internet Engineering Task Force

IMEI International Mobile Equipment Identity

IMSI International Mobile Subscriber Identity

IP Internet Protocol

IPv6 Internet Protocol version 6

ISDN Integrated Services Digital Network

ISM Industrial, Scientific, and Medical

ITU International Telecommunications Union

LAN Local Area Network

M-SCTP Mobile SCTP

MDTP Multi-network Datagram Transmission Protocol

MH Mobile Host

MIS Maximum Inbound Streams

MS Mobile Station

MSC Mobile services Switching Centre

MTU Maximum Transmission Unit

NAV Network Allocation Vector

NIC Network Interface Card

OSI Open Systems Interconnection

OSPF Open Shortest Path First

PC Personal Computer

PCF Point Coordination Function

PDA Personal Digital Assistant

PIN Personal Identity Number

PSTN Public Switched Telephony Network

QoS Quality of Service

RFC Request for Comment

RIP Routing Information Protocol

RTP Real Time Protocol

RTO Retransmission Time-Out

RTS Request To Send

RTT Round Trip Time

SACK Selective Acknowledgement

SCTP Stream Control Transmission Protocol

SDP Session Description Protocol

SGSN Serving GPRS Support Node

SIGTRAN Signalling Transport

SIM Subscriber Identity Module

SIP Session Initiation Protocol

SS7 Signalling System Number 7

SSN Stream Sequence Number

TAD Transport Area Directorate

TCB Transmission Control Block

TCH Traffic Channel

TCP Transmission Control Protocol

TDMA Time Division Multiple Access

TLV Time-Length-Value

TSN Transmission Sequence Number

UA User Agent

UDP User Datagram Protocol

ULP Upper Layer Protocol

UMTS Universal Mobile Telephony System

URI Universal Resource Identifier

VLR Visitor Location Register

VOD Video On Demand

VoIP Voice over Internet Protocol

WISP Wireless Internet Service Provider

WLAN Wireless Local Area Network

WWW World Wide Web

References

[1] F. Halsall. *Data Communications, Computer Networks and Open Systems, Fourth Edition*. Addison Wesley, 1996.

[2] S. Fu, M. Atiquzzaman, L. Ma, W. Ivancic, Y. Lee, J. Jones, and S. Lu. TraSH: A Transport Layer Seamless Handover for Mobile Networks. Technical Report, 2004.

[3] A. Kelly, G.-M. Muntean, P. Perry, and J. Murphy. Delay-Centric Handover in SCTP over WLAN. *Transactions on Automatic Control and Computer Science*, 49(4), 2004.

[4] R. Travis. *Signaling System #7*. McGraw-Hill, 2000.

[5] J. Postel. RFC 793: Transmission Control Protocol, 1981.

[6] R. Stewart and Q. Xie. *Stream Control Transmission Protocol (SCTP): A Reference Guide*. Addison Wesley, 2001.

[7] P. Ferguson and D. Senie. RFC 2827: Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, 2000.

[8] F. Baker and P. Savola. RFC 3704: Ingress Filtering for Multihomed Networks, 2004.

[9] J. Postel. RFC 768: User Datagram Protocol, 1980.

[10] W. R. Stevens. *TCP/IP Illustrated, Volume 1*. Addison Wesley, 1994.

[11] K. Fall and S. Floyd. Simulation-based comparisons of Tahoe, Reno and SACK TCP. *SIGCOMM Comput. Commun. Rev.*, 26(3):5–21, 1996.

[12] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson. RFC 2960: Stream Control Transmission Protocol, 2000.

[13] R. T. Braden. RFC 1122: Requirements for Internet hosts — communication layers, 1989.

[14] S. Fu and M. Atiquzzaman. SCTP: State of the Art in Research, Products, and Technical Challenges. *IEEE Communications Magazine*, 42(4):64–76, 2004.

[15] G. J. Heinz and P. D. Amer. Priorities in Stream Transmission Control Protocol (SCTP) Multistreaming. In *8th World Multi-Conference on Systemics, Cybernetics and Informatics*, 2004.

[16] R. Steele and L. Hanzo. *Mobile Radio Communications, 2nd Edition*. Wiley, 1999.

