Technical Document

GSM PROTOCOL STACK

G23

IP AND ICMP

MESSAGE SEQUENCE CHARTS

Document Number:	8444.201.00.001
Version:	0.4
Status:	Draft
Approval Authority:	Udo Sprute
Creation Date:	1998-Mar-25
Last changed:	2015-Mar-08 by Jacek Kwasnik
File Name:	ip.doc

Important Notice

Texas Instruments Incorporated and/or its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products, software and services at any time and to discontinue any product, software or service without notice. Customers should obtain the latest relevant information during product design and before placing orders and should verify that such information is current and complete.

All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment. TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI products, software and/or services. To minimize the risks associated with customer products and applications, customers should provide adequate design, testing and operating safeguards.

Any access to and/or use of TI software described in this document is subject to Customers entering into formal license agreements and payment of associated license fees. TI software may solely be used and/or copied subject to and strictly in accordance with all the terms of such license agreements.

Customer acknowledges and agrees that TI products and/or software may be based on or implement industry recognized standards and that certain third parties may claim intellectual property rights therein. The supply of products and/or the licensing of software does not convey a license from TI to any third party intellectual property rights and TI expressly disclaims liability for infringement of third party intellectual property rights.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products, software or services are used.

Information published by TI regarding third–party products, software or services does not constitute a license from TI to use such products, software or services or a warranty, endorsement thereof or statement regarding their availability. Use of such information, products, software or services may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Change History

Date	Changed by	Approved by	Version	Status	Notes
1998-Mar-25	NI et al		0.1	Being Processed	1
2000-May-29	SG et al		0.2	Being Processed	
2003-May-15	XGUTTEFE		0.3	Draft	
2003-Aug-30	Jacek Kwasnik	Udo Sprute	0.4	Draft	2

Notes:

1. Initial version
2. Brief review minor corrections

Table of Contents

List of Figures and Tables

List of References

[ISO 9000:2000] International Organization for Standardization. Quality management systems - Fundamentals and vocabulary. December 2000

1 Introduction

The Internet Protocol (IP) has been designed to enable connectionless best-effort delivery of packets (called *datagrams*) host-to-host over a network. It is in use on the world-wide Internet since the early 1980s and has increasingly been adopted for other wide-area and local networks since. The current version of the Internet Protocol is 4 (IPv4). Deployment of the succeeding version 6 (IPv6) has not yet left the R&D area. Due to the large world-wide investments in IPv4-based technology it will take several years until IPv6 becomes dominant on the Internet and other networks.

The transport layer protocols built on top of IP are mainly UDP, which hands through IP's capabilities to applications, and TCP, which offers a reliable byte-stream data transmission. Since TCP is the dominant Internet transport protocol, IP and the protocols built on top of it, are often collectively called *TCP/IP* or the *Internet protocols suite*.

The Internet Control Message Protocol (ICMP) is based on IP. It is used to transmit control information (mostly about error conditions) regarding the communication in the IP layer and the transport layer (usually UDP and TCP). Implementation of ICMP is mandatory for all IP implementations.

1.1 Standards and Supplementary Documents

Internet standards like IP are described in the "Request for Comments" (RFC) series of documents. The RFCs are available online from http://www.rfc-editor.org/ (inside TI from http://chuck/rfcs/). The following RFCs are relevant for IP and ICMP:

[RFC 791]
Jon Postel (ed.): *Internet Protocol*. RFC 791, September 1981.
http://chuck/rfcs/rfc0791.txt or ftp://ftp.isi.edu/in-notes/rfc791.txt
The initial IP specification.

[RFC 792]
Jon Postel: *Internet Control Message Protocol*. RFC 792, September 1981.
http://chuck/rfcs/rfc0792.txt or ftp://ftp.isi.edu/in-notes/rfc792.txt
The initial ICMP specification.

[RFC 1071]
R.T. Braden, D.A. Borman, C. Partridge: *Computing the Internet checksum*. RFC 1071, September 1988.
http://chuck/rfcs/rfc1071.txt or ftp://ftp.isi.edu/in-notes/rfc1071.txt
This RFC gives several examples for implementation of the Internet checksum algorithm. RFC 1141 adds a method for incremental update of the checksum, e. g. on TTL decrement.

[RFC 1122]
Robert. T. Braden (ed.): *Requirements for Internet Hosts – Communication Layers*. RFC 1122, October 1989.
http://chuck/rfcs/rfc1122.txt or ftp://ftp.isi.edu/in-notes/rfc1122.txt
Among other topics, RFC 1122 contains clarifications and additions to the initial IP and ICMP specifications.

[RFC 1700]
Joyce K. Reynolds, Jon Postel: *Assigned Numbers*. RFC 1700, October 1994.
http://chuck/rfcs/rfc1700.txt or ftp://ftp.isi.edu/in-notes/rfc1700.txt
RFC 1700 contains numeric values and other definitions in use with the TCP/IP protocol

suite.

[RFC 1812]
Fred Baker (ed.): *Requirements for IP Version 4 Routers*. RFC 1812, June 1995.
http://chuck/rfcs/rfc1812.txt or ftp://ftp.isi.edu/in-notes/rfc1812.txt
This RFC contains an updated list of the ICMP types and codes.

[RFC 2474]
Kathleen Nichols, Steven Blake, Fred Baker, David L. Black: Definition of the Differentiated Services Field (DS Field)in the IPv4 and IPv6 Headers. RFC 2474, December 1998.
http://chuck/rfcs/rfc2474.txt or ftp://ftp.isi.edu/in-notes/rfc2474.txt
This document redefines the meaning of the *Type of Service* field in the IP header.

In addition to the RFCs, the following books are valuable sources of information:

[MacKusick et al. 1996]
Marshal Kirk McKusick, Keith Bostic, Michael J. Karels, John S. Quarterman: *The Design and Implementation of the 4.4 BSD Operating System*. Addison-Wesley 1996.
Chapter 13 contains a very concise overview of IP and ICMP.

[Stevens 1994a]
W. Richard Stevens: *TCP/IP Illustrated Volume 1 – The Protocols*. Addison-Wesley, 1994.
This book contains an excellent introduction to the TCP/IP protocol suite. Relevant to IP and ICMP are the chapters 3, 6, and 9.

[Stevens 1994b]
W. Richard Stevens: *TCP/IP Illustrated Volume 2 – The Implementation*. Addison-Wesley, 1994.
In this book Stevens annotates the complete 4.4 BSD TCP/IP source code. Relevant to IP and ICMP are chapters 8 to 11.

These RFCs and books were the main sources used in preparation of this document.

The 4.4 BSD TCP/IP software documented in [Stevens 1994a] is an excellent example of a full-featured and very mature implementation of the Internet protocol suite. While it is for several reasons not suited for direct integration into a mobile phone, it is a good reference for any TCP/IP implementer. The TCP/IP code of FreeBSD, a direct successor of 4.4 BSD, can been found at ftp://ftp.freebsd.org/pub/FreeBSD/FreeBSD-stable/src/sys/netinet/ (current version of the "stable" development branch) or, inside TI, at http://chuck/freebsd/src/sys/netinet/ (from FreeBSD release 3.4 of December 1999).

1.2 IP Basics

1.2.1 Features

IP is the network layer (ISO/OSI layer 3) protocol of the TCP/IP protocol suite. ICMP messages are encapsulated in IP datagrams, but ICMP belongs conceptually to the same layer as it transmits control information related to the network layer.

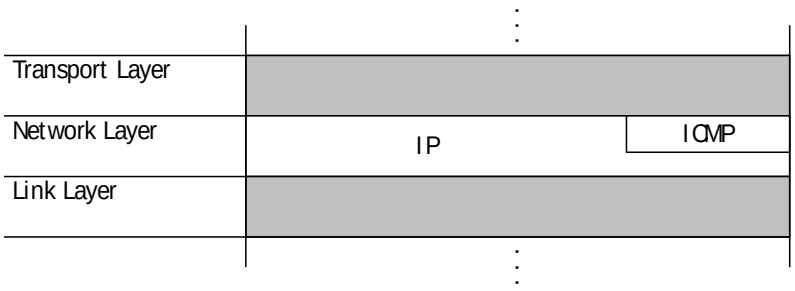


Figure 1-1: ISO/OSI model layer of IP and ICMP

From [Stevens 1994a]:

> IP is the workhorse of the TCP/IP protocol suite. All TCP, UDP, ICMP, and IGMP data gets transmitted as IP datagrams [...]. A fact that amazes many newcomers to TCP/IP, especially those from an X.25 or SNA background, is that **IP provides an unreliable, connectionless datagram delivery service.**
>
> By *unreliable* we mean there are no guarantees that an IP datagram successfully gets to its destination. IP provides a best effort service. When something goes wrong, such as a router temporarily running out of buffers, IP has a simple error handling algorithm: throw away the datagram and try to send an ICMP message back to the source. Any required reliability must be provided by the upper layers (e. g., TCP).
>
> The term *connectionless* means that IP does not maintain any state information about successive datagrams. Each datagram is handled independently from all other datagrams. This also means that IP datagrams can get delivered out of order. If a source sends two consecutive datagrams (first A, then B) to the same destination, each is routed independently and can take different routes, with B arriving before A.

