Umeå University
Department of Computing Science
Simon Nordberg
tfy99sng@cs.umu.se
3rd December 2003

3G IPv6 Node Emulator

Graduation / Master's Thesis

Advisors
Karim El-Malki, Gianluca Verin, Thomas Nilsson

Examiner
Per Lindström

Abstract

The mobility market and the increasing demand for multimedia services has spurred the development of several technologies aiming to satisfy the demands and to introduce new services. The Universal Mobile Telecommunications System (UMTS) being developed today is expected to significantly improve and extend the services provided by today's cellular mobile communication systems such as GSM. In the area of data communication, the next generation Internet Protocol named IPv6 aims to provide services surpassing those currently offered by IPv4. This paper presents the in-depth implementation details of a Gateway GPRS Support Node (GGSN) emulator designed to run on a PC. The emulator is intended to be used in a closed testing environment for end-to-end testing of IPv6 terminals. Furthermore the fusion of cellular mobile communication systems and data communication systems with respect to UMTS and IPv6 is examined. Several issues and their solutions mainly concerning multimedia services such as video conferencing, Quality of Service (QoS) and real-time traffic are highlighted.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Definition of terms

- The term “octet” refers to 8 bits of data. In most computer architectures, this also constitutes a “byte”.
- The term “host” in this paper, refer to a computer connected to a TCP/IP network. Another synonym is “node”, same as in graph theory.
- The term “packet” refers to a transport layer header plus payload.
- The term “path” denotes the set of links traversed by a packet between a source node and a destination node.
- The term “MTU” denotes the Maximum Transfer Unit, i.e. the maximum size of a frame to be transmitted across a link.
- The term “path MTU” denotes the smallest of the MTUs of the links composing the path.

Report outline

This paper begins by defining the purpose and motivation of the thesis in chapter 2. In chapter 3 the history of mobility and the Internet is discussed. In chapter 4 and 5 the reader is acquainted with the role of an internet protocol and provided with background information on the technology used in today’s Internet called TCP/IP. Chapters 6, 7 and 8 provides an introduction to UMTS systems, IPv6 and IPv6 over UMTS systems respectively. The implementation details of the GGSN can be found in chapter 9, including design choices and other details of interest. Finally, chapter 10 draws conclusions from this work and discusses issues to consider for future refinements.

It is assumed the reader has a basic understanding of concepts and techniques in computer networking. For a more in-depth look on the subjects of IPv6, an appendix is available. To fully grasp the complex UMTS system discussed, the reader is encouraged to acquire understanding of the GSM technology.

Chapter 2

Purpose and motivation

The purpose of this thesis was to design and implement a Gateway GPRS Support Node (GGSN) using IPv6 as user transport protocol. The work was conducted after an initial proposal by Ericsson AB and performed as a Master's Thesis project.

The purpose of the resulting system was a platform to be used for software and service testing in a laboratory environment. The node would be compatible with existing systems with respect to control messages and networking interfaces. The GGSN "emulator" was meant to be placed in an existing 3G network for testing purposes.

Implementation was conducted with basis in specifications by 3GPP to ensure compatibility.

The report focuses on the merging of the two recent technologies: 3G Mobile systems and IPv6. An overview of the systems are presented separately, highlighting key details.

Due to the industrial and academic nature of the project three advisors were involved, Karim El-Malki (Ericsson AB), Gianluca Verin (Ericsson AB) and Thomas Nilsson (Department of Computing Science, Umeå University).

Chapter 3

Brief history of mobility and the Internet

The Internet was launched in the late 60's as a result of work by Defense Advanced Research Projects Agency (DARPA), who saw great potential value in allowing computers to share research and military information between distant parts of the world. [18]

In the beginning there were only a handful of hosts connected, but soon more and more universities joined to create a US national network. The Internet, then known as Advanced Research Project Agency Network (ARPANET), was used exclusively by computer experts, scientists and engineers, leaving little room for non-technical users. [11]

As more protocols got standardized, the Internet became more user friendly allowing less technical skilled people to learn how to use the network. Still it was by no means trivial, judging by today's standards. [11]

With research being conducted within the areas of transport and routing (Directing packets through a datagram network), an architecture called Transmission Control Protocol / Internet Protocol (TCP/IP) found its way into the Internet in the 70's. [18]

The two protocols deal with separate issues within this packet oriented network approach. To deliver a single packet of data across a network consisting of several hops (i.e. routers), decisions have to be made in each forwarding node on where to send it next. This is the basic functionality for IP, providing a way of navigating data through a network of nodes. TCP's task is to ensure a reliable point to point connection (i.e. not being concerned with the number of hops and other network layout) between two hosts over several hops.

As a result of this technological breakthrough the Internet gained tremendous momentum and took a great leap forward. A large portion of the success of TCP/IP was due to the University of California in Berkeley that published their own implementation of the architecture as public domain software. The Berkeley System Distribution (BSD) with its openness and superior network "stack" became a foundation for (not exclusively, but among other things) the network facilities in all modern operating systems. Due to this, the Internet as we know it owes its existence to the networking code originating from BSD. [31, 24, 30]

In 1991 came the first user friendly interface to the Internet, developed at the University of Minnesota. This system, called *Gopher* made it possible for people without UNIX skill to use the Internet. During the same time period, another protocol for information distribution was developed at the European Laboratory for Particle Physics (also known as CERN). This protocol, using embedded text links called hypertext, later became what we today know as the World Wide Web (WWW). [23, 18]

A tremendous amount of work was done by the National Center for Supercomputing (NCSA) in the beginning of the 90's to create a graphical browser for the WWW protocol. This software called *Mosaic*, combined with a rapidly growing user base, gave the protocol its big boost. The lead people behind this software moved on to form Netscape Corporation, later developing the most popular graphical browser for public use. [18, 20]

Around the same time several Internet Service Providers (ISP) launched, intensifying the commercially based Internet. With the commercial aspects in mind, Microsoft gave full throttle straight for the up and coming market. With their goal set on bringing the Internet to the general audience, Internet Explorer (IE) was tightly integrated with their next Windows release, Windows 98. Helped by Microsoft, the Internet community grew and attracted more companies wanting a share of the possibly profitable cake. During the following years companies were desperately trying to find a suitable economical model to generate income. The solution came with the introduction of Internet advertising, helping to divert the costs away from the consumer.

In 1997 a stepping stone for mobile computing was reached when Wireless Application Protocol (WAP) was specified by Nokia, Ericsson, Motorola and a browser company called Unwired Planet. The idea was to create a standard that would reach most end-users, and to be agreed on by service providers. WAP helped to bring Internet services to mobile phones and other wireless terminals.

Moving from the existing 2G mobile systems (i.e. GSM) to a more packet oriented approach was pursued and executed in 2000. The purpose of this advancement was to make it transparent for mobile terminals and mobile phones to be connected to the Internet, General Packet Radio Service (GPRS) was born.

To offer the ever growing number of mobile phone subscribers a more flexible usage including telephony, paging, messaging, Internet and broadband data a standardisation for a third generation system was started by the International Telecommunication Union (ITU). The European Telecommunications Standards Institute (ETSI) was responsible for the UMTS standardisation process in Europe, and in 1998 the Third Generation Partnership Project (3GPP) was formed to continue the work of the two predecessors. With packet oriented data (i.e. Internet) in mind, the existing GPRS system was used for that task. [34]

Due to careless delegation of Internet addresses back when IPv4 was first launched and with the unexpected explosion of number of users connected, the amount of available addresses is rapidly running low. To solve this problem a lot of work has been spent on a new version of the Internet Protocol. The result of this work is a standard describing the Internet Protocol version 6 (IPv6), dealing not only with the increased address space, but also including built-in security and authentication, simplified routing architecture and additional network management features, to name a few.

As the number of mobile subscribers and wireless Internet devices increases, the requirement for internet addresses grows. With a vision of a future where all users will be “always connected, always online”, a new addressing scheme for Internet hosts needs to be considered. This has led to 3GPP’s decision to mandate IPv6 for all new services.

Chapter 4

The role of an Internet Protocol

The basic role of an Internet Protocol (IP) is to provide the necessary mechanisms to interconnect hosts in or across several networks. In order for this to operate properly, IP needs to be implemented in all intermediate nodes and both end systems.

Whenever a source node wants to send data to a destination node across a network, the data is passed from higher layers (application, transport etc) down to the IP layer for encapsulation. The Protocol Data Unit (PDU) is than sent down to lower layers for transmission on the physical link. Figure 4.1 shows a schematic view of a typical network.

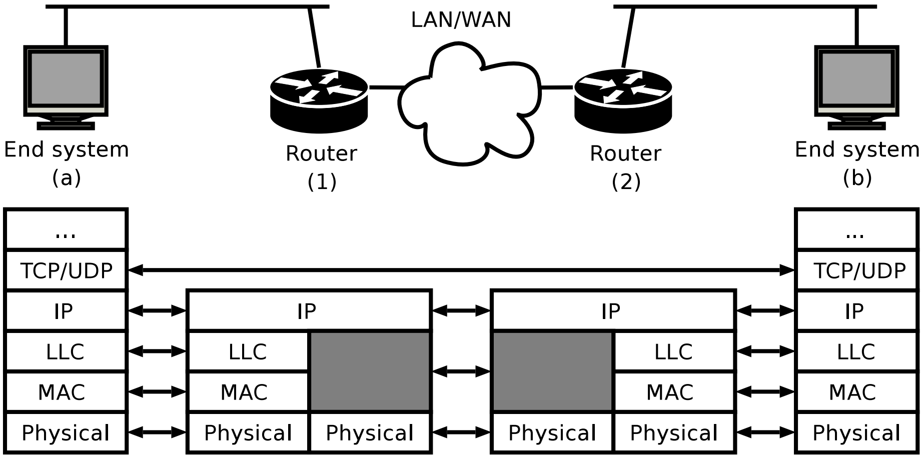


Figure 4.1: Protocol architecture overview

Consider the case when an application in a source node (a) wants to send data to a corresponding application in a destination node (b). The IP layer in node (a) receives data from upper layers (e.g. TCP or UDP) to be transmitted and attaches a header that states the destination IP address of node (b). After this a lookup is made in the local routing table of (a) for the destination network identifier. A route is found recognizing that the destination is on another subnetwork. Next, the packet is sent (using lower layers) to the router looked up in the routing table, here (1).

When the packet reaches (1), the lower layer headers are stripped off and again processed by the IP layer on the way up. Here the IP header is analyzed to determine the final destination of the packet. Based on the content of the routing table, and the destination of the packet, two possibilities open up

- (b) is determined to be on the same subnetwork as the router itself.
- (b) is determined to be one or more networks away.

Here, the destination node (b) is not directly connected to router (1), resulting in the packet being sent to router (2). When received in router (2) the same procedure is followed determining that the destination node (b) is on the same subnetwork as the router. The packet is then sent to the destination node (b), which strips the headers off and delivers the data to the application.

IP is an unreliable service: packets sent are not guaranteed to be delivered and the delivered data is not guaranteed to arrive in proper order. To achieve this supplementary support need to be implemented at higher layers (i.e. TCP). Due to the unreliable nature of IP with no guaranteed delivery order, successive packets may take different paths from source to destination. This enables the protocol to adapt to changing network conditions such as congestion and change of routes.