[17] D. M. Balston. Pan-European Cellular Radio or 1991 and all that. *Electronics and Communications Engineering Journal*, pages 7–13, 1989.

[18] R. Steele, C. Lee, and P. Gould. *GSM, cdmaOne and 3G Systems*. Wiley, 2001.

[19] A. Mehrotra. *GSM System Engineering*. Artech House, 1997.

[20] A. Mehrotra. *Cellular Radio Performance Engineering*. Artech House, 1994.

[21] M. Mouly and M.B. Pautet. *The GSM System for Mobile Communications*. Published by the authors, 1992.

[22] A. S. Tanenbaum. *Computer Networks, Fourth Edition*. Prentice Hall PTR, 2003.

[23] D. C. Plummer. RFC 826: An Ethernet Address Resolution Protocol – or – Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware, 1982.

[24] C. Perkins. RFC 3344: IP Mobility Support for IPv4, 2002.

[25] C. E. Perkins. Mobile Networking Through Mobile IP. *IEEE Internet Computing*, 2(1):58–69, 1998.

[26] K. El Malki. Low Latency Handoffs in Mobile IPv4. Internet Draft, draft-ietf-mobileip-lowlatency-handoffs-v4-09.txt, expires Dec. 2004, 2004.

[27] G. Montenegro and V. Gupta. RFC 2356: Sun's SKIP Firewall Traversal for Mobile IP, 1998.

[28] A. G. Valko. Cellular IP - A New Approach to Internet Host Mobility. *ACM SIGCOMM Computer Communication Review*, 29(1):50–65, 1999.

[29] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. RFC 3261: SIP: Session Initiation Protocol, 2002.

[30] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 3550: RTP: A Transport Protocol for Real-Time Applications, 2003.

[31] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. RFC 2616: Hypertext Transfer Protocol – HTTP/1.1, 1999.

[32] E. Wedlund and H. Schulzrinne. Mobility Support using SIP. In *WOWMOM*, pages 76–82, 1999.

[33] H. Shulzrinne and E. Wedlund. Application-Layer Mobility Using SIP. *SIGMO-BILE Mobile Computing and Communications Review*, 4(3):47–57, 2000.

[34] J. Rosenberg, H. Shulzrinne, and J. Peterson. Third Party Call Control in SIP, 2000. Internet Draft, draft-rosenberg-sip-3pcc-00.txt, expired,.

[35] R. Sparks. The SIP Referred-By Mechanism, 2004. Internet Draft, draft-ietf-sip-referredby-05, expired.

[36] N. Akhtar, M. Georgiades, C. Politis, and R. Tafazolli. SIP-based End System Mobility Solution for All-IP Infrastructures. *IST Mobile and Wireless Communications Summit*, 2003.

[37] A. Dutta, J.C. Chen, S. Das, M. Elaoud, D. Famolari, S. Madhani, A. McAuley, M. Tauil, S. Baba, T. Maeda, N. Nakajima, Y. Ohba, and H. Schulzrinne. Implementing a Testbed for Mobile Multimedia. *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)*, 2001.

[38] A. Snoeren and H. Balakrishnan. An End-to-End Approach to Host Mobility. In *Proc. ACM MOBICOM*, pages 155–166, 2000.

[39] P. Mockapetris. RFC 1035: Domain names - implementation and specification, 1987.

[40] R. Droms. RFC 1531: Dynamic Host Configuration Protocol, 1993.

[41] R. Moscowitz. Host Identity Payload and Protocol. Internet Draft, draft-moscowitz-hip-05.txt expired, 2001.

[42] T. R. Henderson. Host Mobility for IP Networks: A Comparison. *IEEE Network*, 17(6):18–26, 2003.

[43] A. Kelly, P. Perry, and J. Murphy. A Modified SCTP Handover Scheme for Real Time Traffic. In *Proc. of HETNETs 03 : First International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks*, 2003.

[44] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modelling TCP Throughput: A Simple Model and its Empirical Validation. *IEEE/ACM Transactions on Networking*, 8(2), 2000.