[Emphasis in the first paragraph is mine.]

The main features of IP are:

Addressing and Routing

IP addresses are 32-bit values, usually written as four decimal numbers, each representing an 8-bit-byte portion of the address, most significant byte first. This notation is called “decimal dotted” notation of IP addresses. Example: 172.20.48.48 for the binary value 10101100000101000011000000110000 (most significant bit first).
An IP packet contains the address of the destination host as well as the address of the source host. If an IP packet cannot be sent on a direct point-to-point link or shared network from source to destination, it must travel via several connections (*hops*), on the way being forwarded by intermediate hosts acting as *routers*. A router is a *dual-* or *multi-homed* host (i. e. a host with two or more network interfaces) that can forward packets destined to another host from one link or network to another.

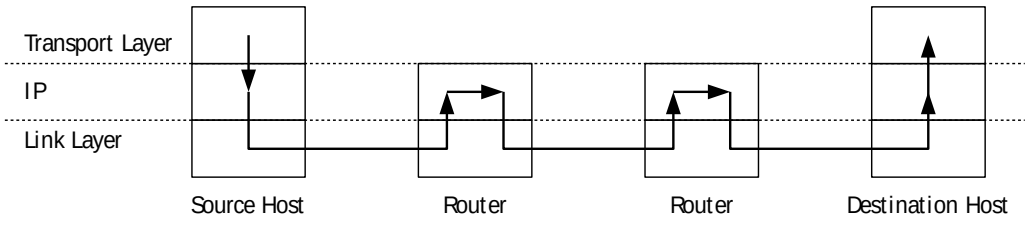


Figure 1-2: IP Routing

In each router, the packet is handed to the IP layer, which decides whether and which way to forward the packet according to the destination address of the packet.
To bound maximum packet lifetime, e. g. when encountering routing loops, a *Time-To-Live* (TTL) field in the packet header is decremented by each router. When the TTL value reaches zero, the packet is discarded.

Fragmentation and Reassembly

If an IP packet (either a complete datagram or a packet already containing a datagram fragment) is too large to be sent over a link, the IP layer in the sending host (either the source of the datagram or a router on the way) can *fragment* the packet. It is split into smaller packets called *fragments*, each containing an IP header and a portion of the payload. The IP layer in the destination host then *reassembles* the original datagram from the fragments received. If not all fragments reach the destination host, the datagram is discarded. (To be precise, the datagram is discarded when the *reassembly timer* expires before all fragments are received.)

> "[...] note the terminology: an *IP datagram* is the is the unit of end-to-end transmission at the IP layer (before fragmentation and after reassembly), and a *packet* is the unit of data passed between the IP layer and the link layer. A packet can be a complete IP datagram or a fragment of an IP datagram." [Stevens 1994a]

There are several *IP options* that can be set in a datagram. They are more or less obsolete in the Internet context and can safely be ignored in the mobile phone context. They must be *safely* ignored, though, in the sense that incoming IP datagrams with options set must not cause the IP layer to crash or deliver invalid data to the upper layers.

IP supports *multicast* (all hosts of a specified group) and *broadcast* (all hosts on a specified network) destinations. In the context of a mobile phone both do not make much sense and can be ignored.

1.2.2 IP Netmasks, Address Classes, Subnetting, Supernetting (CIDR)

Applicability

At least currently netmasks will not be relevant in the mobile phone context; this description is provided only for completeness. Even if another IP-capable device uses the mobile to access the Internet, the connection between the mobile and the other device will be a point-to-point link, such that handling two routes will suffice – a host route to the other device and a default route over the air interface. The concept of netmasks does not apply here.

Netmasks

An IP address consists of a network part and a host part. The network part is determined by the *netmask*. For example, in the IP address 172.20.48.48 with a netmask of 255.255.0.0, the first 16 bits determine the network address (172.20.0.0), and the last 16 bits determine the host inside this network. In words of the C programming language: *network* = *address* & *netmask*.

```
address bits:          10101100.00010100.00110000.00110000 (172.20.48.48)
netmask bits:          11111111.11111111.00000000.00000000 (255.255.0.0)
network address:             10101100.00010100.00000000.00000000
(172.20.0.0)
```

Figure 1-3: IP Netmask Example

The first possible address in a network (in this case 172.20.0.0) is consequently used as a symbol for the network itself, and the last possible address (in this case 172.20.255.255) is used as

the broadcast address for that network. The host part of the address (here 0.0.48.48) is rarely if ever used separately. The network part of the address is used by routers – this way a router needs to know only where to send packets destined for a network, but not for every single host address in that network.

Address Classes

Originally IP addresses have been divided into five *address classes*:

Class	Address bits	Decimal Addresses	Netmask	Max. hosts	Purpose
A	0xxxxxxx.*	0.0.0.0 – 126.255.255.255	255.0.0.0	16777214	Very large networks
B	10xxxxxx.*	128.0.0.0 – 191.255.255.255	255.255.0.0	65534	Mid-size networks
C	110xxxxx.*	192.0.0.0 – 223.255.255.255	255.255.0.0	254	Small networks
D	1110xxxx.*	224.0.0.0 – 239.255.255.255	n. a.	n. a.	Multicast addresses
E	1111xxxx.*	240.0.0.0 – 255.255.255.254	n. a.	n. a.	Experimental

Table 1-1: IP Address Classes

The addresses 127.0.0.0 – 127.255.255.255 are reserved for loopback interfaces; the address 255.255.255.255 is an additional address for a broadcast on the local network.

Subnetting

To enable separation of a network into several segments, e. g. for different buildings, *subnetting* has been introduced. With subnetting, each network interface is assigned an explicit netmask along with its IP address. This way a class C network can be separated into several *subnetworks*, e. g. four separate networks with a netmask of 255.255.255.192, each with up to 62 hosts, or even into three networks with one using a netmask of 255.255.255.128 and two using 255.255.255.192.

With subnetting, the netmask technically needs not be a consecutive bitmask, but it usually is, except for pathological cases. For convenience, the netmask of a network or subnet is thus usually described only by its length and written with a slash behind the (often abbreviated) network address: 192.168.220/24 is the network 192.168.220.0 with a 24-bit long netmask (255.255.255.0), i. e. a regular class C network.

An example may show how a subnetted network may look like:

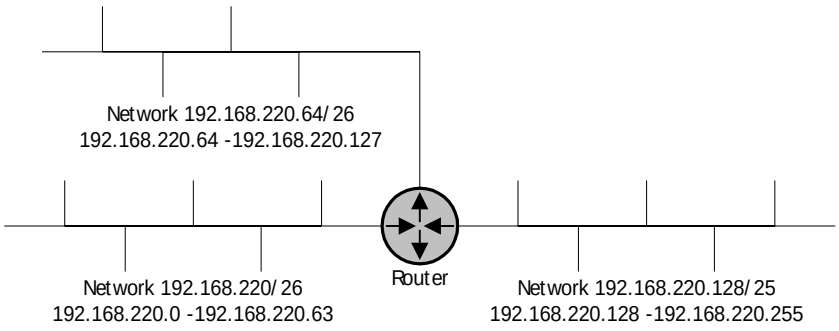


Figure 1-4: IP Subnetting Example

Supernetting (CIDR)

With the growth of the Internet, two problems became evident:

1. The division of the IP address space into classes proved to be not flexible enough.

2. The routing tables in Internet routers grew beyond the abilities of hardware, software, and people.

Supernetting, also called *Classless Inter-Domain Routing* (CIDR), was introduced as a solution. CIDR abandons the concept of network classes at all[1]. Every network address is now associated with a netmask regardless of its class. This way class A networks can be divided into networks with e. g. 24-bit netmasks or even smaller ones; routes to several networks can, if possible, be aggregated to a route to a *supernet*, e. g. all networks containing addresses from 133.0.0.0 – 133.255.255.255, class B networks in the old notation, can be seen as a supernet 133.0.0.0/8 or shorter 133/8, needing only one entry in the routing table of an Internet router instead of 256. (RFCs 1517 – 1519)

1.2.3 IP Packet header

All IP header fields are in *network order*, i. e. big endian (most significant bit/byte first). C library implementations usually supply the functions or macros ntohl(), htonl(), ntohs(), htons() to convert long and short values between network order and host order.