Chapter 5

Brief background of TCP/IP

The year was 1969 and ARPANET had begun its operation thanks to research funded by Defense Advanced Research Projects Agency (DARPA) for the US Department of Defense (DoD). To aid network communication, programmers quickly developed a protocol called Network Control Protocol (NCP). It soon became clear that this host-to-host protocol was inadequate for the types of networks in use due to performance, usability and financial reasons. The protocol was complex and became very expensive to implement, which in the end led to ambiguous implementations. As a consequence development on a new host-to-host protocol was started in 1973. The result of this long and hard effort were two new protocols called Internet Protocol (IP) and Transmission Control Protocol (TCP). [11]

These new protocols allowed hosts across several networks to share a common communication environment. The DoD quickly adopted the two protocols for use in all their packet driven networks. With this genuine support from the US government, a significant step towards a homogeneous Internet was taken. [23, 11]

A plan was then laid out on how to complete the switch from NCP to TCP/IP. This plan included the use of relay hosts, that implemented both NCP and TCP/IP. During the transition these hosts supported Telnet, FTP and mail services to bridge the two worlds together. As more hosts with the new protocol duo got connected, the old NCP protocol soon became obsolete, fully completing the switch. [11, 18]

The currently used version of IP has a 32-bit address field organized in a simple two-level hierarchy; network and host numbers as shown in figure 5.1. To make

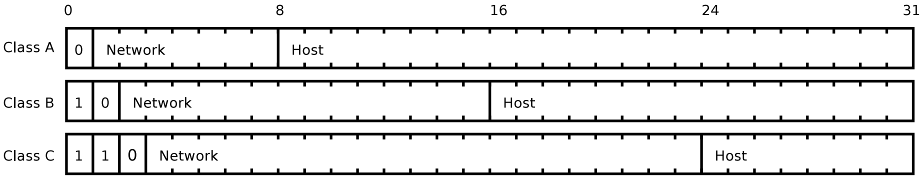


Figure 5.1: IPv4 addressing

the best use of the available 32 bits, three kinds of unicast address classes have been set up

- Class A addresses consists of a 7-bit network number, with a 24-bit host number. This address class is intended for use by the largest organizations in the world, each possible to assign 2^{24} (16,777,216) unique hosts.
- Class B addresses consists of a 14-bit network number, with a 16-bit host number. Within this address class a total number of 2^{16} (65,536) hosts can be assigned.
- Class C addresses consists of a 21-bit network number, with a 8-bit host number. Each Class C network can uniquely assign 2^8 (256) addresses.

At first glance, it may seem like there would be more than enough addresses to go around. The problem is not that there aren't enough address bits, its merely how the bits are grouped in the simple two-level hierarchy. In the early stages of the Internet, a company with more than 256 computers would apply for a Class B address, allocating more than 64 thousand addresses. Luckily this issue was discovered a couple of years ago, with drastic measures as a result. [2]

The temporary solution to extend the 32 bit address lifetime until the adaptation of IPv6, is a technique called Classless Inter Domain Routing (CIDR), also known as supernetting. The idea is the exact opposite of subnetting. Instead of borrowing bits from the host part of the address to increase the number of bits used for the network part, bits are borrowed from the network part to use for the host part of the address. This increasing the number of available addresses within a single network. As seen in figure 5.2, the number of hosts in a network segment increase by the corresponding value of the bits borrowed. In this example the single bit shift result in an increase from 2^8 (256) to 2^9 (512) available addresses. [2]

Using this technique several consecutive Class C networks may be aggregated to build mini Class B networks. [2]

IP packets are routed through a IPv4 network by first looking up the destination network in the routing table and then forwarding it out the corresponding

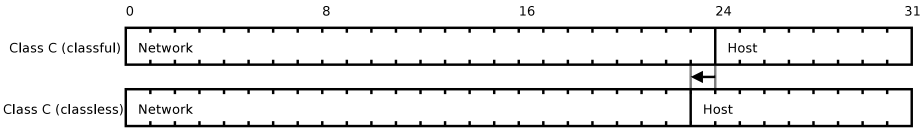


Figure 5.2: CIDR example

interface. Back when the Internet was launched, due to the restricted number of hosts, it was possible for every router to maintain a list of all available networks. In today's Internet the routers would have to maintain millions of paths to all Class C networks, in other words an impossibility. This is another aspect where CIDR has come to play an important role with its aggregation scheme, keeping the routing tables of routers world wide in manageable sizes. [17]

With CIDR being able to buy the Internet community valuable time before the 32 bit addressing scheme runs out, drastic measures need to be taken to ensure that the number of available Internet addresses never comes to a zero. [2]

Chapter 6

Introduction to UMTS

Mobility is changing and an increasing demand for global mobility can be seen in the wide range of services available including telephony, paging, messaging, Internet and broadband data. To satisfy the increasing demands, a standardisation process for a third generation mobile system (3G) was initiated by the International Telecommunication Union (ITU), referred to as International Mobile Telecommunications 2000 (IMT-2000). The goal of the new mobile system was to address the issues of the previous GSM system with respect to the new type of services demanded. [33]

In Europe, the European Telecommunications Standards Institute (ETSI) was responsible for the Universal Mobile Telecommunications System (UMTS) standardisation process, releasing the initial documents. Later, in 1998 the Third Generation Partnership Project (3GPP) was formed to continue the technical specification work from ETSI. Several work groups were formed to cope with the separate parts of the system.

Among the main driving forces behind a third generation mobile system is an increasing demand for wireless services with more capacity, more Internet and multimedia services, audio and video on demand, the desire to access data anytime and anywhere, and finally the need for telecommunication services in developing countries.

The most important requirements to be fulfilled by UMTS is the introduction of an open interface standard to enable interworking of equipment produced by different manufacturers. Furthermore the goals of seamless global and internetwork roaming and the support for both symmetric and asymmetric data need to be addressed. In addition there need to exist an evolution plan for the core network architecture used in GSM to allow current GSM operators to protect their infrastructure investments during the upgrade to UMTS. For the transition from GSM to UMTS to be successful, a contemporary coexistence and seamless roaming has to be worked out.

The following sections provides a brief overview of the UMTS architecture, followed by a review of the Packet Data Protocol (PDP) context. The section is concluded with a more in-depth look at the key parts for this thesis, i.e. the core network with respect to the Gateway GPRS Support Node (GGSN).

6.1 UMTS Architecture

A UMTS network can be separated into three interacting domains, Core Network (CN), UMTS Terrestrial Radio Access Network (UTRAN) and User Equipment (UE).

This section will describe the fundamental tasks of these domains, highlighting the enhancements from GSM. A more in-depth review of the CN is available in section 6.2.

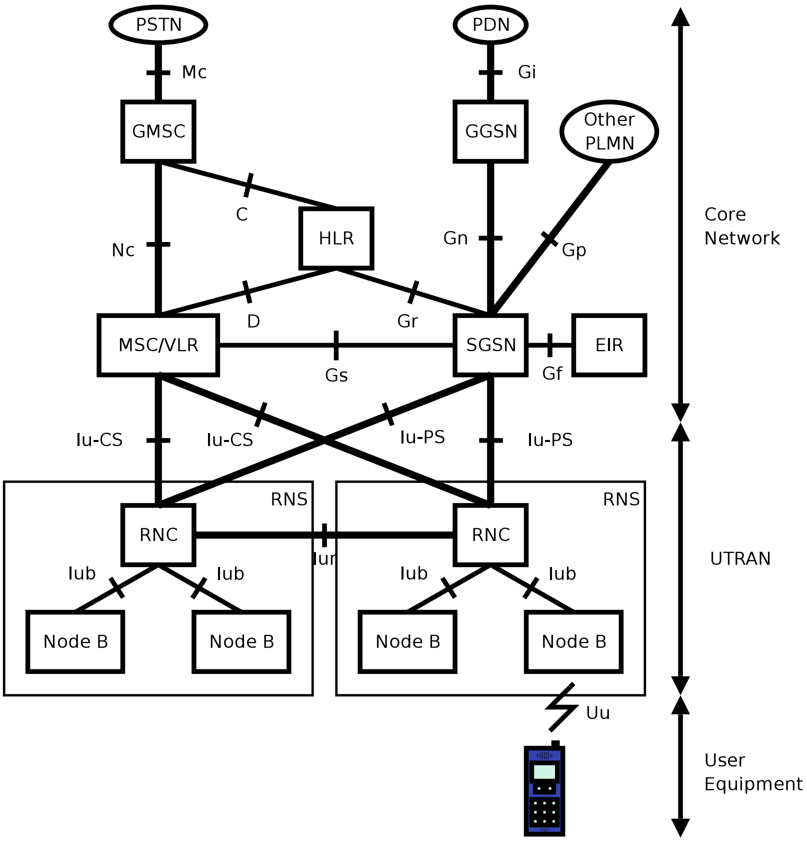


Figure 6.1: UMTS architectural overview

As seen in figure 6.1 the UTRAN is connected via the I_u to the respective CN domain, I_u-PS for packet switched data (i.e. Internet, intranet etc), and I_u-CS for circuit switched data (i.e. telephony, messaging etc). The UMTS UE is connected to the network via the U_u UMTS air interface with support for data transfer rates of up to almost 2 Mbps, increasing the previous GSM GPRS data rate with almost a factor ten.

UMTS Terrestrial Radio Access Network

The UMTS Terrestrial Radio Access Network (UTRAN) is a WCDMA radio interface for land-based communications, and is the main topic of the Rel '99 UMTS specification. UTRAN is an extension to the GSM RAN and supports two duplex operating modes, Time Division Duplex (TDD) and Frequency Division Duplex (FDD).

WCDMA uses code multiplexing, where the user data is multiplied with orthogonal spreading codes to separate the channels and to prevent interference. Careful cell planning and power control have to be used to manage the signal to interference ratio. In general the power limitation in the UE combined with the amount of interference determines the coverage of a cell. While WCDMA is significantly more complex than TDMA and FDMA, it is also significantly more spectral efficient, making it attractive for future mobile architectures. [33]

Two new network elements are introduced, these are Radio Network Controller (RNC) and Node B. The UTRAN is divided into individual Radio Network Systems (RNS). Each RNS is controlled by an RNC which is connected to a set of Node B elements.

The RNC will eventually replace the Base Station Controller (BSC) of the GSM network, while the functionality of a Node B is comparable to that of the Base Transceiver Station. UMTS defines four new interfaces for the UTRAN

- U_u : UE to Node B (WCDMA air interface)
- I_u : RNC to SGSN (Iu-CS for circuit switched data and Iu-PS for packet switched data
- I_{ub} : RNC to Node B
- I_{ur} : RNC to RNC

User Equipment

The UMTS User Equipment (UE) is based on the same fundamental principles as the GSM Mobile Station (MS), separating the Mobile Equipment (ME) from the UMTS Subscriber Identity Module (USIM). The information kept in the USIM enables the user to subscribe and enable access to the UMTS mobile network of the subscribed operators network. Another separation of the UE is the case of TE (Terminal Equipment) and MT (Mobile Terminal), where the TE could be a laptop or a PDA and the MT would be the actual mobile phone.

Three operating modes have been defined for the UE and Mobile Terminal (MT) depending on the services required, Packet Switched (PS) and Circuit Switched (CS)

- PS mode : The UE is restricted to services only within the PS domain. CS characteristic services such as Voice over IP (VoIP) however, is not restricted.
- CS mode : The UE may only use services within the CS domain, i.e. telephony.
- PS / CS mode : The UE may use services from both the PS and the CS domain.

6.2 Core Network

The Core Network infrastructure is based on that of GSM with GPRS functionality, preserving the fundamental ideas. The reason for this is to simplify the upgrading of existing GPRS core networks to support 3G. Due to the two service types (PS and CS) of the UE, the core network is divided into a PS and a CS domain.

The packet switched domain adopts two new network elements compared to GSM, the Serving GPRS Support Node (SGSN) and the Gateway GPRS Support Node (GGSN). Both of these nodes act as relay nodes for user data traffic between the UE and the external Packet Data Network (PDN). Communication between the SGSN and the GGSN is accomplished by the means of tunneling using the GPRS Tunneling Protocol (GTP) over the G_n interface. The G_i interface connects an external PDN with the CN through the GGSN.

The circuit switched domain include the Mobile Switching Center (MSC), Visitor Location Register (VLR) and Gateway MSC (GMSC).

The two roles of IP

IP is deployed in the mobile CN for two distinct purposes. Figure 6.2 shows an end-to-end protocol stack for an IPv6 application running over a UMTS packet-switched network where IPv4 is the network protocol.