[45] R. Rajamani, S. Kumar, and N. Gupta. SCTP versus TCP: Comparing the Performance of Transport Protocols for Web Traffic. University of Wisconsin-Madison, 2002.

[46] G.-M. Muntean, P. Perry, and L. Murphy. A New Adaptive Multimedia Streaming System for All-IP Multi-Service Networks. *IEEE Transactions on Broadcasting*, 50(1):1–10, 2004.

[47] D. He and C.Q. Shen. Simulation Study of IEEE 802.11e EDCF. In *Proc. of IEEE Vehicular Technology Conference (VTC-03 Spring)*, 2003.

[48] G. Camarillo, R. Kantola, and H. Schulzrinne. Evaluation of Transport Protocols for the Session Initation Protocol. *IEEE Network*, 17(5):40–46, 2003.

[49] G. Camarillo, H. Schulzrinne, and R. Kantola. A transport protocol for SIP. Ericsson, Finland, 2001.

[50] G. De Marco, D. De Vito, M. Longo, and S. Loreto. SCTP as a transport for SIP: a case study. In *SCI 2003*, 2003.

[51] M. Atiquzzaman and W. Ivancic. Evaluation of SCTP Multistreaming over Satellite Links. In *12th International Conference on Computer Communications and Networks*, 2003.

[52] R. Alamgir, M. Atiquzzaman, and W. Ivancic. Effect of Congestion Control on the Performance of TCP and SCTP over Satellite Networks. In *NASA Earth Science Technology Conference*, 2002.

[53] G. Ye, T. Saadawi, and M. Lee. SCTP Congestion Control Performance in Wireless Multi-hop Networks. In *MILCOM 2002 - IEEE Military Communications Conference*, pages 933–938, 2002.

[54] L. Ma, F. Yu, V. Leung, and T. Randhawa. A New Method to Support UMTS/WLAN Vertical Handover Using SCTP. In *Proc. of IEEE Vehicular Technology Conference (VTC-03 Fall)*, 2003.

[55] R. Stewart, M. Ramalho, Q. Xie, M. Tuexen, and P. Conrad. Stream Control Transmission Protocol (SCTP) Dynamic Address Reconfiguration, 2004. Internet Draft, draft-ietf-tsvwg-addip-sctp-09.txt, expires Dec. 2004.

[56] J. Noonan, A. Kelly, P. Perry, S. Murphy, and J. Murphy. Simulations Of Multimedia Traffic Over SCTP Modified For Delay-Centric Handover. In *Proc. of World Wireless Congress*, 2004.

[57] International Telecommunication Union. G.723.1: Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s,, 1996. Series G: Transmission Sytems and Media, Digital Sytems and Networks.

[58] T. V. Prabhakar, J. Kuri, K. N. Kiran, and M. Kartik. An Adaptive Timeout Scheme For Short TCP Flows Over 3G Wireless Networks. 2003.

[59] K. Parsa, S.S. Ghassemzadeh, and S. Kazeminejad. Systems Engineering of Data Services in UMTS W-CDMA Systems. In *International Conference on Communications - ICC 2001*, 2001.

[60] G. Cunningham, P. Perry, and L. Murphy. Soft, Vertical Handover of Streamed Video. In *Proc. of IEE 5th International Conference on 3G Mobile Communications Technologies*, 2004.

[61] Q. Zhang, C. Guo, Z. Guo, and W. Zhu. Efficient Mobility Management for Vertical Handoff between WWAN and WLAN. *IEEE Communications Magazine*, 41(11):102–108, 2003.

List of Publications

[3] A. Kelly, G.-M. Muntean, P. Perry, and J. Murphy. Delay-Centric Handover in SCTP over WLAN. *Transactions on Automatic Control and Computer Science*, 49(4), 2004.

[56] J. Noonan, A. Kelly, P. Perry, S. Murphy, and J. Murphy. Simulations Of Multimedia Traffic Over SCTP Modified For Delay-Centric Handover. In *Proc. of World Wireless Congress*, 2004.

[43] A. Kelly, P. Perry, and J. Murphy. A Modified SCTP Handover Scheme for Real Time Traffic. In *Proc. of HETNETs 03 : First International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks*, 2003.