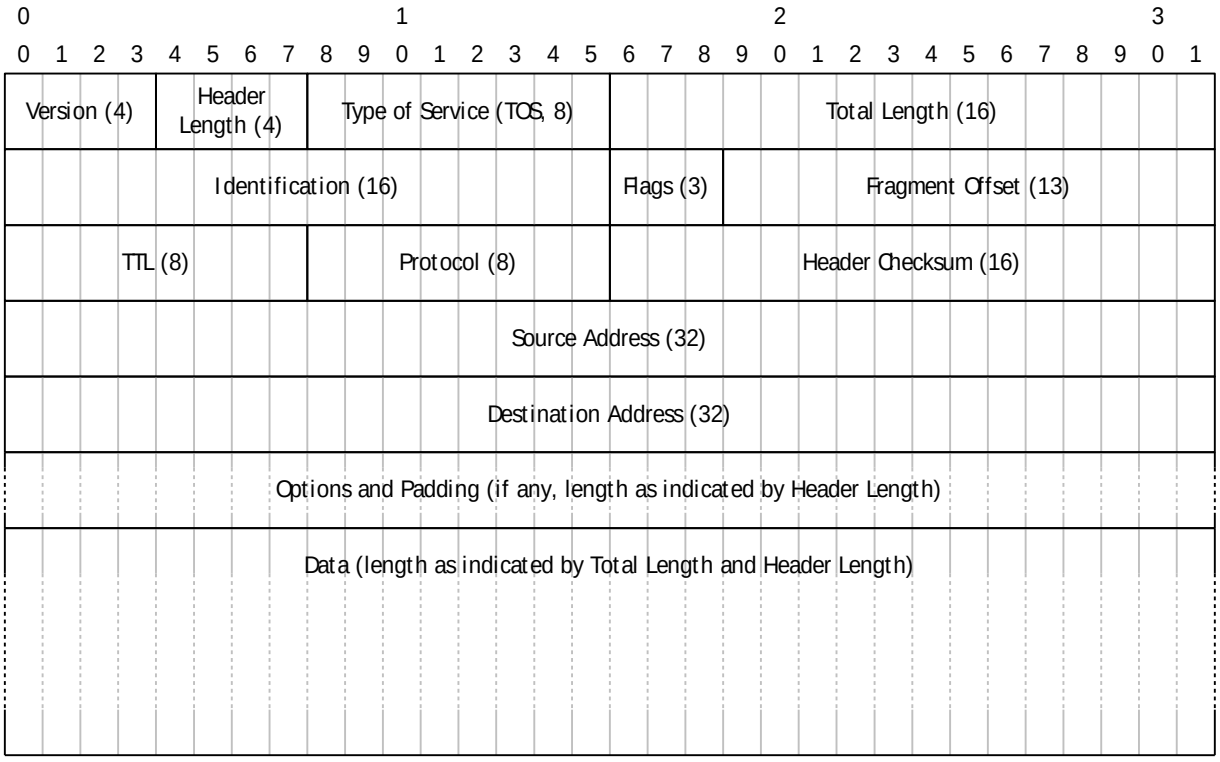


Figure 1-5: IP Packet Header

The lengths in the diagram above are given in bits.

[1] Sometimes, however, it is still convenient to talk about networks with netmasks of 8, 16, and 24 bits as class A, B, and C networks, regardless of their place in the address space, e. g. "the class B network 62.207.0.0".

Version

Protocol version of IP. Always 4 with IP version 4.

Internet Header Length (IHL)

Length of the IP packet header in 32-bit words. The minimum value (i. e. if no IP options are present) is 5.

Type Of Service (TOS)

The TOS field allows the sender to select a tradeoff between high throughput, high reliability, and low delay. As the meaning of the TOS field is currently undergoing changes (a standard is proposed as [RFC 2474]), it is best to send IP packets with a TOS of zero if there is no compelling reason to do otherwise (i. e. answering to an IMCP query with TOS set).

Total Length

Total length of the packet in “octets” (8-bit bytes). The total length is adjusted to the length of the fragment when a packet is fragmented. The maximum value is 65535 (2^{16}–1), but since a datagram of this size would for most link layers have to be split into lots of fragments, this value is not used in common applications. Common lengths are up to 8192 bytes in LANs and 1500 bytes (or less) on WANs. Each IP implementation must be capable of receiving datagrams with a total length of up to 576 bytes. (The total length of a fragmented datagram is not known on the receiving side before the last fragment arrives.)

Identification

Identification of the datagram, used when the datagram is reassembled from fragments. The combination of source address, destination address, protocol number, and identification must be unique (at least until all reassembl timers have expired). The identification is usually incremented with each datagram sent by a host.

Flags

Fragmentation flags. Bit 0: reserved, must be zero; Bit 1: (DF) 0 = May Fragment, 1 = Don't Fragment; Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments. MF is set to 1 in all fragments of a datagram except the last.

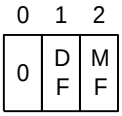


Figure 1-6: IP Datagram Fragmentation Flags

Fragment Offset

Contains the offset of the current fragment from the beginning of the datagram in units of 8 8-bit bytes. The first fragment (or the only packet of a non-fragmented datagram) has a fragment offset of 0.

The meaning of the various combinations of the flags and offset values can best be put in a table:

DF	MF	Offset	
0	0	0	unfragmented datagram
0	0	!= 0	last fragment
0	1	0	first fragment
0	1	!= 0	fragment in the middle
1	0	0	unfragmented datagram
1	0	!= 0	*invalid combination*

1	1	0	*invalid combination*
1	1	!= 0	*invalid combination*

Table 1-2: IP Flags and Fragment Offset Combinations

When sending an IP packet, we need not set DF.

Time to Live

The Time to Live (TTL) is set to an initial value by the IP layer of the source host. It is decremented by each router that forwards the packet. When the TTL has reached zero, the packet is discarded and an ICMP error of the type “TTL expired” is sent to the source host. The currently recommended initial value for the TTL is 64 [RFC 1700]. No packet must be sent with a TTL of 0.

Protocol

Protocol number of the next-higher-level protocol. The protocol numbers are defined in [RFC 1700], or, more current, in http://www.isi.edu/in-notes/iana/assignments/protocol-numbers. Relevant in this context are the numbers for ICMP (1) and UDP (17).

Header Checksum

Checksum over all header fields. This checksum must be recalculated on each change of a header field. This includes fragmentation and TTL decrement.
“The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero.” [RFC 791] “A host MUST verify the IP header checksum on every received datagram and silently discard every datagram that has a bad checksum.” [RFC 1122]
A C implementation of the checksum algorithm can be found in [RFC 1071]. Unfortunately this implementation is incorrect. A corrected version looks like this:

```
{
        /* Compute Internet Checksum for "count" bytes
         *         beginning at location "addr", which is a
         *         pointer to unsigned short (16 bits).
         */
    register long sum = 0;

    while( count > 1 )  {
        /*  This is the inner loop */
            sum += *addr++;
            count -= 2;
    }

        /*  Add left-over byte, if any */
    if( count > 0 )
            sum += * (unsigned char *) addr;

        /*  Fold 32-bit sum to 16 bits */
    while (sum>>16)
        sum = (sum & 0xffff) + (sum >> 16);

    checksum = ~sum;
}
```

Source Address,
Destination Address

IP address of the interface of the source host that sends this packet; IP address of the destination host.

Options

Setting of IP options is optional, but not their implementation. In our context, though, it is legitimate to ignore the next-higher-level protocol’s request to set IP options, as well as to ignore IP options in incoming datagrams. This *is* a violation of the standard, but the use of IP options can be considered non-necessary today. IP options can, though, without any effort be passed to the next higher-layer protocol, if simply the whole IP datagram is passed.

It is necessary, of course, to handle incoming IP packets with IP options correctly, even when ignoring the option values. The full length of the option fields is determined through the Header Length field.

1.3 ICMP messages

An ICMP message is embedded in an IP packet. The IP header field "protocol" is 1 for ICMP, Type of Service depends on the type of the message; all other fields are set as determined by the usual considerations. ICMP messages are of different *types*; some types carry additionally different *codes*.

ICMP messages can be *error messages* or *query messages*. ICMP error messages must never be sent in response to

- an ICMP error message (as opposed to an ICMP query message)
- an IP datagram sent as IP broadcast or link-layer broadcast
- a fragment other than the first
- a datagram with a source address that does not define a single host, i. e. a multicast or broadcast address, a network address, a loopback address, or a class E address.

According to http://www.isi.edu/in-notes/iana/assignments/icmp-parameters, the ICMP types shown in the table below are currently defined. The code in the "Relevant" column shows whether the specified type is to be implemented in the mobile phone context (•) or not (–). The relevant types are described in the following sections.