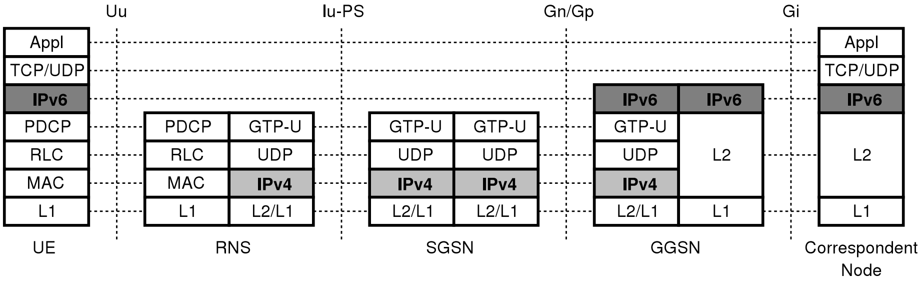


Figure 6.2: UMTS end-to-end protocol architecture

The lower layer signifies the IP transport layer (light grey), used in the CN. This layer transports user and control data between the entities within the mobile network and is only significant herein. The transport IP layer is terminated at the UTRAN on one end, and the GGSN before leaving the Public Land Mobile Network (PLMN), on the other.

The upper layer signifies the IP application layer (dark grey) which connects the UE with an external node. This is the typical layer seen in for instance, today's Internet.

These two levels of transport IP layers are independent of each other, leaving it optional to deploy for instance IPv4 in the lower transport layer, whilst using IPv6 in the application transport layer.

6.3 PDP Context

In order for the user to transmit data, a Packet Data Protocol (PDP) context must be activated in the UE, the SGSN and finally the GGSN. This procedure is initiated by the user and is similar to logging on to the destination network. Figure 6.3 shows the steps taken during the PDP Context Activation phase.

Once this procedure is completed a virtual tunnel is created between the UE and the GGSN. Data transfer may now be issued between the UE and the external network connected to the GGSN. The virtual tunnel activated between the UE and the GGSN consists of two parts, first the PDCP protocol between the UE and the RNS, and second the GTP protocol between the RNS and the two GSNs. This procedure can be seen in figure 6.2. [1]

The GTP used in a UMTS network is a variant of the GPRS GTP protocol. This protocol is used between GSN nodes and is defined for both the G_n interface (i.e. between GSNs within a PLMN) and the G_p (i.e. between GSNs belonging to different PLMNs). GTP is encapsulated within a UDP datagram and allows

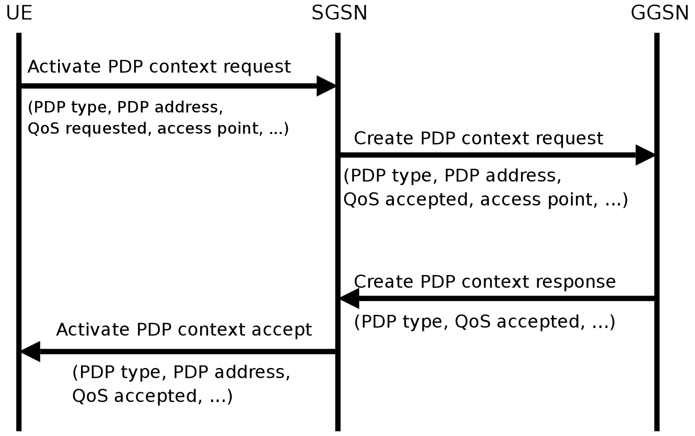
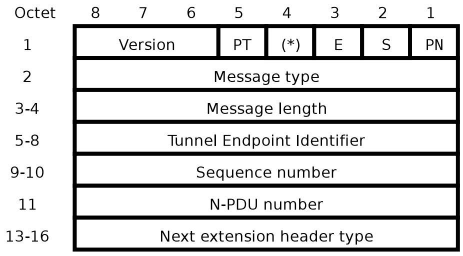

UE
SGSN
GGSN
Activate PDP context request
(PDP type, PDP address,
QoS requested, access point, ...)
Create PDP context request
(PDP type, PDP address,
QoS accepted, access point, ...)
Create PDP context response
(PDP type, QoS accepted, ...)
Activate PDP context accept
(PDP type, PDP address,
QoS accepted, ...)

Figure 6.3: PDP Context Activation procedure

multiprotocol packets to be tunneled through the UMTS CN. An SGSN may provide service to many GGSNs, and a single GGSN may be associated with many SGSNs, hence a many-to-many relationship is assumed.

Two operation modes has been specified for GTP, the control (signaling) and the data (transmission) plane. In the control plane, GTP specifies a tunnel management protocol allowing the SGSN to provide network access for a UE including tunnel creation, modification and deletion. In the data plane GTP uses a tunneling mechanism for carrying user data packets through the tunnel previously set up by using the control plane. [1]

The GTP header is a fixed size header used by all GTP messages in both control and data plane. The headers is shown in figure 6.4. [1]



Octet	8	7	6	5	4	3	2	1
1	Version			PT	(*)	E	S	PN
2	Message type							
3-4	Message length							
5-8	Tunnel Endpoint Identifier							
9-10	Sequence number							
11	N-PDU number							
13-16	Next extension header type							

Figure 6.4: Outline of the GTP header

- Version : This field determines the version of the GTP protocol.
- Protocol Type (PT) : Used as a protocol discriminator between GTP and GTP' (not covered here).

- Extension header flag (E) : Used to flag the presence of a meaningful value of the Next extension header field. When it is set to '0', the value of the Next extension header field should not be interpreted.

- Sequence number flag (s) : Indicates the presence of a meaningful value of the Sequence number field. When it is set to '0', the value of the Sequence number field should not be interpreted.

- N-PDU number flag (PN) : Indicates the presence of a meaningful value of the N-PDU number field. When it is set to '0', the value of the N-PDU number field should not be interpreted.

- Message type : Indicates the type of GTP message to follow.

- Length : Indicates the length in octets of the payload, i.e. the remaining packet following the mandatory part of the GTP header (i.e. first eight octets). The Sequence number, the N-PDU number and any Extension headers should be considered a part of the payload and hence be included in the length count.

- Tunnel Endpoint Identifier (TEID) : Identifies a tunnel endpoint in the receiving GTP-U or GTP-C end. This value is locally assigned by the receiving side of a GTP tunnel and is exchanged between the endpoints using GTP-C.

- Sequence number (**optional**) : Contains a transaction identity in the form of a serial number for transmitted G-PDU packets. When transmission order must be preserved in the user plane, an increasing serial number is used.

- N-PDU number (**optional**) : Field used in some handover procedures, for instance between 2G and 3G radio access networks.

- Next extension header type (**optional**) : This field determines the type of extension header that follows.

Special attention need to be taken when using IPv6 due to the optional use of either stateless or stateful address configuration. The selection between the two is prescribed by the Router Advertisement sent by the GGSN. Stateless address configuration is mandatory, whilst stateful is left optional. [1]

When using IPv6 address auto configuration (either stateless of stateful), the process is setting up the access to an external network involves two signaling phases. The first phase is conducted in the control plane and consists of the PDP context activation, followed by a second phase performed in the user plane consisting of an address configuration procedure (stateless or stateful). For every PDP context, precisely one prefix should be used. [1]

The stateless configuration procedure include only the participation of the UE and the GGSN, whilst the stateful configuration procedure also involves an external address pool, such as a Dynamic Host Configuration Protocol (DHCPv6) server and participation of the SGSN. The SGSN is does charging and needs to know the actual address occupied by the UE. [1]

Regardless of the type requested, the auto configuration procedure is configured per Access Point Name (APN) in the GGSN, i.e. addresses should be used accordingly. When using stateless configuration, the GGSN should only use the prefix part of the IPv6 address when forwarding packets to and from the mobile, this since the mobile itself is responsible for setting the interface identifier. [1]

6.4 Gateway GPRS Support Node

The Gateway GPRS Support Node (GGSN) is responsible for interconnecting a GPRS network via the SGSN, with an external PDN (i.e. the Internet or a corporate intranet). The GGSN delivers the packets from the UE, which previously has been relayed by the SGSN, to the external PDN. In the opposite direction, all traffic from the external data network is delivered to the UE. From the external PDN, the GGSN is seen as a router for all the UE attached to it.

The main requirements of a GGSN in a GPRS based network is to

- Manage connections to external PDN, including the Internet and corporate intranets.
- Perform user authentication, authorization and accounting.
- Perform data tunneling and security functionality.
- Manage the assignment of IP addresses to the users.
- Provide the data relay service with Quality of Service (QoS) parameters, such as rate and delay variations.
- Transparently act as a router to external networks such as the Internet or intranets.
- Generate charging information based on the amount of traffic forwarded.

The GGSN consists of two main interfaces between which data is being forwarded. The G_n interface is faced in the direction of the CN, interconnecting the GGSN with the SGSN. Traffic flowing between an external PDN and the GGSN passes through the G_i interface. These two interfaces may be of any type ranging from ATM to Ethernet.

6.5 Routing examples

In this section the two common cases of incoming and outgoing packets will be shown with simple examples. In both cases, the UE uses the IPv6 address 3ffe:ab32::2 and the Internet node (for instance a web server) uses the IPv6 address 3ffe:0510::efab.

This example assumes that there exist a GTP tunnel between the SGSN and the GGSN, and that all necessary signaling has been performed (i.e. there exist a PDP context for the UE).

Outgoing IP packet

Figure 6.5 shows the necessary steps taken to send a packet from a UE to a node on an external packet network, in this case the Internet. First, an address conversion is performed between the IPv6 source address and an NSAPI. The packet is sent over the air interface using the PDCP protocol and is then received by the SGSN. The SGSN performs another address conversion between the NSAPI to get a TEID and a GGSN IPv4 address to which the packet is then forwarded to. Upon the reception of the GTP packet by the GGSN on the G_n interface, the remaining IPv6 packet is then transmitted to the external PDN where it is processed accordingly.

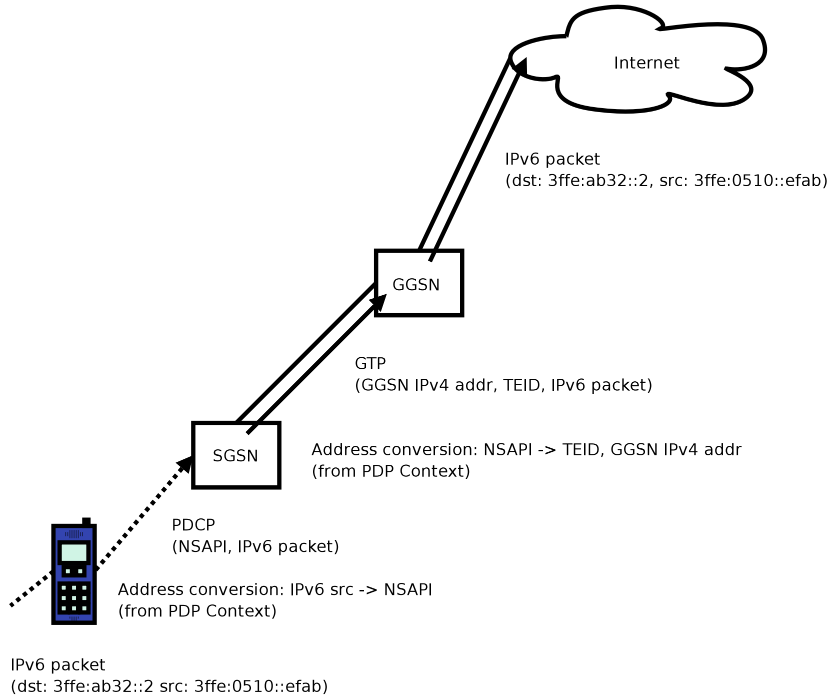

Internet
IPv6 packet
(dst: 3ffe:ab32::2, src: 3ffe:0510::efab)
GGSN
GTP
(GGSN IPv4 addr, TEID, IPv6 packet)
SGSN
Address conversion: NSAPI -> TEID, GGSN IPv4 addr
(from PDP Context)
PDCP
(NSAPI, IPv6 packet)
Address conversion: IPv6 src -> NSAPI
(from PDP Context)
IPv6 packet
(dst: 3ffe:ab32::2 src: 3ffe:0510::efab)

Figure 6.5: Routing example: Outgoing IP packet

Incoming IP packet

When an incoming packet is received by the GGSN on the G_i interface, a GTP header is added containing the TEID corresponding to the IPv6 destination address. Then the packet is forwarded to the appropriate SGSN provided by the lookup from the IPv6 destination address. As the SGSN receives the GTP packet and address conversion from the provided TEID to an NSAPI is conducted, and the packet is forwarded to the requested UE over the air interface using the PDCP protocol. Figure 6.6 shows the course of events when an incoming packets is processed by the network.

6.6 Conclusion

UMTS can be seen as an evolution of GSM with a new air interface, accompanied by a packet oriented IP centric CN. The capabilities of UMTS provide a whole new set of services, including multimedia and quality of service options. Backward compatibility and handover with GSM ensure a coexistence of the two generations for many years, facilitating the complete transition.

In order for packet transmission to be initiated, a PDP context must first be

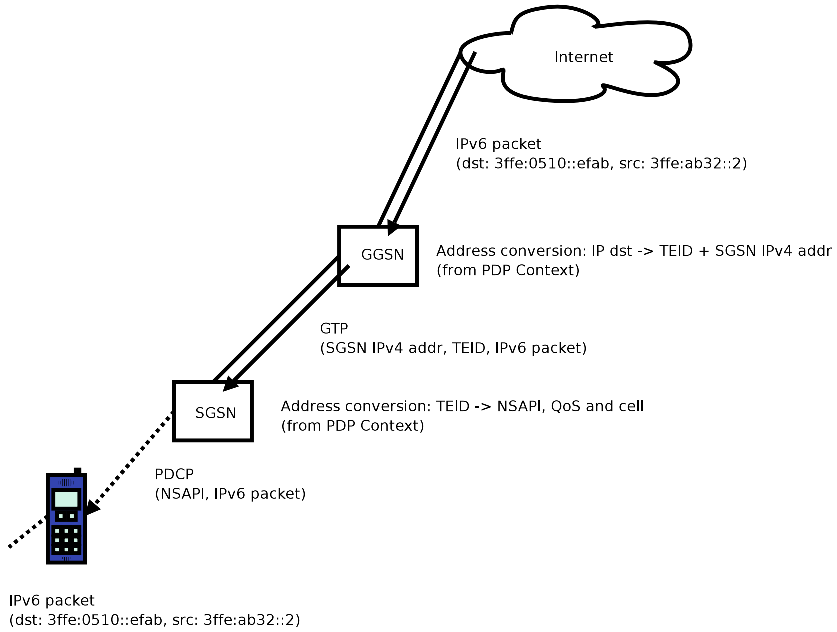

Internet
IPv6 packet
(dst: 3ffe:0510::efab, src: 3ffe:ab32::2)
GGSN
Address conversion: IP dst -> TEID + SGSN IPv4 addr
(from PDP Context)
GTP
(SGSN IPv4 addr, TEID, IPv6 packet)
SGSN
Address conversion: TEID -> NSAPI, QoS and cell
(from PDP Context)
PDCP
(NSAPI, IPv6 packet)
IPv6 packet
(dst: 3ffe:0510::efab, src: 3ffe:ab32::2)

Figure 6.6: Routing example: Incoming IP packet

set up in the control plane. After the PDP context successfully has been established, an address configuration phase follows in the user plane. This phase may either be stateless or stateful. In the case of the mandatory stateless address configuration, a prefix is given to the mobile by the GGSN, leaving the mobile to generate the full address.

The GGSN is responsible for connecting a UE with an external packet data network. Data is being transmitted by the means of tunneling using GTP, operating on any UDP/IP based transport architecture. Other duties of the GGSN include those of authentication, IP address assignment and to generate charging information.

Chapter 7

Introduction to IPv6

With the ever growing Internet gaining popularity worldwide, it became evident that the current version of IP was inadequate to meet future demands. The Internet Engineering Task Force (IETF) noticed the need for improvements and initiated a working group to work out a proposal for a new Internet Protocol, IP next-generation (IPng).

The main driving force behind a new protocol was the need for an increased address space, being an increasingly limiting resource. With this in mind, several proposals were published and a final design for IPng emerged in 1994. The publication of RFC 1752: "The Recommendation for the IP Next Generation Protocol", was the first step towards a new standard. This document describes requirements and recommendations on an upcoming Internet Protocol. Also included in the document is a transition plan to the new IPng architecture, gradually phasing out IPv4. Several other concerns are discussed and can be summarized within the following topics

- Addressing
- Performance
- Network services
- Security
- Transition mechanism

Each of these topics are discussed in appendix A, providing the background thoughts and solutions adopted in IPng. The Internet Assigned Numbers Authority (IANA) assigned version number 6 to IPng, hence the protocol is called IPv6. [3]

IPv6 was designed as an evolution of its precursor, keeping functions generally seen as working in IPv4. Less frequently used or poorly working functions were left out or made optional. New functionality was then added where needed, never loosing focus on design simplicity.

Key features of IPv6 include

- New addressing and routing architecture : Increased address size from 32 bits to 128 bits, dramatically increasing the number of addressable nodes. The new addressing architecture also offers more levels of hierarchical routing and simpler auto configuration of addresses.
- Powerful transition mechanisms from IPv4 to IPv6 : In order to deploy the new addressing architecture, a transition plan has been laid out.
- Simplified header format : Several IPv4 header fields have either been dropped or made optional to reduce processing costs and lower packet overhead in the common routing case. Even though the IPv6 addresses are four times as long as the IPv4 addresses, the IPv6 header is only twice the length of the IPv4 header. As opposed to IPv4, IPv6 uses a fixed length header to simplify processing.
- Separation of header and options into extension headers : Optional fields are placed in separate headers following the IPv6 header, before the transport layer header. This resulting in a substantial performance increase since most options are never processed by intermediate routers.
- Auto configuration : Several mechanisms are supported for automatic address configuration, either stateless configuration or stateful configuration using Dynamic Host Configuration Protocol (DHCP).
- Quality of Service : By labelling traffic between two endpoints, traffic engineering and sender-specific requirements can be met.
- Authentication and privacy : IPv6 provides support for authentication, privacy and data integrity. This enables secure communication with all types of applications and overlying protocols, regardless of their security awareness.
- Improved Mobile IP : A powerful mechanism to allow hosts to seamlessly move between subnetworks maintaining IP connectivity.

Chapter 8

Overview of IPv6 over UMTS

8.1 Introduction

A vision popular to service providers and networks operators worldwide is the paradigm of an "Always Connected, Always Online" user. A result of this paradigm would be a whole set of new services, such as paging, positioning and others.

The explosive increase in the number of Internet users combined with the ever increasing popularity for wireless Internet devices, has lead to the demand for a dynamic IP technology to cope with the growth. This together with new services such as IP multimedia that assume globally unique addressing, makes IPv6 an attractive candidate as future mobile IP technology. [22]

The nature of wireless networks sets high requirements in terms of scalability, quality of service and security, all of which are addressed by IPv6. Some of the most attractive features in the IPv6 architecture that make it suitable for an IP based wireless network are

- Greatly improved address space
- Simplified routing architecture
- Security functions
- Quality of Service (QoS) capabilities
- Mobility support
- Network management features

On the other hand, it should be noted that IPv6 yields some extra overhead compared to IPv4, which may be noticeable over extremely lossy wireless connections. However, mechanisms to deal with this, such as header compression, have been evaluated and implemented in IPv6. [2]

The following sections will investigate the various aspects of IPv6 in a UMTS network with respect to addressing, security, migration, mobility, network management and finally its suitability for operating over an air interface.

8.2 Addressing

All devices connected to the Internet need a globally unique address, a problem that is becoming increasingly more evident with the explosion in the number of hosts such as Personal Digital Assistants (PDA), laptops, digital cameras etc. The usable lifetime of IPv4 has significantly been extended by using a technology called Network Address Translation (NAT), a mechanism that allows enterprises to deploy large networks using a shared address space.

Hence, NAT provides an effective way to allow more nodes to participate in today's Internet, than would be possible if all nodes required a routeable unique address. This features comes with a price of lost key functionality. As the Internet grows, several new advanced applications have begun to arise, requiring end-to-end connectivity and real time services. A selection of these applications include peer-to-peer solutions such as real-time audio and video, file sharing and games.

By increasing the number of addressing bits in the header, IPv6 provides a straightforward solution to the above problems by enabling each host connected to the Internet to have a unique address. One of the key results of the increased address space is the true end-to-end connectivity, where each connected device can exchange data without intermediate manipulation (i.e. NAT).

8.3 Routing

Today's IPv4 provides a flat addressing hierarchy, where routers on all networks, regardless of its size, has to contain information on all other networks, resulting in huge routing tables. This problem has been somewhat helped in IPv4 using a technique called Classless Inter Domain Routing (CIDR).

IPv6 solves this problem by using a address hierarchy, instead of the flat structure of IPv4. Here, addresses are gathered in logical groups aiding the process of routing a packet. Routing is hereby delegated, leaving smaller routing tables even in the core of the Internet, hence increasing performance. With the large amount of potential customers, the delegated routing helps keeping routing tables in UMTS core networks to a minimum. [19]

8.4 Security

Since IPsec is left as optional for IPv4, it hasn't been widely implemented or adopted by software developers. This has led to the majority of today's Internet traffic being transferred in clear text, unless application level security such as SSH[25] or SSL[32] has been applied.

IPsec is mandatory in the IPv6 specification, guaranteeing its availability on all networking platforms including mobile phones, PDAs and other platforms currently with little or no security mechanism.

8.5 Quality of Service

The Internet today, offers packet delivery on a first-come, first-served basis, using equal priority for all packets. E-mail is delivered with the same priority as real-time applications such as streaming audio and video.

With IPv6 it's possible to provide QoS by using the standard header for packet labeling and include priority control in intermediate routers. The QoS capabilities are not specified in detail, and are instead left to the service provider to make best use of.

Recent discussions within IETF concentrate on the usage of the flow label to accomplish QoS support by using for instance RSVP (Resource Reservation Protocol). [28]

With the IP centric approach of the UMTS core networks, IPv6 may be used as an aid to provide traffic classification herein, enabling flow based priority controlled routing.

8.6 Mobility

Traditionally, the Internet was a collection of stationary nodes where address assignment was performed on a less frequent basis. Today, as the Internet evolves, the concept of "mobile computing" is becoming more evident, including devices such as mobile phones, PDAs, laptops and more. [18]

Mobile IP was designed to provide mobile devices with a mechanism to enable a continuous service while moving between networks. Consider the case of participating in a video conference call while riding a train across the country connected with an access point in the train. Most likely, you'll be passing through several network segments as the train moves, disrupting any in-flight datagrams. Mobile IP provides a solution for this by using the concept of "foreign agents", to which packets are forwarded from your "home network".

A technique called Movement Detection is used to determine if a new care-of address is required. The detection is conducted by using a combination of methods, such as monitoring for new prefixes, hints from physical and lower layer protocols, or by monitoring TCP acknowledgments etc.

Mobile IP exists for IPv4, but contains a number of disadvantages to its successor in IPv6, most of which are inherent features of IPv6 itself. As the case with IPsec, Mobile IP is mandatory in the IPv6 specification, guaranteeing world wide deployment.

IPv6 provide a mechanism to eliminate the problem of triangle routing, a problem evident in IPv4 mobile networks. The problem arises when incoming packets to a mobile node are being routed through the home agent, whilst outgoing packets are routed straight to the correspondent node. Due to the enormous availability of IPv6 addresses, Mobile IPv6 provides a topologically correct address when moving to a new network. This eliminates the need for mobility management functions such as the foreign agent, present in Mobile IPv4. More information on the mobility support in IPv6 can be found in appendix A.3.

Mobile IP in 3GPP networks is only used for inter-system mobility, i.e. mobility between e.g. UMTS and 802.11 or HIPERLAN. A layer two mobility mechanism is built into the UMTS/GPRS CN to handle intra-system mobility, i.e. mobility within a UMTS network.