Type	Name	Relevant
0	Echo Reply	•
1	Unassigned	–
2	Unassigned	–
3	Destination Unreachable	•
4	Source Quench	–
5	Redirect	–
6	Alternate Host Address	–
7	Unassigned	–
8	Echo	•
9	Router Advertisement	–
10	Router Selection	–
11	Time Exceeded	•
12	Parameter Problem	–
13	Timestamp	–
14	Timestamp Reply	–
15	Information Request	–
16	Information Reply	–
17	Address Mask Request	–
18	Address Mask Reply	–
19	Reserved (for Security)	–
20-29	Reserved (for Robustness Experiment)	–
30	Traceroute	–
31	Datagram Conversion Error	–
32	Mobile Host Redirect	–
33	IPv6 Where-Are-You	–
34	IPv6 I-Am-Here	–
35	Mobile Registration Request	–

36	Mobile Registration Reply	–
37	Domain Name Request	–
38	Domain Name Reply	–
39	SKIP	–
40	Photuris	–
41-255	Reserved	–

Figure 1-7: ICMP Message Types

An ICMP error message is always sent with a Type Of Service (TOS field of the IP header) of 0. An ICMP request message may be sent with any value in the TOS field. An ICMP reply message is sent with the same value in the TOS field as was used in the corresponding ICMP request message.

We need not send any ICMP code not marked as relevant in the table above. Incoming ICMP messages with these codes can be ignored.

1.3.1 Destination unreachable (Error message)

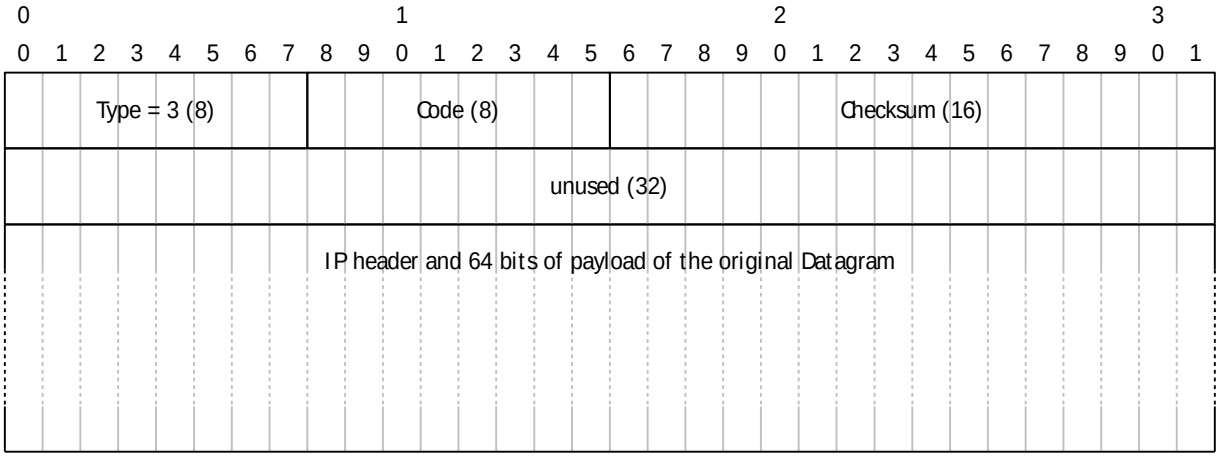


Figure 1-8: ICMP Header Destination unreachable

The ICMP type "destination unreachable" has the value 3. The different codes are described in the table below. When receiving these messages, not all of these codes need not be maintained as separate error conditions in the mobile phone context; the codes can be grouped into three different error conditions, which can then be passed to the application:

CONNECTION_REFUSED
: The IP packet reached the destination host, but on the destination host there is no entity listening for the specified transport protocol and the specified port. ("Protocol Unreachable" and "Port Unreachable")

MESSAGE_TOO_LONG
: The IP packet could not be routed to the destination host because for the next hop link the packet would have to be fragmented, but the DF (don't fragment) flag is set in the packet. ("Fragmentation Needed and DF Set")

NO_ROUTE_TO_HOST
: The IP packet could not be routed to the destination host for some other reason. (all other codes)

Code	Value	Description	Condition
Network Unreachable	0	generated by a router if a forwarding path (route) to the destination network is not available	NO_ROUTE_TO_HOST
Host unreachable	1	generated by a router if a forwarding path (route) to the destination host on a directly connected network is not available	NO_ROUTE_TO_HOST
Protocol Unreachable	2	generated if the transport protocol designated in a datagram is not supported in the transport layer of the final destination	CONNECTION_REFUSED
Port Unreachable	3	generated if the designated transport protocol (e.g., UDP) is unable to demultiplex the datagram in the transport layer of the final destination but has no protocol mechanism to inform the sender	CONNECTION_REFUSED
Fragmentation Needed and DF Set	4	generated if a router needs to fragment a datagram but cannot since the DF flag is set	MESSAGE_TOO_LONG
Source Route Failed	5	generated if a router cannot forward a packet to the next hop in a source route option	NO_ROUTE_TO_HOST
Destination Network Unknown	6	This code SHOULD NOT be generated since it would imply on the part of the router that the destination network does not exist (net unreachable code 0 SHOULD be used in place of code 6)	NO_ROUTE_TO_HOST
Destination Host Unknown	7	generated only when a router can determine (from link layer advice) that the destination host does not exist	NO_ROUTE_TO_HOST
Network Unreachable For Type Of Service	11	generated by a router if a forwarding path (route) to the destination network with the requested or default TOS is not available	NO_ROUTE_TO_HOST
Host Unreachable For Type Of Service	12	generated if a router cannot forward a packet because its route(s) to the destination do not match either the TOS requested in the datagram or the default TOS (0)	NO_ROUTE_TO_HOST
Communication Administratively Prohibited	13	generated if a router cannot forward a packet due to administrative filtering	NO_ROUTE_TO_HOST
Host Precedence Violation	14	Sent by the first hop router to a host to indicate that a requested precedence is not permitted for the particular combination of source/destination host or network, upper layer protocol, and source/destination port	NO_ROUTE_TO_HOST
Precedence cutoff in effect	15	The network operators have imposed a minimum level of precedence required for operation, the datagram was sent with a precedence below this level	NO_ROUTE_TO_HOST

Table 1-3: ICMP Codes for Destination Unreachable [RFC 1812]

1.3.2 Time Exceeded (Error message)

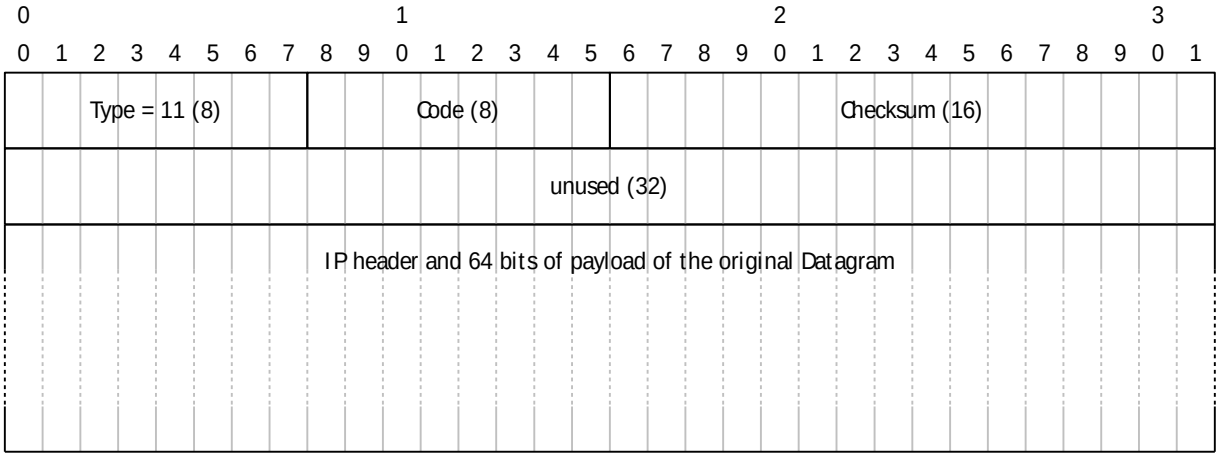


Figure 1-9: ICMP Header Time Exceeded

The "Time Exceeded" type has the value 11. When receiving these messages, both codes must be reported to the transport layer as separate error conditions.