Chapter 9

Implementation

The GGSN emulator is written in the C programming language under the FreeBSD[31] operating system. Several tools and third-party libraries have been used to maintain portability, first and foremost between the members of the BSD family, FreeBSD, OpenBSD[24] and NetBSD[30]. These tools and libraries are introduced and discussed in section 9.4.

9.1 Requirements

The goal of the resulting implementation is a basic 3G GGSN node emulator capable of forwarding IPv6 packets. The emulator is to be used in a closed testing environment for end-to-end testing of IPv6 terminals and be compatible with existing SGSN services.

The GTP (GPRS Tunneling Protocol) specification[1] defines more than 40 control messages, some of which are necessary to the implementation. Due to the large amount of messages in the specification, a subset has been selected as mandatory and further some as optional. The mandatory messages are those essential for path and tunnel management such as context setup and tear down. Required messages for the implementation to handle correctly are

- Echo Request : This message is used to find out if the peer GSN is alive and may be sent by both the SGSN and the GGSN.
- Create PDP Context Request : This message is sent by a SGSN to a GGSN as a part of the GPRS PDP Context Activation procedure. Contains information on connection setup.
- Update PDP Context Request : This message is sent by a SGSN to a GGSN to update information previously set by a corresponding Create

PDP Context Request. Used for connection management on already set up connections.

- Delete PDP Context Request : This message is send by a SGSN to a GGSN as a part of the GPRS PDP Context Detach procedure. Contains information on connection tear down and should be used to deactivate an activated PDP Context.
- G-PDU : This message contains a user data message and constitutes the data plane.

A complete implementation satisfying all above requirements and the whole of the specification is a cumbersome and time consuming task. In order to have a working implementation completed within the time frame of the thesis, some restrictions on the standards were injected. Aside from the set of messages excluded, mechanisms such as QoS may if necessary be left as future work.

The layer two protocol to be used on both the G_n and the G_i interface is standard Ethernet. In order to perform careful testing, a working SGSN with IPv6 support is assumed during development.

9.2 Design

The GGSN emulator is designed with the 3GPP recommendations as the starting point. The standard recommends separating the control and user plane into separate blocks named control and user block. To further evolve the modular design, several other blocks have been specified to each handle a well defined task. These blocks are GTP encoder and decoder block, a Router Advertisement (RA) block, a prefix pool and finally a forwarding table. Figure 9.1 shows a block diagram of the design.

Communication between the blocks is conducted by the means of two asynchronous queues for input and output respectively. Upon receiving a packet on the G_n, the data is placed in the input queue and the socket call immediately returns. The input queue is processed by a separate thread and appropriate operations for each message are performed. If the operations result in a response, this message is placed in the output queue and the processing of the input queue continues. A second thread is monitoring the output queue for activity and processes any entries by transmitting the data out on either the G_n or the G_i.

The control block is responsible for signaling and control setup and tear down. This information is written to a forwarding table that is then used by the user block to determine whether the datagram should be forwarded or discarded. The user block also uses the RA block to trigger RA being requested through

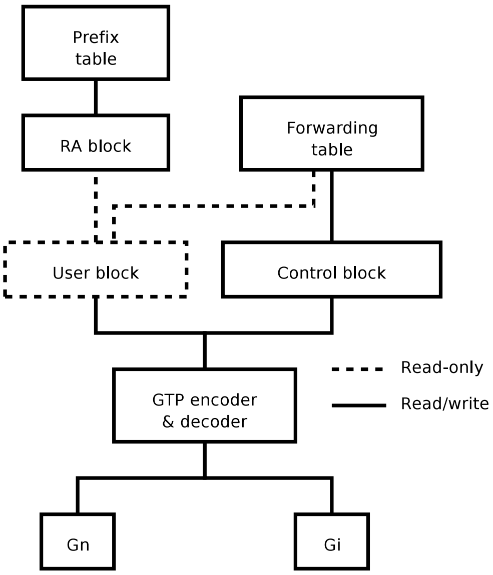


Figure 9.1: Block diagram

a Router Solicitation (RS) message. The RA block itself is responsible for the periodical (timer based) RA transmissions, by using a timer in the forwarding table. The prefixes included in the RA messages are taken from the prefix pool, that holds all prefixes delegated to the GGSN.

To correctly validate, encapsulate and decapsulate GTP messages, C structures are sent and returned from the GTP encoder and decoder functions.

Input and output queues

The input and output queues are asynchronous queues from the GLib[26] library package. Each entry within a queue consists of

- Payload type
- Payload length
- SGSN address
- Pointer to data

For packets either transmitted or received on the G_n interface, the port number indicates the payload type (2123 and 2152) for the input and output queue respectively. To indicate an incoming or outgoing G_i delivery, a payload type of 1, named GTP_L2 is used.

The payload type is used by the queue processing threads to determine where to transmit or how to process the attached data.

For the input queue, the following interpretation is made of the payload type

GTP-U	User data, perform table lookup ($G_n \rightarrow G_i$)
GTP-C	Control message, do operation accordingly
GTP_L2	Layer two data, perform table lookup ($G_i \rightarrow G_n$)

In a similar fashion for the output queue

GTP-U	User data, send packet to SGSN on port 2152
GTP-C	Control message, send packet to SGSN on port 2123
GTP_L2	Layer two data, inject the data on the G_i interface

Hence, the table lookup is performed on input rather than on output. If no entry is found during forwarding lookups, the data is discarded without notification.

The G_n interface

The G_n interface consists of two blocking UDP/IPv4 sockets listening on ports 2152 and 2123 for GTP-U and GTP-C respectively. Upon receiving a UDP datagram, the packet is inserted into the input queue and labeled with the respective type (GTP-U or GTP-C).

The G_i interface

The G_i interface is made up of two layer two routines, one for packet capturing and one for packet injection.

When the packet capturing routine receives an incoming ethernet frame, the payload type is examined and stored. If the type is IPv6 a lookup for the destination prefix is performed in the forwarding table. If a match is found, a GTP-U message is constructed and inserted into the output queue. If no match is found, the frame is dropped without notification.

The packet injection routine is triggered by the output queue processing thread for every outgoing GTP-U message. First an IPv6 user packet is constructed from the information. Second an ethernet header is added followed by the frame being transmitted.

GTP encoder and decoder

All messages in the GTP protocol shall be transmitted in network octet order starting with octet 1. The Most Significant Bit (MSB) of an octet in a GTP message is bit 8. If a value spans several octets, the MSB is bit 8 in the octet with lowest number. To facilitate the transition between human readable and GTP PDU form, encoder and decoder functions was created.

Given a structure containing Information Element (IE) specific data, the encoder function outputs a binary representation ready to be transferred. Conversely, given a binary represented IE, the decoder function return programming language structures easily manageable to the programmer.

Both the encoder and decoder contains error checking functionality to ensure that valid input and output data is transmitted and received.

Router Advertisement block

The Router Advertisement block is responsible for both periodical and on-demand transmission of RA messages.

The periodical transmission is managed by the RA block by using a RA counter for every occupied (i.e. used by a UE) entry in the prefix table. A RA message containing corresponding prefix is constructed and inserted into the output queue for delivery.

Whenever a RA request made by the user block, an appropriate RA with the corresponding prefix is inserted into the output queue for delivery.

Prefix pool

The GGSN maintains a pool of available prefixes that are leased to a user during the PDP Context Activation phase. Each entry in the pool consists of

- Access Point Name (APN) : A fully qualified domain name.
- Prefix : An IPv6 prefix of length 64.
- Occupied flag : Flag to identify if the prefix is currently in use.

The pool is implemented as a double linked list where each entry is made up of a static structure containing the above elements.

Forwarding table

The forwarding table can be thought of as a routing table with some additional information. Each entry in the table consists of

- Network Service Access Point Identifier (NSAPI) : Access point to the service network.
- International Mobile Subscriber Identity (IMSI) : Uniquely identifies a mobile subscriber.
- Charge ID : Used to identify all charging records produced in the SGSNs and the GGSNs for this PDP context. The Charge ID is generated by the GGSN and should be unique herein.
- End user address : Contains the address (for instance IPv4 or IPv6) of the end user.
- SGSN data & control TEID : Tunnel Endpoint Identifier for the peer SGSN, data and control plane.
- GGSN data & control TEID : Tunnel Endpoint Identifier for the local side, data and control plane.
- SGSN data & control address : IPv4 address for the SGSN, data and control plane.
- RA prefix and counter : IPv6 prefix associated with the PDP context and a counter to determine when to send next RA message. This is a reference to the corresponding entry within the prefix pool.

Upon reception of a PDP Context Activation Request message a lookup is conducted for the included NSAPI and IMSI. If a match is found, an update to the entry is made.

If no match is found, a new entry with values corresponding to the PDP Context Activation Request message is inserted into the forwarding table.

Initially the forwarding table was implemented using a hash table, but due to multiple-key constraints involving the end user address, the TEID and the IMSI, this idea was abandoned in favor of a double linked list. Both the hash table and the double linked list is a part of the GLib[26] library package.

Each entry in the forwarding table is realized by a structure containing the variables. The prefix to be announced in the RA is included as a reference to the corresponding entry in the prefix pool. The layout is shown in figure 9.2.

Control block

The control block contains the logical framework of the GGSN for connection setup and tear down. All valid incoming packets on the GTP-C port on the G_n interface are processed by the control block. First the type of the request is determined followed by a processing part for that particular type. The GTP

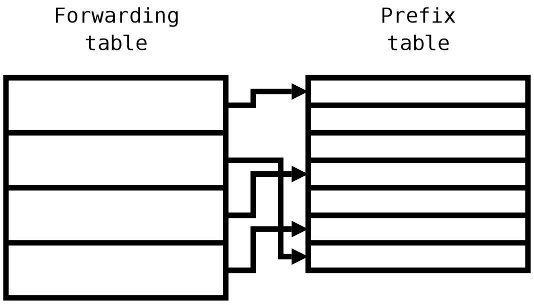


Figure 9.2: Forwarding-prefix table correlation

message types correctly handled by the implementation are Echo Request (GTP-U and GTP-C), Create PDP Context Request (GTP-C), Update PDP Context Request (GTP-C) and Delete PDP Context Request (GTP-C). An algorithmic description for the processing of the most complex of the above, the Create PDP Context Request, is provided in section 9.3.

User block

The user block is a strict logical unit, using a read-only connection to the forwarding table. When a datagram is received on the G_n interface, the TEID in the GTP header is looked up in the forwarding table. If a match is found, the remaining payload (following the GTP header) is placed in the output queue for G_i delivery, otherwise the datagram is dropped.

9.3 Algorithms

The processing of a Create PDP Context Request message is performed using the following algorithm

```
1 Perform a forwarding table lookup for the NSAPI
2 If a NSAPI match is found
    2.1 Perform a forwarding table lookup for the IMSI
    2.2 If a IMSI match is found
        2.2.1 Modify the entry found in the forwarding table to correspond to
              the values included in the Create PDP Context Request message
              and send a Update PDP Context Response to the SGSN
3 If no NSAPI match is found
```

3.1 Verify that the message include IEs for Control TEID, User TEID, NSAPI, GSN Signaling Address, GSN Data Address and QoS Profile

3.2 If any of the preceding IEs are missing

   3.2.1 Deny the Create PDP Context Request and send a Create PDP Context Response to the SGSN with the cause set accordingly

3.3 If all required IEs are included

   3.3.1 Verify that the included APN is available in the prefix pool, deny the Create PDP Context Request if missing

   3.3.2 Determine whether the user has requested a dynamic or static IP address

      3.3.2.1 If static, provide the user with the address requested in the Create PDP Context Request message. This is only valid if the address type requested is IPv4

      3.3.2.2 If dynamic, provide the user with a free address from the address pool, for the requested APN. The address type is determined by the type number in the request

   3.3.3 Insert the prefix of the above address into the forwarding table

   3.3.4 Encode the whole Create PDP Context Response message and attach a GTP header

   3.3.5 Send the encoded message on the G_n to the SGSN with the signaling address and TEID provided in the request

9.4 Tools

Programming library: GLib

GLib is a low-level programming library that provides a wide variety of well tested, portable and documented functions and definitions. These include functions for type conversions, byte ordering, memory allocation, string utilities plus a number of data structures such as lists, queues, hash tables, strings, arrays, trees etc. [26].

The GLib data structures primarily used in this implementation are double linked lists (GList), hash tables (GHashTable) and asynchronous queues (G-AsyncQueue).

Programming library: libpcap

Libpcap is a OS and architecture independent C programming interface for packet capturing. The library provides the programmer with a framework for low-level network monitoring.

From a programmer's point of view, libpcap may be operated in two different manners. The straightforward method is to use a programming loop (i.e. while() or for()) wrapped around the pcap_next() which is a blocking function that returns a pointer to the next packet when received. The other method is to use the pcap_loop() function, which takes a function pointer as an argument. The provided function is called by the library whenever a new packet is received, hence escaping the need of a separate thread for the capturing. Due to the complexity of the remaining system, the latter approach is used to keep the number of active threads at a minimum. [14]

Programming library: libnet

The libnet C programming library allows the programmer to construct and inject network packets. Libnet offers creation interfaces for both the IP (version 4 and 6) and link layer.

The construction and injection of a IPv6 packet is performed in four steps

1 libnet_init() : Initialize the device that should inject the packet, returning an output handler.

2 libnet_build_ethernet() : Build an ethernet frame containing the payload, set to type ETHERTYPE_IPV6 and attach to the output handler.

3 libnet_write() : Writes the previously created packet, identified by the output handler, to the wire.

4 libnet_destroy() : Perform cleanup of the memory allocations previously made by prior libnet function calls.

Chapter 10

Conclusions

10.1 Statement of results

The result of this Master's Thesis is a working prototype of a GGSN for 3G with support for IPv6 as user transport protocol. Initially the work was mainly concentrated around literary studies, reading and interpreting standards and recommendations from miscellaneous sources. Development continued as an iterative process, mixing coding with further studies of standards.

The main goal of completing a basic prototype for laboratory testing was achieved within the time frame of the thesis work. IPv6 router functionality such as Router Advertisement and handling of Router Solicitation was added in the final stages of the practical work in order to facilitate address configuration and help testing. End-to-end testing was possible to conduct with the finished prototype, all according to the requirements.

Implementation testing was greatly complicated due to the lack of an existing SGSN with IPv6 support. This implication led to the implementation of a SGSN-like application for generating control and data streams.

Further implications were introduced as the protocol analyzer by RADCOM[27] was unable to correctly decode IPv6 packets over GTP, even still the specification suggested it would. This issue was solved by the introduction of an open source ethernet protocol analyzer called Ethereal[7].

10.2 Limitations and future work

The initial idea for the implementation was to tightly couple the GGSN functionality with the kernel of the operating system, providing highest possible throughput. After conducting some initial tests it became clear that such an implementation would require significantly more development effort resulting in delays. Despite the certain loss of performance, the tests clearly implied that an application level implementation would be more suitable considering the time frame of the project.

The main target for future work is to complete the kernel implementation of the GGSN module by using a system called Netgraph[6]. Netgraph is a modular system for networking within a FreeBSD[31] kernel. The architecture consists of interconnected nodes, forming a graph structure. Each node performs a single well defined task which may be combined into a more complex protocol combination. The proposed name of the Netgraph kernel module is ng_gtp.

Additional supplements to the current implementation is support for a wider range of control messages and a functional Quality of Service mechanism.

Chapter 11

Acknowledgments

Special thanks to Karim El-Malki and Gianluca Verin at Ericsson, for introducing me to this thesis proposal and providing me with their invaluable suggestions, support, knowledge and ideas.

Big thanks to Thomas Nilsson at Umeå University, for his encouragement and guidance when writing this thesis.

Thanks to Erik Heneryd and Claes Mogren for their valuable paper reviews, their comments and suggestions.

Also thanks to all the people who has supported me throughout this work, you know who you are ☺.

Appendix A

Overview of IPv6

A.1 IPv6 packet format

Following the lines of simplicity and performance, mandatory fields are kept to a minimum. The only required header is referred to as the *IPv6 header* and contain sufficient information for both end systems and intermediate routers to deliver a packet through an IP network.

The general form of a IPv6 protocol data unit (PDU) has the form shown in figure A.1.

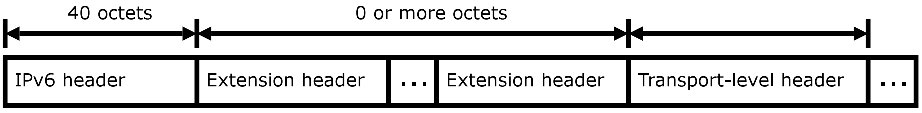


Figure A.1: IPv6 protocol data unit

Options are placed in extension headers to be included as required. All headers, including the IPv6 header, include a next header field to identify the type of header following immediately after. The IPv6 standard recommends that when using numerous extension headers, they should be placed in the following order:

- IPv6 header
- Hop-by-hop extension header
- Routing header
- Fragment header
- Authentication header

- Encapsulating security payload header
- Destination options header

Maintaining this order is not necessary, although strongly desirable to increase overall processing efficiency. Next, the IPv6 header and the extension headers will be explained greater detail.

IPv6 Header

The IPv6 header was designed with simplicity as main concern, using a minimalistic approach. As opposed to IPv4, IPv6 uses a fixed length 40 octet header. Using a fixed length header simplifies hardware processing in routers, hence increasing performance.

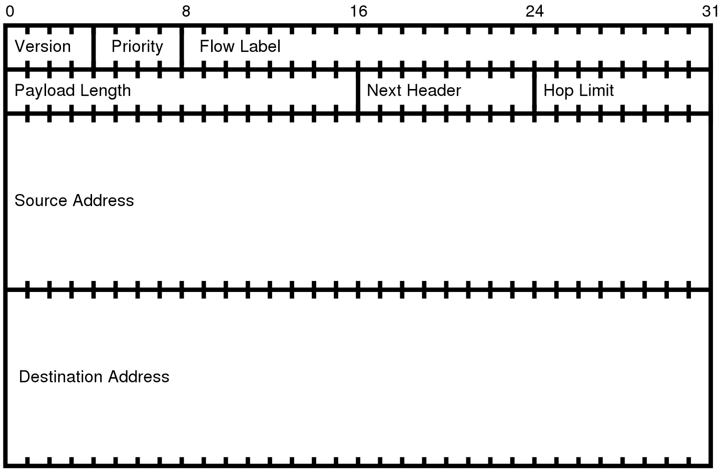


Figure A.2: IPv6 header

The IPv6 header is roughly twice the length of its IPv4 predecessor but contains fewer field, resulting in simplified routing and processing. The following fields make up the IPv6 header

- Version (4 bits) : IP version number; set to 6.
- Priority (4 bits) : Traffic priority value, discussed below.
- Flow label (24 bits) : Enabling labelling of packets between a source and a destination to control traffic, discussed below.
- Payload length (16 bits) : Length in octets of the remaining IPv6 packet following the IPv6 header, including extension headers and transport layer PDU.

- Next header(8 bits) : Identifies the header type following the IPv6 header.
- Hop limit (8 bits) : Contains the remaining number of allowable hops for the packet. The source sets a desired value that is decreased for every hop. If the counter reaches zero, the packet is discarded.
- Source address (128 bits) : IPv6 address of the packet originator.
- Destination address (128 bits) : IPv6 address of the packet destination.

Traffic priority enables a source to classify the type of traffic being delivered relative to other packets from the same source. The priority field allows two types of classification depending on whether the source supports congestion control or not. Within these two priority classes, each packet is assigned one of eight priority levels. The priority classes give an indication of what order packets are being discarded in case of network congestion, relative only within their separate classification (congestion or non-congestion controlled traffic).

Congestion controlled traffic refers to traffic that enables the source to slow down, or "back off" whenever congestion occur. An example of this is TCP, which uses missing or delayed packet acknowledgments as an indication of congestion.

Characteristic applications of congestion controlled traffic are tolerant to variable delivery delays and out of order delivery. Only concern here is a delivery guarantee, that packets will arrive "eventually". IPv6 traffic priority defines several categories for congestion controlled traffic, in decreasing priority

- Internet control traffic : Traffic used to control the Internet, such as routers or other nodes that should be reachable even during severe congestion. Routing protocols such as Open Shortest Path First (OSPF), RIP (Routing Information Protocol) are included in this category. Other critical control and maintenance protocols such as Simple Network Management Protocol (SNMP) to perform dynamic configuration are also subject to this priority category.
- Interactive traffic : The second most important priority category is interactive traffic, such as a terminal sessions between a user and a host. Rapid response times is of the essence for this type of traffic, that this category tries to uphold.
- Attended bulk transfer : This type of traffic is the most frequently used one in today's civil Internet. Characteristic are those as with interactive traffic, with the difference that the user is prepared to accept a larger amount of delay than during an interactive session. These transfers usually involve large amounts of data being transferred, with a user waiting for the process

to complete. Two good examples for this category is File Transfer Protocol (FTP) and Hyper Text Transfer Protocol (HTTP).

- Unattended data transfer : This traffic category includes scenarios where large amount of data is to be transferred, initiated by a user that, by instead of waiting for the transfer to be complete, goes on doing other tasks. The most compelling example in this category is electronic mail.
- Filler traffic : Traffic in this category is meant to be transferred in the background, after all other traffic has been delivered. An good example here is USENET messages.
- Uncharacterized traffic : If no directive is given by the upper layer application, on what traffic category to use, it is placed in this category. Traffic in this category is delivered with lowest priority, surpassed by traffic from all other categories.

Non-congestion controlled traffic is characterized by the lack of feedback from the network in case of congestion. With this kind of traffic, fairly constant data rates and constant delivery delays are desirable. Examples of this type of traffic include streaming audio and video, such as on-line conferencing. Due to the real-time nature of these types of applications, it makes little sense to retransmit unacknowledged data. An example of a transport layer protocol falling under this category is UDP.

As with congestion controlled traffic, eight traffic priority classes exist to distinguish in what order to start discarding packets in case of filled buffers due to congestion.

Audio such as a voice conversation would typically be placed in a high priority class, as dropped or delayed packets would be experienced as clicks and pops to the end user. Loss-insensitive traffic on the other hand, will generally be placed in a lower prioritized class since occasionally discarded packets are less noticeable. An example of this is highly redundant video, where an occasional lost packet makes little difference to the result.

A “flow” in IPv6 is defined as a sequence of packets from a single specific source, to a single specific destination, where destination may be either unicast or multicast. A flow is uniquely identified by combining the source address with a non-zero 24-bit flow label in the IPv6 header. In this manner, all packets originating from the same source, carrying the same flow label belong to the same flow. The source may request special handling by the packets within a flow, to be applied by all intermediate routers on its way to the destination.

Initially, no special importance exist for a particular flow label. Instead, the behaviour of a specific flow must be negotiated by means of either a control protocol or by using an header extension such as the hop-by-hop extension header. Special behaviour with regard to buffer sizes, forwarding priority and other quality of service options is then applied to intermediate routers. [5]

Hop-by-hop Extension Header

The Hop-by-hop header carries information to be processed by every node along the path of a packet from source to destination. [5]

Routing Header

The Routing header allows a source to specify intermediate nodes for a packet to visit before reaching its ultimate destination. The Routing header is not processed until the packet has reached the destination specified in the IPv6 Header. [5]

Fragment Header

The Fragment header is used in IPv6 to send packets larger than the link MTU (Maximum Transfer Unit) of the path. The header includes fields for fragment offset, fragment identification and to determine if the current fragment is followed by other ones. The motivation behind the inclusion of this header is that fragmentation in IPv6 is performed only by source nodes and not by nodes along a packets delivery path. This topic will be discussed in greater detail in section A.4. [5]

Authentication Header

The Authentication header is used to provide a connectionless data integrity and authentication service for IP datagrams. More information on this topic can be found in RFC 2402. [15]

Encapsulation Security Payload Header

The Encapsulation Security Payload (ESP) header is used either as a separate mechanism, or in combination with the authentication header, to provide a great mixture of security services for IPv6. More information on this topic can be found in RFC 2406. [16]

Destination Options Header

The Destination Options header is used to transport optional information only processed by the destination node. [5]

A.2 Addressing

As mentioned earlier, the strongest attraction for defining a new IP was the urgent need for more addresses. IPv6 defines three types of addresses to be allocated within its 128 bit addressing length

- Unicast : Identifies a single interface. A packet sent to this address is delivered to the identified interface.
- Multicast : Identifies a set of interfaces. A packet sent to this address is delivered to all interfaces identified by this address.
- Anycast : Identifies a set of interfaces. A packet sent to this address is delivered to precisely one address identified by this address.