Code	Value	Description	Comment
TTL exceeded in transit	0	generated by a router when it has dropped an IP packet because the TTL field of the packet is zero	The mobile should not throw away IP packets with a TTL of zero that are addressed to the mobile; however, it *must* drop them when it routes IP packets between a locally attached device and the Internet, and send a TTL exceeded message.
Fragment reassembly time exceeded	1	generated by the destination host when it has not received all fragments of a datagram during the configured reassembly time	

Table 1-4: ICMP Codes for Time Exceeded

1.3.3 Echo/Echo Reply (Query message)

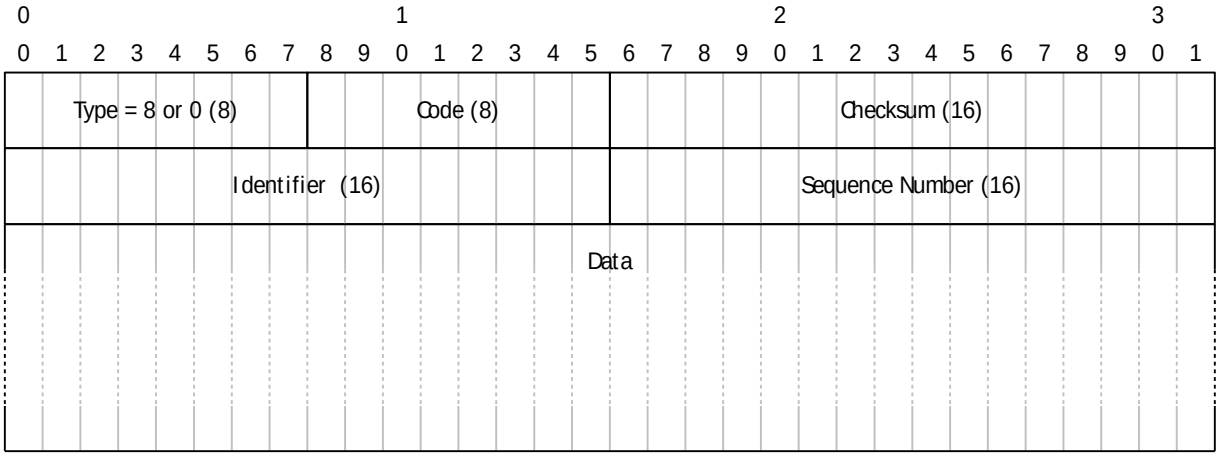


Figure 1-10: ICMP Header Echo/Echo Reply

The "Echo" type (value 8) requests the destination host to send back an ICMP message of type "Echo reply" (0). The Code field is zero for both types. "The identifier and sequence number may be used by the echo sender to aid in matching the replies with the echo requests. For example, the identifier might be used like a port in TCP or UDP to identify a session, and the sequence number might be incremented on each echo request sent. The echoer returns these same values in the echo reply." [RFC 792]

The Data is arbitrary data of arbitrary length as chosen by the sender of the Echo request; the data received must be returned in the Echo Reply message.

The ICMP Echo and Echo Reply messages are used by the "ping" program.

1.4 IP Algorithm

This section is not intended as a comprehensive description of the IP layer algorithms, but rather as a general outline.

The numbered headlines in ths section are taken from [McKusick et al. 1996] (with omissions).

1.4.1 IP output

The IP layer is called to send a message for ICMP or some upper layer. This message may already have a pseudo IP header attached with some values filled in (e. g. destination address).

1. Insert any IP options.

(We omit this step.)

2. Fill in the remaining header fields (IP version, zero offset, header length, and a new packet identification) if the packet contains an IP pseudoheader.

3. Determine the route (i. e., outgoing interface and next-hop destination).

If we have only a WAP browser running on the mobile, we need not worry about routes at all – there is only one route via the air interface. If an IP-capable device (i. e. a host) is attached to one of the local ports (USART, IRDA, Bluetooth, etc.), we need to remember this port as a route

to this host (host route); all packets destined to other hosts are routed via the air interface (default route).

4. Check whether the destination is a multicast address.

For now, we need not treat multicast addresses any different from unicast addresses.

5. Check whether the destination is a broadcast address.

The mobile won't have any broadcast-capable IP interfaces, but it can send broadcast packets over the default route. It must not send packets with the broadcast address 255.255.255.255.

6. If the packet size is no larger than the maximum packet size for the outgoing interface, compute the checksum and call the interface output routine.

7. If the packet size is larger than the maximum packet size for the outgoing interface, break the packet into fragments and send each in turn.

On fragmentation, the Offset, Flags, and Total Length fields must be adjusted.

1.4.2 IP input

The IP layer is called to handle a packet that has been received by the link layer.

1. Verify that the packet is at least as long as an IP header, and ensure that the header is contiguous.

If it is not, drop the packet. (I am not sure what McKusick et al. mean with "contiguous".)

2. Checksum the header of the packet, and discard the packet if there is an error.

3. Verify that the packet is at least as long as the header indicates, and drop the packet if it is not.

4. Process any IP options in the header.

Here: Ignore any IP options in the header.

5. Check whether the packet is for this host. If it is, continue processing the packet. If it is not, and if doing packet forwarding, try to forward the packet. Otherwise, drop the packet.

If the packet is for a host attached to the mobile:

> Decrement the packet's TTL. If it is zero:
>
> > Drop the packet and send an ICMP time exceeded message with the code 0 (TTL exceeded in transit).
>
> Otherwise forward the packet over the link to this host.

If the packet is for the mobile:

> Continue with step 6.

If the packet is neither for an attached host nor for the mobile itself, we return an ICMP destination unreachable message with code 7 (Destination host unknown).

6. If the packet has been fragmented, keep it until all fragments are received and reassembled, or until it is too old to keep.

If this is the first packet, start the reassembly timer with the reassemby timeout value. If it expires before all fragments have arrived, drop the packet and return an ICMP time exceeded message with code 1 (Fragmentation reassembly time exceeded).

7. Pass the packet to the input routine of the next-higher-level protocol.

1.5 Requirement Levels

[RFC 1122] lists a couple of features that an IP/ICMP implementation *must*, *should*, *may*, *should not*, and *must not* implement. The meaning of these terms is defined in RFC 2119 ("Key words for use in RFCs to Indicate Requirement Levels", Scott Bradner, March 1997):

1. MUST This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.
2. MUST NOT This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.
3. SHOULD This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
4. SHOULD NOT This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
5. MAY This word, or the adjective "OPTIONAL", mean that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

The standard RFC 1122 is targeted at general-purpose Internet host. The situation with the mobile phone is vastly different, so the standard is not applicable ("n. a.") to all issues mentioned in RFC 1122. The restrictions are:

- Sending IP options is not supported; incoming options are passed to the application layer, but not evaluated in the IP layer.
- The only ICMP types supported are 0, 3, 8, 11 (Echo Request and Reply, Destination Unreachable, Time Exceeded).
- Multihoming is not supported, but will probably be in a later version, when a locally attached IP-capable device is supported.
- Netmasks are not supported. (There is currently only one point-to-point link to the ISP, so netmasks are not needed.)

The following table contains the list of requirement levels for IP from [RFC 1122].

Req. Level	Feature	We do	Comment
MUST	Implement IP and ICMP	•	
MUST	Handle remote multihoming in application layer	•	i. e. the IP layer does not care
MAY	Support local multihoming	–	not at the moment
MUST	Meet gateway specs if forward datagrams	–	n. a.
MUST	Configuration switch for embedded gateway	–	n. a.
MUST	Config switch default to non-gateway	–	n. a.
MUST NOT	Auto-config based on number of interfaces	–	n. a.
SHOULD	Able to log discarded datagrams	–	not necessary or desirable in a mobile
SHOULD	Record in counter	–	
MUST	Silently discard Version != 4	•	
MUST	Verify IP checksum, silently discard bad dgram	•	
	Addressing:		
MUST	Subnet addressing (RFC-950)	–	n. a.
MUST	Src address must be host's own IP address	•	
MUST	Silently discard datagram with bad dest addr	•	
MUST	Silently discard datagram with bad src addr	•	
MUST	Support reassembly	•	
MAY	Retain same Id field in identical datagram		
	TOS:		
MUST	Allow transport layer to set TOS	–	obsolete and unnecessary
SHOULD	Pass received TOS up to transport layer	•	this is easier than not doing it
SHOULD NOT	Use RFC-795 link-layer mappings for TOS	–	
	TTL:		
MUST NOT	Send packet with TTL of 0	–	
MUST NOT	Discard received packets with TTL < 2	–	
MUST	Allow transport layer to set TTL	•	
MUST	Fixed TTL is configurable	•	at ROM programming time
	IP Options:		
MUST	Allow transport layer to send IP options	–	no need to set IP options for our applications
MUST	Pass all IP options rcvd to higher layer	•	whole IP packet is passed to transport layer
MUST	IP layer silently ignore unknown options	•	
MAY	Security option	–	
SHOULD NOT	Send Stream Identifier option	–	
MUST	Silently ignore Stream Identifier option	•	
MAY	Record Route option	–	
MAY	Timestamp option	–	
	Source Route Option:		
MUST	Originate & terminate Source Route options	–	
MUST	Datagram with completed SR passed up to TL	•	whole IP packet is passed to transport layer
MUST	Build correct (non-redundant) return route	–	
MUST NOT	Send multiple SR options in one header	–	
	ICMP:		
MUST	Silently discard ICMP msg with unknown type	–	
MAY	Include more than 8 octets of orig datagram	–	
MUST	Included octets same as received	•	
MUST	Demux ICMP Error to transport protocol	•	