A single interface may have multiple unique unicast addresses assigned to it. The type of an IPv6 address is determined by the leading bits of the address. In an initial phase, some allocations have been made of the total address space and are shown in table A.1.

The global unicast addresses are aggregatable by using bit-wise masks in a similar fashion to the IPv4 CIDR. The general format for IPv6 global unicast addresses is as shown in figure A.3.

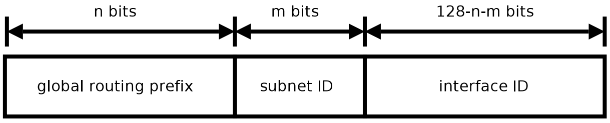


Figure A.3: IPv6 unicast address allocation hierarchy

Two other unicast addressing scopes have also been defined in Link-Local and Site-Local, to be used on a single link and within a single site respectively. As a recent note, IETF decided to deprecate the Site-Local address prefix during the San Francisco meeting in 2003. [10]

Allocation	Prefix (binary)	Fraction of address space
Reserved	0000 0000	1/256
Unassigned	0000 0001	1/256
Reserved for NSAP Allocation	0000 001	1/128
Reserved for IPX Allocation	0000 010	1/128
Unassigned	0000 011	1/128
Unassigned	0000 1	1/32
Unassigned	0001	1/16
Global Unicast Addresses	001	1/8
Unassigned	010	1/8
Unassigned	011	1/8
Unassigned	100	1/8
Unassigned	101	1/8
Unassigned	110	1/8
Unassigned	1110	1/16
Unassigned	1111 0	1/32
Unassigned	1111 10	1/64
Unassigned	1111 110	1/128
Unassigned	1111 1110 0	1/512
Link-Local Unicast Addresses	1111 1110 10	1/1024
Site-Local Unicast Addresses	1111 1110 11	1/1024
Multicast Addresses	1111 1111	1/256

Table A.1: IPv6 address allocation

A.3 Mobility support

To cope with the changing nature of the Internet, evolving from a set of fixed hosts to a more mobile environment, some issues with regard to mobility need to be addressed

- To keep alive a communication between a Mobile Node (MN) and a correspondent node while the MN moves between two separate IP subnets. The operation should be as transparent to the users as possible.

- Allow a MN to be addressed with the same IP address, regardless of the current subnet it's connected to.

IPv6 provide techniques for dealing with the above issues, maximizing performance and minimizing the need of packet tunneling as seen in Mobile IPv4. The basic idea behind Mobile IP and the Home Agent is similar to the situation of moving from one apartment to another, leaving a forwarding address with your old post office. The old post office (i.e. Home Agent) forwards mail to the new post office, which in turn forwards the mail to you.

Movement detection

When a MN moves to a foreign network it starts by detecting its movement, i.e. discover the new subnet. The MN then obtains a temporary address in the foreign network, also referred to as a Care-of address (CoA). The new address is acquired by using either stateless or stateful auto address configuration. The mobile node then sends an update message regarding its movement to its Home Agent (HA) and any correspondent nodes involved.

The MN detects movement from one IP subnet to another using the Neighbor Discovery Protocol (NDP). Rather than waiting for a router or a neighbor to announce their presence, the MN broadcasts a Router Solicitation message and waits for a response in a Router Advertisement message. This information is kept in a cache in the MN called Neighbor Cache together with a list of prefixes and of default routers available. When a MN detects it's no longer in contact with its current default router a new one is chosen from the list and a Binding Update (BU) is issued. When a new CoA is assigned a BU message is sent back to the HA.

Mobile IPv6 messages

The most significant signaling messages introduced to enable mobility support are

- Binding Refresh Request (BRR) : The BRR is sent by correspondent nodes to a MN requesting it to update its mobility binding.
- Binding Update (BU) : The BU is used by a MN to notify its HA and other correspondent nodes of its new CoA.
- Binding Acknowledgment (BA) : The BA is sent as a receipt to the BU if an acknowledgment is requested.
- Binding Error (BE) : The correspondent node uses the BE message to signal a mobility related error to the MN.

Triangle routing

Consider the MN has registered its CoA to the HA. While being away from home, any packets addressed to the MN are interrupted by the HA and forwarded to the MN.

Packets sent by the MN to a correspondent node are sent directly to the destination with source address of the packet being set to the home address. The scenario is shown in figure A.4.

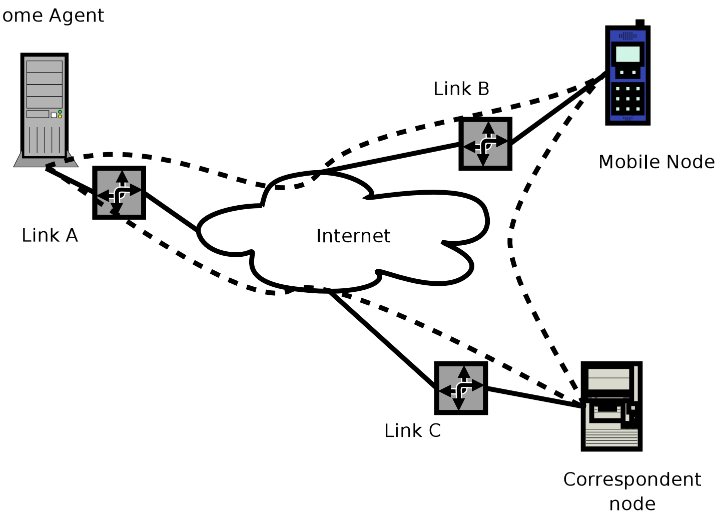


Figure A.4: Triangle routing between the Mobile Node and a correspondent node through the Home Agent.

Incoming packets are routed through the HA while outgoing packets are routed straight to the correspondent node. This routing is normally referred to as triangle routing[13].

Route optimization

To avoid triangle routing a MN can send Binding Updates to any correspondent nodes it's communicating with. This combined with the Binding Cache allows the correspondent node to use the current CoA of the MN, instead of the home address.

Any IPv6 node transmitting a packet first checks the Binding Cache for this destination address. If an entry is found the packet will be sent directly to the MN using a routing header. The route specified by this header includes two hops, first the CoA and then the home address of the MN. When the packet is received in the MN, it "forwards" the packet to the next hop, i.e. the home address. Hence the packet is looped back internally within the MN, making the optimization transparent to the application.

Chances are that no entry exists in the Binding Cache, resulting in the packet being sent the normal way. The packet is routed to the specified network and received by the destination node. If the node is a MN away from home, the packet is intercepted by the HA on the home link and tunneled to the MN. [13]

Route Optimization, a voluntary option in IPv4, but a mandatory extension in IPv6. This mandatory mechanism solves the triangle routing problem in a simple and elegant manner.

Binding management

When a MN configures a CoA it has to re-register the new address to its HA and other correspondent nodes. To accomplish this a BU is issued to the nodes that needs the update.

By inspecting the header of an incoming packet, a MN can detect if the correspondent node transmitting the packet has a Binding Cache entry for the MN. If an entry exists in the correspondent node, the packet is addressed directly to the MN's CoA, otherwise the destination is set to the MN's home address. From there the packet is tunneled by the HA to the MN. If the MN detects that no entry exist in the correspondent node, it may issue a BU.

Each entry in the Binding Cache is coupled with an expiration timer, before which the correspondent node need to initiate a BRR to ensure correct operation.

Home Agent discovery

In the case when the MN does not know the IP address of its HA, Mobile IPv6 provides a mechanism for the MN to dynamically locate the HA's IP. This is possible by only knowing the subnet prefix of its own home network.

To accomplish this the MN sends a Binding Update addressed to the HA's subnet anycast address hence reaching the routers on its home link acting as a HA. The one HA receiving the message, rejects the Binding Update and sends a Binding Acknowledgment containing a list of all HAs on the home link. This list is periodically updated in all routers by transmitting and receiving Router Advertisement messages. As the list is constructed the order of the HAs in the list is decided by a preference value in each node.

After receiving the list of available HAs the MN sends a BU to one of these addresses, starting with the one with highest priority. If no acknowledgment is received from the HA the next node from the list is selected and an Binding Update transmitted to. This continues until a confirmation of the update is received and the binding is completed. [12]

A.4 Performance

One of the key design issues when designing IPv6 was how to cope with future performance demands and to enhance the routing efficiency. Several ideas emerged and some were included in the final specification. One of the issues that needed attention was the fact that addresses now were 128 bits instead of the 32 bit, inevitably increasing the size needed for the header. [29]

By decreasing the number of required fields in the IPv6 header, processing overhead can be greatly reduced. As further assistance the header is kept static, making silicon router implementation easier. [29]

Thanks to the aggregation possibilities of IPv6 addressing architecture, routing tables in the core of the Internet can be kept relatively small. Table lookups are made with less effort, reducing one routing bottleneck. [29]

To further simplify the network layer, checksum has been completely removed and responsibility being moved to upper layers (e.g. UDP, TCP, ICMPv6) as well as accomplished by most layer two protocols such as Ethernet.

As opposed to IPv4, the only node responsible for fragmentation is the source node. The receiving node reassembles the datagram once all fragments have been delivered. Before the source initiate a transition it first determines the path MTU, and then sets the fragment size accordingly. The fragment size may be set up in two ways. Either set it to the IPv6 minimum link MTU (specified in [5] to 1280 octets), or use the path MTU discovery protocol for IPv6. [21]

The source node assumes that the path MTU is the known MTU of the first hop in the path. If any packet sent out on this path is to large to be forwarded by the next node along the path, a ICMPv6 Packet Too Big message is generated and sent back to the source. When receiving such a message, the source reduces the MTU of outgoing packets to the value indicated in the Packet Too Big message. This is an iterative process which adopts the path MTU whenever the topology changes. The source may occasionally try and raise the MTU in order to adopt to a possible increase in the path MTU. By restricting packet fragmentation to the source node only, intermediate nodes don't have to perform any packet fragmentation or reassembly. This will help to increase throughput performance by decreasing the processing costs for each node. [21]

A.5 Network services

Perhaps one of the most significant improvements offered by IPv6 is its address auto configuration features. As the Internet is rapidly evolving from a set of stationary hosts to a more fluid and dynamic network of mobile devices.

The IPv6 mobility allows a mobile device to quickly acquire and transition between addresses as they move to foreign networks, without any need for a foreign agent. Address auto configuration provides a strong foundation for plug-and-play connectivity, eliminating the need of manual configuration at a single host basis.

IPv6 address auto configuration may be performed in two different manners, either stateful or stateless.

Stateless address auto configuration is conducted by using a number of ICMPv6 control messages where the most important ones are the Router Advertisement (RA) and the Router Solicitation (RS) messages. Any router supporting IPv6 address auto configuration announces its presence by periodical transmission of RA messages, containing among other information, a list of prefixes identifying the subnets of that link. The site administrator is responsible for determining which type of address auto configuration to use by setting the M bit in the RA message.

Hosts wishing to gain a fully qualified IPv6 address using auto configuration, listens for RA messages. When a RA message is received by the host, it may construct its own address by appending a token to the prefix contained within the message received. An appropriate token may be to use the MAC address of the interface card. A host may request a RA message by sending a RS message to the router.