SHOULD	Send ICMP error message with TOS=0	•	
	Send ICMP error message for:		
MUST NOT	- ICMP error msg	–	
MUST NOT	- IP b'cast or IP m'cast	–	
MUST NOT	- Link-layer b'cast	–	
MUST NOT	- Non-initial fragment	–	
MUST NOT	- Datagram with non-unique src address	–	
MUST	Return ICMP error msgs (when not prohibited)	•	
	Dest Unreachable:		
SHOULD	Generate Dest Unreachable (code 2/3)	•	
MUST	Pass ICMP Dest Unreachable to higher layer	•	
SHOULD	Higher layer act on Dest Unreach		this is up to the higher layer
MUST	Interpret Dest Unreach as only hint		this is up to the higher layer
	Redirect:		
SHOULD NOT	Host send Redirect	–	
MUST	Update route cache when recv Redirect	–	n. a.
MUST	Handle both Host and Net Redirects	–	n. a.
SHOULD	Discard illegal Redirect	•	
	Source Quench:		
MAY	Send Source Quench if buffering exceeded	–	
MUST	Pass Source Quench to higher layer	–	obsolete by modern flow-control algorithms
SHOULD	Higher layer act on Source Quench	–	obsolete by modern flow-control algorithms
MUST	Time Exceeded: pass to higher layer	•	
	Parameter Problem:		
SHOULD	Send Parameter Problem messages	–	due to performance and memory reasons
MUST	Pass Parameter Problem to higher layer	–	
MAY	Report Parameter Problem to user	–	user is not interested
	ICMP Echo Request or Reply:		
MUST	Echo server and Echo client	•	
SHOULD	Echo client	•	
MAY	Discard Echo Request to broadcast address	•	
MAY	Discard Echo Request to multicast address	•	
MUST	Use specific-dest addr as Echo Reply src	•	
MUST	Send same data in Echo Reply	•	
MUST	Pass Echo Reply to higher layer	•	
SHOULD	Reflect Record Route, Time Stamp options	–	IP options not supported
MUST	Reverse and reflect Source Route option	–	IP options not supported
SHOULD NOT	ICMP Information Request or Reply:	–	
MAY	ICMP Timestamp and Timestamp Reply:	–	
SHOULD	Minimize delay variability	–	
MAY	Silently discard b'cast Timestamp	•	
MAY	Silently discard m'cast Timestamp	•	
MUST	Use specific-dest addr as TS Reply src	–	n. a.
SHOULD	Reflect Record Route, Time Stamp options	–	n. a.
MUST	Reverse and reflect Source Route option	–	n. a.
MUST	Pass Timestamp Reply to higher layer	–	
MUST	Obey rules for "standard value"	–	n. a.
	ICMP Address Mask Request and Reply:		

MUST	Addr Mask source configurable	–	
MUST	Support static configuration of addr mask	–	
MAY	Get addr mask dynamically during booting	–	
MAY	Get addr via ICMP Addr Mask Request/Reply	–	
MUST	Retransmit Addr Mask Req if no Reply	–	
SHOULD	Assume default mask if no Reply	–	
MUST	Update address mask from first Reply only	–	
SHOULD	Reasonableness check on Addr Mask	–	
MUST NOT	Send unauthorized Addr Mask Reply msgs	–	
MUST	Explicitly configured to be agent	–	
SHOULD	Static config=> Addr-Mask-Authoritative flag	–	
MUST	Broadcast Addr Mask Reply when init.	–	
	ROUTING OUTBOUND DATAGRAMS:		
MUST	Use address mask in local/remote decision	–	n. a.
MUST	Operate with no gateways on conn network	–	n. a.
MUST	Maintain "route cache" of next-hop gateways	•	configured by PPP
SHOULD	Treat Host and Net Redirect the same	•	i. e. not
MUST	If no cache entry, use default gateway	•	
MUST	Support multiple default gateways	–	
MAY	Provide table of static routes	•	i. e. one static default route
MAY	Flag: route overridable by Redirects	–	
MAY	Key route cache on host, not net address	–	
SHOULD	Include TOS in route cache	–	
MUST	Able to detect failure of next-hop gateway	•	
SHOULD NOT	Assume route is good forever	–	until PPP gives an error
MUST NOT	Ping gateways continuously	–	
MUST	Ping only when traffic being sent		will not ping gateway
MUST	Ping only when no positive indication		will not ping gateway
SHOULD	Higher and lower layers give advice	•	PPP
MUST	Switch from failed default g'way to another	–	
MUST	Manual method of entering config info	–	configured by PPP
	REASSEMBLY and FRAGMENTATION:		
MUST	Able to reassemble incoming datagrams	•	
MUST	At least 576 byte datagrams	•	
SHOULD	EMTU_R configurable or indefinite	•	
MUST	Transport layer able to learn MMS_R	•	configured by PPP
MUST	Send ICMP Time Exceeded on reassembly timeout	•	
SHOULD	Fixed reassembly timeout value	•	
MUST	Pass MMS_S to higher layers	•	
MAY	Local fragmentation of outgoing packets	•	
MUST	Else don't send bigger than MMS_S	–	n. a.
SHOULD	Send max 576 to off-net destination	–	common value is 1500 now
MAY	All-Subnets-MTU configuration flag	–	n. a.
	MULTIHOMING:		
SHOULD	Reply with same addr as spec-dest addr	•	
MUST	Allow application to choose local IP addr	•	if applicable
MAY	Silently discard d'gram in "wrong" interface	•	
MAY	Only send d'gram through "right" interface	•	
	SOURCE-ROUTE FORWARDING:		
MAY	Forward datagram with Source Route option	–	
MUST	Obey corresponding gateway rules	–	

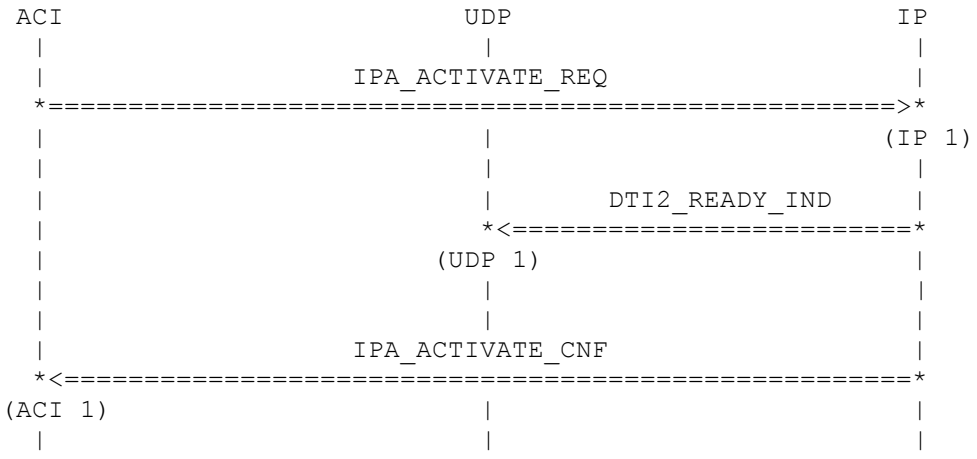
MUST	Update TTL by gateway rules	–	
MUST	Able to generate ICMP err code 4, 5	–	4 only
MAY	IP src addr not local host	–	
MUST	Update Timestamp, Record Route options	–	
MUST	Configurable switch for non-local SRing	–	
MUST	Defaults to OFF	•	
MUST	Satisfy gwy access rules for non-local SRing	–	
SHOULD	If not forward, send Dest Unreach (cd 5)	–	
	BROADCAST:		
MUST NOT	Broadcast addr as IP source addr	–	
SHOULD	Receive 0 or -1 broadcast formats OK	–	
MAY	Config'ble option to send 0 or -1 b'cast	–	
SHOULD	Default to -1 broadcast	–	n. a.
MUST	Recognize all broadcast address formats	–	n. a.
MUST	Use IP b'cast/m'cast addr in link-layer b'cast	–	n. a.
SHOULD	Silently discard link-layer-only b'cast dg's	•	
SHOULD	Use Limited Broadcast addr for connected net	–	n. a.
	MULTICAST:		
SHOULD	Support local IP multicasting (RFC-1112)	–	
MAY	Support IGMP (RFC-1112)	–	
SHOULD	Join all-hosts group at startup	–	
SHOULD	Higher layers learn i'face m'cast capability	–	
	INTERFACE:		
MUST	Allow transport layer to use all IP mechanisms	–	all except options
MUST	Pass interface ident up to transport layer	•	
MUST	Pass all IP options up to transport layer	•	
MUST	Transport layer can send certain ICMP messages	•	
MUST	Pass spec'd ICMP messages up to transp. layer	•	
MUST	Include IP hdr+8 octets or more from orig.	•	
SHOULD	Able to leap tall buildings at a single bound	–	mobile can be thrown by the user

Table 1-5: Summary of the Requirements for Internet Hosts (IP and ICMP)

2 Protocol

2.1 Deactivated State

2.1.1 Activation of IP

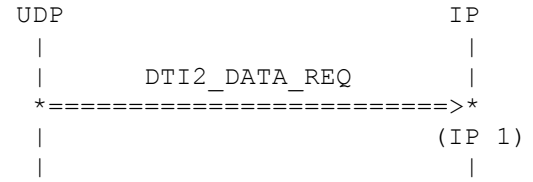


(IP 1)
The IP entity receives the IPA_ACTIVATE_REQ and is switched to the activated state. At this point every related transport layer must be known.

(UDP 1)
IP informs UDP that it is ready to receive data. (Occurs with each transport layer)

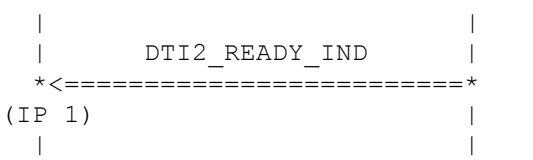
(ACI 1)
IP informs ACI that the entity is activated.

2.1.2 Reception of DTI2_DATA_REQ in Deactivated State



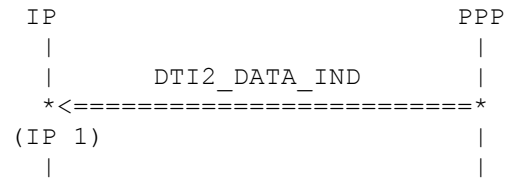
(IP 1)
The IP entity receives the DTI2_DATA_REQ from a transport layer and ignores it.

2.1.3 Reception of DTI2_READY_IND in Deactivated State



(IP 1)
The IP entity receives the DTI2_READY_IND from an interface layer and stores it.

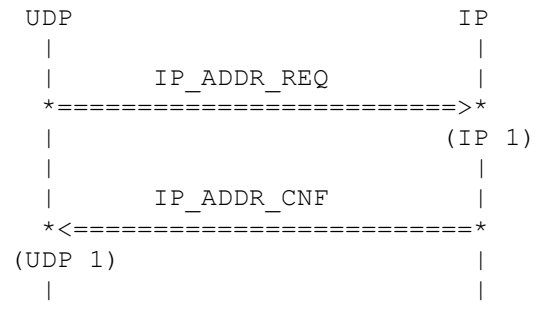
2.1.4 Reception of DTI2_DATA_IND in Deactivated State



(IP 1)

The IP entity receives the DTI2_DATA_IND from an interface layer and ignores it.

2.1.5 Reception of IP_ADDR_REQ in Deactivated State



(IP 1)

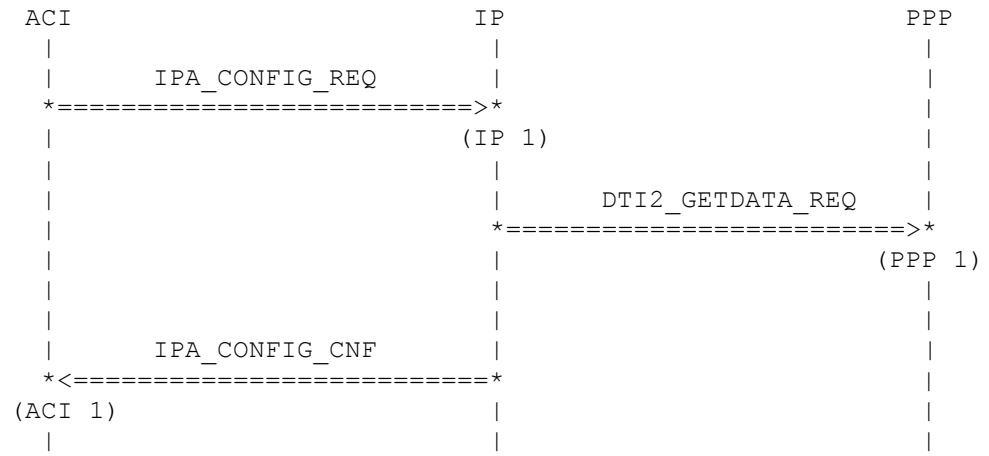
The IP entity receives the IP_ADDR_REQ.

(UDP 1)

IP sends an IP_ADDR_CNF with the err field set to IP_NOT_READY.

2.2 Activated and not-configured State

2.2.1 Configuration of an Interface (up)



(IP 1)

ACI configures a transport layer interface (ip, mtu, ttl, netmask, UP)

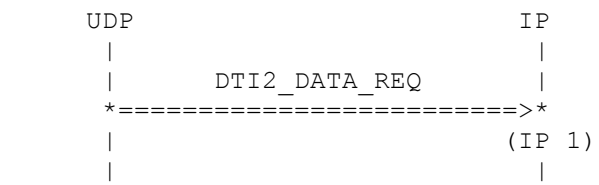
(PPP 1)

IP indicates that the entity is ready to get the next DTI2_DATA_IND primitive from the transport layer.

(ACI 2)

IP confirms the configuration and is now in configured and activated state.

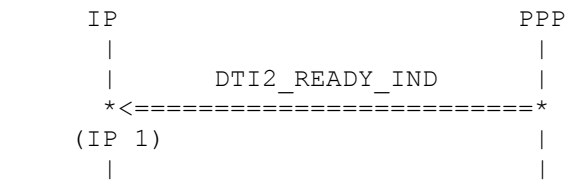
2.2.2 Reception of DTI2_DATA_REQ in not-configured State



(IP 1)

The IP entity receives the DTI2_DATA_REQ from a transport layer and ignores it.

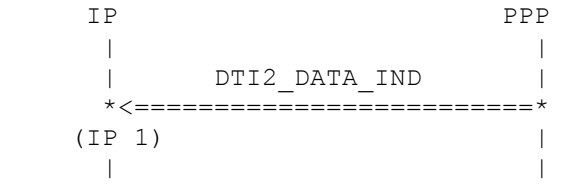
2.2.3 Reception of DTI2_READY_IND in not-configured State



(IP 1)

The IP entity receives the DTI2_READY_IND from an interface layer and stores it.

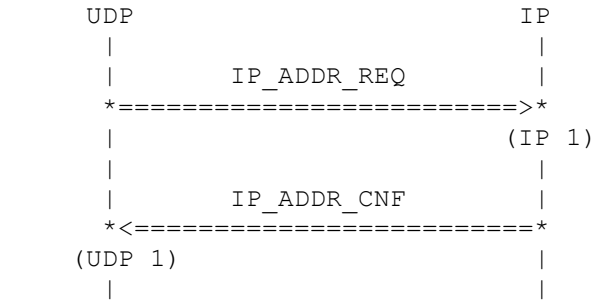
2.2.4 Reception of DTI2_DATA_IND in not-configured State



(IP 1)

The IP entity receives the DTI2_DATA_IND from an interface layer and ignores it.

2.2.5 Reception of IP_ADDR_REQ in not-configured State

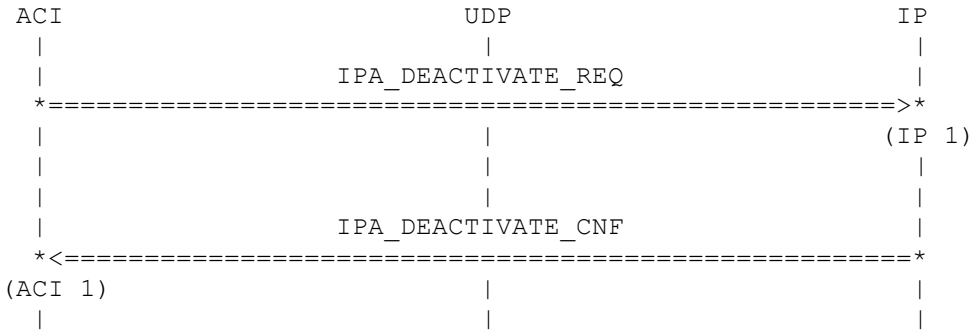


(IP 1)

The IP entity receives the IP_ADDR_REQ.

(UDP 1)
IP sends an IP_ADDR_CNF with the err field set to IP_NOT_READY.

2.2.6 Deactivation of IP

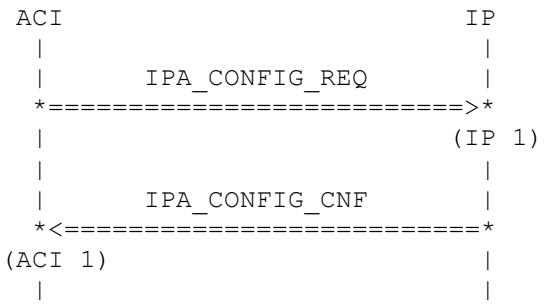


(IP 1)
The IP entity receives the IPA_DEACTIVATE_REQ and is switched to the deactivated state.