Stateful address auto configuration uses an external address pool such as a DHCPv6 server to obtain addresses from. The DHCPv6 server maintains a database of the addresses available, and has a tight control over address assignments. When a host wishes to obtain an address using stateful address auto configuration, it sends a DHCPv6 solicitation message to which the DHCPv6 server responds with a DHCPv6 advertisement message containing the leased address.

A.6 Security

IPv4 and IPv6 share a common security mechanism called “IPsec” that provides strong encryption, strong authentication, message non-repudiation and message integrity. One problem with IPsec under IPv4 is when a host is located behind a NAT firewall breaking the end-to-end semantics. Since IPv6 provide each connected device with a topologically correct address, this is not an issue.

IP level security in IPv6 consists of two functional areas, authentication and privacy. The authentication mechanism guarantees that a received packet was in fact transmitted by the host as in the source of the packet header. This mechanism also ensures that the received packet wasn't altered in transit.The privacy mechanism enables a source to send messages encrypted to a source to prevent eavesdropping.

Since IPsec is mandatory in IPv6, security functionality is guaranteed to be available on all networking platforms using IPv6.

A.7 Transition mechanism

The advantages of IPv6 over IPv4 presented this far is not enough to successfully declare IPv6 the standard Internet Protocol of the future. The key issue for the transition between IPv4 to IPv6 is to maintain compatibility with the tremendous base of installed IPv4 host while deploying IPv6 and gradually phasing out IPv4.

Several mechanisms have been designed to be employed by IPv6 hosts and routers that need connectivity with existing IPv4 infrastructures, either directly or to access other IPv6 nodes. The most eminent transition mechanisms are

- Dual stack
- Point-to-Point IPv6 over IPv4 tunnel
- IPv4 embedded IPv6 address

Dual stack

The most straightforward approach to preserve backward compatible with IPv4 is to provide the entire IPv4 stack while employing IPv6 in parallel. Nodes implementing both stacks are capable of simultaneously sending and receiving both IPv4 and IPv6 packets. Dual stack nodes can because of their interoperability with both IP families, be used as tunnel endpoints.

While this seems like an all pros solution, its coupled with some issues. First, it does not reduce the demand for globally routeable IPv4 addresses. Secondly it increases network complexity due to the need for a double (IPv4/IPv6) routing infrastructure. [9]

Point-to-Point IPv6 over IPv4 tunnel

One solution that minimizes the routing infrastructure complexity is to use IPv4 tunnels to carry IPv6 packets between “islands” of IPv6 hosts through a network of IPv4 routers. The tunneling is performed by encapsulating IPv6 packets within IPv4 headers between manually configured fixed endpoints. [4]

6bone[8] is a worldwide experimental IPv6 network mainly built on top of the Internet using this tunneling technique. The 6bone project was initially started as an effort to test the standards and the initial implementations. Lately the project has turned focus to more on testing the transition and operational procedures.

IPv4 embedded IPv6 address

This is a technique also referred to as automatic tunneling, where a IPv4 address is embedded within a IPv6 address using the low-order 32 bits. IPv4 compatible addresses in IPv6 are structured as shown in figure A.5.

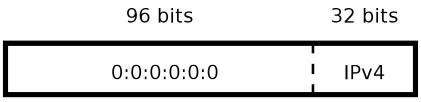


Figure A.5: IPv4 compatible IPv6 address format

The automatic tunneling is performed by the tunnel endpoint at the transmitting tunnel endpoint where IPv6 is used. A routing table entry for the prefix $0:0:0:0:0:0/96$ can be used to forward matching packets to a pseudo interface which performs automatic tunneling. Since all IPv4 compatible IPv6 addresses match this prefix, all packets to that destination will be auto tunneled. The pseudo interface encapsulates the packet within a IPv4 header and sends it to the low-order 32 bits of the IPv6 address specified.

Abbreviation guide

2G	Second Generation
3G	Third Generation
3GPP	Third Generation Partnership Project
AAL	ATM Adaption Layer
APN	Access Point Name
ARPANET	Advanced Research Project Agency Network
ATM	Asynchronous Transfer Mode
BA	Binding Acknowledgment
BE	Binding Error
BRR	Binding Refresh Request
BSC	Base Station Controller
BSD	Berkeley System Distribution
BU	Binding Update
CDMA	Code Division Multiple Access
CIDR	Classless Inter Domain Routing
CN	Core Network
CoA	Care-of Address
CS	Circuit Switched
DARPA	Defence Advanced Research Projects Agency
DHCP	Dynamic Host Configuration Protocol
DHCPv6	DHCP version 6

DoD	Department of Defense
ESP	Encapsulation Security Payload
ETSI	European Telecommunications Standards Institute
FDD	Frequency Division Duplex
FDMA	Frequency Division Multiple Access
FTP	File Transfer Protocol
GGSN	Gateway GPRS Support Node
GMSC	Gateway Mobile Switching Center
GPRS	General Packet Radio Service
GSM	Global System Mobile
GSN	GPRS Support Node
GTP	GPRS Tunneling Protocol
HA	Home Agent
HIPERLAN	High Performance LAN
HLR	Home Location Register
HTTP	Hyper Text Transfer Protocol
IANA	Internet Assigned Numbers Authority
IE	Information Element
IETF	Internet Engineering Task Force
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
IPng	IP next-generation
IPsec	IP security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISP	Internet Service Provider
ITU	International Telecommunication Union
LAN	Local Area Network
LSB	Least Significant Bit

MIP	Mobile IP
MIPv4	Mobile IPv4
MIPv6	Mobile IPv6
MN	Mobile Node
MS	Mobile Station
MSB	Most Significant Bit
MSC	Mobile Switching Center
MT	Mobile Terminal
MTU	Maximum Transfer Unit
NAT	Network Address Translation
NCP	Network Control Protocl
NDP	Neighbor Discovery Protocol
NSAPI	Network Service Access Point Identifier
OS	Operating System
OSPF	Open Shortest Path First
PDA	Personal Digital Assistant
PDCP	Packet Data Convergence Protocol
PDN	Packet Data Network
PDP	Packet Data Protocol
PDU	Protocol Data Unit
PLMN	Public Land Mobile Network
PS	Packet Switched
QoS	Quality of Service
RA	Router Advertisement
RFC	Request For Comment
RIP	Routing Information Protocol
RNC	Radio Network Controller
RNS	Radio Network Systems
RS	Router Solicitation

RSVP	Resource Reservation Protocol
SGSN	Serving GPRS Support Node
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol / Internet Protocol
TDD	Time Division Duplex
TDMA	Time Division Multiple Access
TE	Terminal Equipment
TEID	Tunnel Endpoint Identifier
UDP	User Datagram Protocol
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
UMTS-2000	International Mobile Telecommunications 2000
USIM	UMTS Subscriber Identity Module
UTRAN	UMTS Terrestrial Radio Access Network
VLR	Visitor Location Register
VoIP	Voice over IP
WAP	Wireless Application Protocol
WCDMA	Wideband CDMA
WWW	World Wide Web

Bibliography

[1] 3rd Generation Partnership Project, *General packet radio service (gprs); gprs tunnelling protocol (gtp) across the gn and gp interface.* Web site, 23 June 2000. `http://www.3gpp.org/ftp/Specs/html-info/-29060.htm`.

[2] J. Bound and A. Cini, *Ipv6 (ipng) - the coming "Big Bang" in cyberspace*. Web site, 11 Aug. 2003. `http://www.cs-ipv6.lancs.ac.uk/ipv6/-documents/papers/bound`.

[3] S. Bradner and A. Mankin, *RFC 1752: The recommendation for the ip next generation protocol*, Jan. 1995.

[4] B. Carpenter and K. Moore, *RFC 3056: Connection of ipv6 domains via ipv4 clouds*, Feb. 2001.

[5] S. Deering and R. Hinden, *RFC 2460: Internet protocol, version 6 (ipv6) specification*, Dec. 1998.

[6] J. Elischer and A. Cobbs, *The netgraph networking system.* Web site, 03 Nov. 2003. `http://www.elischer.org/netgraph`.

[7] Ethereal, *The ethereal network analyzer.* Web site, 04 Nov. 2003. `http://www.ethereal.com`.

[8] B. Fink, *6bone home page.* Web site, 19 Aug. 2003. `http://www.6bone.net`.

[9] R. Gillian and E. Nordmark, *RFC 2893: Transition mechanisms for ipv6 hosts and routers*, Aug. 2000.

[10] R. Hinden and S. Deering, *RFC 3513: Internet protocol version 6 (ipv6) addressing architecture*, Apr. 2003.

[11] Internet Society, *A brief history of the internet.* Web site, 11 Aug. 2003. `http://www.isoc.org/internet/history/brief.shtml`.

[12] D. Johnson and S. Deering, *RFC 2526: Reserved ipv6 subnet anycast addresses*, 10 Feb. 2003.

[13] D. Johnson and C. Perkins, *Mobility support in ipv6.* Web site, 09 Feb. 2003. `draft-ietf-mobileip-ipv6-20.txt`.

[14] JWS, *Tcpdump public repository.* Web site, 28 Feb. 2003. `http://www.tcpdump.org`.

[15] S. Kent and R. Atkins, *RFC 2402: Ip authentication header*, Nov. 1998.

[16] ——, *RFC 2406: Ip encapsulating security payload (esp)*, Nov. 1998.

[17] G. C. Kessler, *An overview of tcp/ip protocols and the internet.* Web site, 11 Aug. 2003. `http://www.garykessler.net/library/tcpip.html`.

[18] D. Kristula, *The history of the internet.* Web site, 11 Aug. 2003. `http://www.davesite.com/webstation/net-history.shtml`.

[19] E. M. Wasserman, *RFC 3314: Recommendations for ipv6 in third generation partnership project (3gpp) standards*, Sept. 2002.

[20] D. Mayr, *The history of the internet and the www.* Web site, 11 Aug. 2003. `http://members.magnet.at/dmayr/history.htm`.

[21] J. McCann, S. Deering, and J. Mogul, *RFC 1981: Path mtu discovery for ip version 6*, Aug. 1996.

[22] D. Mitzel, *RFC 3002: Overview of 2000 iab wireless internetworking workshop*, Dec. 2000.

[23] Moschovitis Group, *History of the internet.* Web site, 11 Aug. 2003. `http://www.historyoftheinternet.com`.

[24] OpenBSD, *Openbsd.* Web site, 6 Aug. 2003. `http://www.openbsd.org`.

[25] ——, *Openbsd.* Web site, 11 Sept. 2003. `http://www.openssh.org`.

[26] T. G. Project, *Glib reference manual.* Web site, 01Aug. 2003. `http://developer.gnome.org/doc/API/2.0/glib`.

[27] RADCOM, *Radcom - network testing and quality management.* Web site, 04 Nov. 2003. `http://www.radcom.com`.

[28] J. Rajahalme, A. Conta, B. Carpenter, and S. Deering, *Ipv6 flow label specification.* Web site, Oct. 2003. `draft-ietf-ipv6-flow-label-08.txt`.

[29] W. Stevens, *Ipv6: The new internet protocol.* Web site, 11 Aug. 2003. `http://www.cs-ipv6.lancs.ac.uk/ipv6/documents/papers/-stallings`.

[30] The BSD Foundation, *The netbsd project.* Web site, 4 Aug. 2003. `http://www.netbsd.org`.

[31] The FreeBSD Project, *The freebsd project.* Web site, 6 Aug. 2003. http://www.freebsd.org.

[32] The OpenSSL Project, *Openssl: The open source toolkit for ssl/tls.* Web site, 11 Sept. 2003. http://www.openssl.org.

[33] UMTSWorld.com, *Cdma tutorial.* Web site, 28 Oct. 2003. http://www.umtsworld.com/technology/cdmabasics.htm.

[34] ——, *The history of umts and 3g development.* Web site, 28 Oct. 2003. http://www.umtsworld.com/umts/history.htm.