(ACI 1)
IP informs ACI that the entity is deactivated.

2.3 Configured State

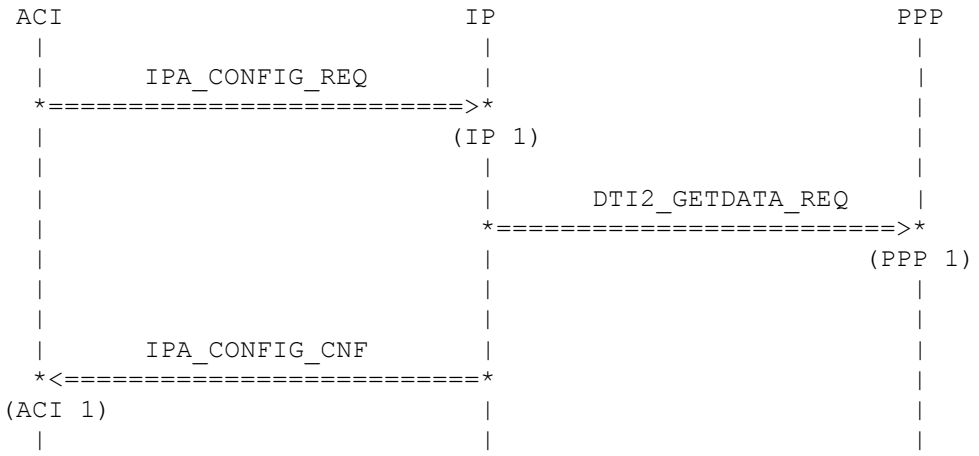
2.3.1 Configuration of an Interface (down)



(IP 1)
ACI configures a transport layer interface (..., DOWN)

(ACI 2)
IP confirms the configuration and ignores all occurring DTI2_DATA_IND from the closed interface layer.
If no other interface is configured, the unit goes in the not-configured state.

2.3.2 Configuration of an Interface (up)



(IP 1)

ACI configures a transport layer interface (ip, peer_ip, mtu, ui, netmask, UP)

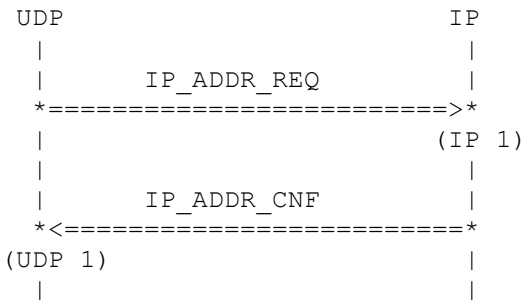
(PPP 1)

IP indicates that the entity is ready to get the next DTI2_DATA_IND primitive from the transport layer.

(ACI 2)

IP confirms the configuration.

2.3.3 Reception of IP_ADDR_REQ in Configured State



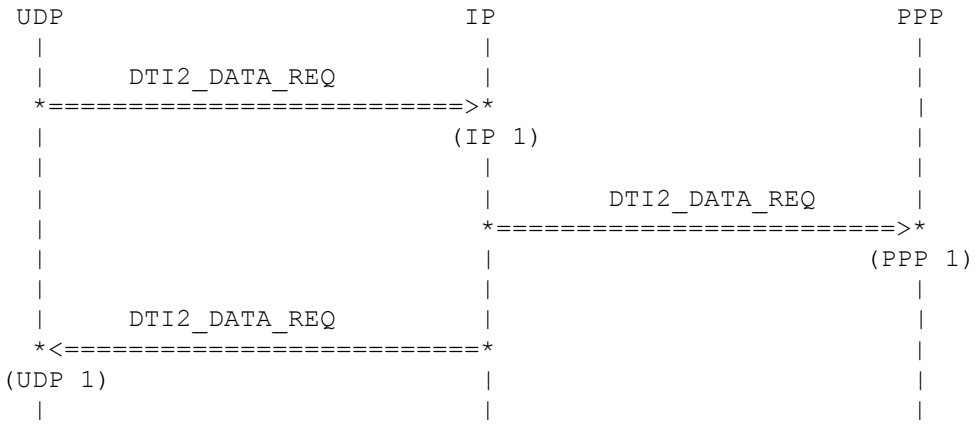
(IP 1)

The IP entity receives the IP_ADDR_REQ.

(UDP 1)

IP sends an IP_ADDR_CNF with the src_addr field set to the correct source address or the err field set to IP_ADDR_NOROUTE.

2.3.4 Reception of DTI2_DATA_REQ in Configured State



(IP 1)

IP receives a datagram or an ICMP message from the transport layer.

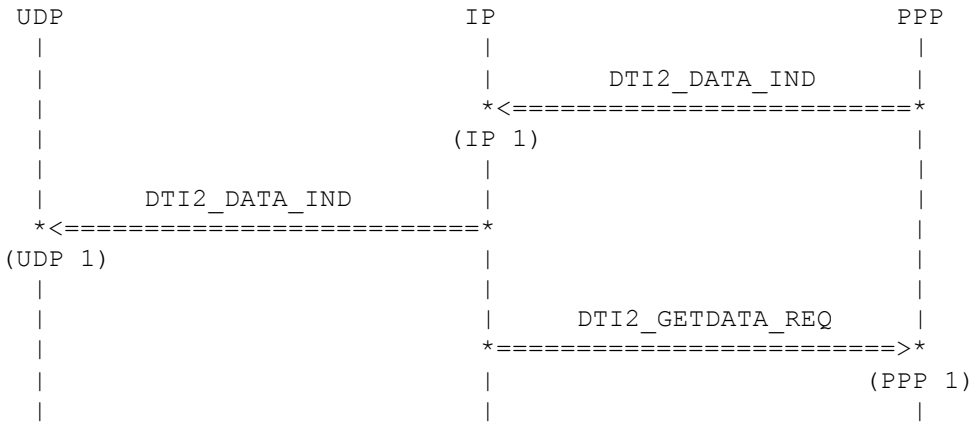
(PPP 1)

IP determines the relevant interface and sends an IP datagram via DTI2_DATA_REQ.

(UDP 1)

IP indicates that the entity is ready to get the next DTI2_DATA_REQ primitive from this transport layer.

2.3.5 Reception of DTI2_DATA_REQ in Configured State



(IP 1)

IP receives a datagram from the interface layer.

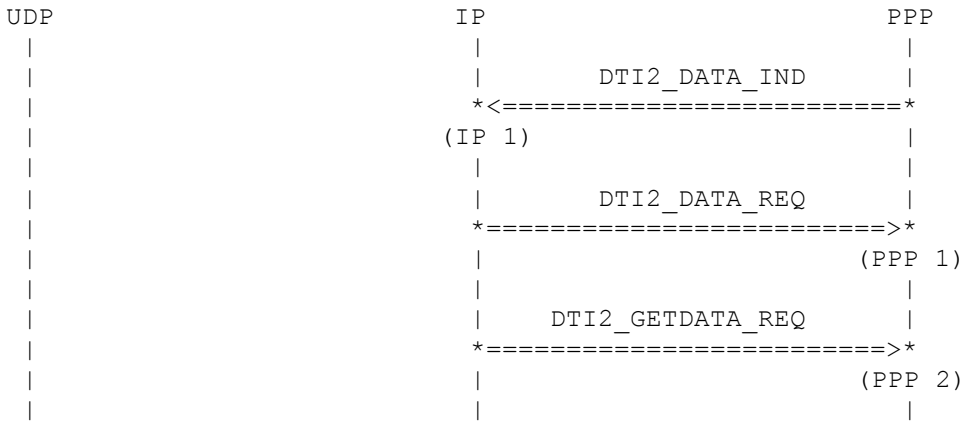
(UDP 1)

IP determines the relevant transport layer and sends an IP datagram via DTI2_DATA_REQ.

(PPP 2)

IP indicates that the entity is ready to get the next DTI2_DATA_IND primitive.

2.3.6 Reception of DTI2_DATA_REQ in Configured State (ERROR/ICMP)



(IP 1)

IP receives a datagram from the interface layer.

(PPP 1)

IP sends an ICMP Message via DTI2_DATA_REQ.

(PPP 2)

IP indicates that the entity is ready to get the next DTI2_DATA_IND primitive.

Appendices

A. Acronyms

DS-WCDMA	Direct Sequence/Spread Wideband Code Division Multiple Access

B. Glossary

International Mobile Telecommunication 2000 (IMT-2000/ITU-2000)	Formerly referred to as FPLMTS (Future Public Land-Mobile Telephone System), this is the ITU's specification/family of standards for 3G. This initiative provides a global infrastructure through both satellite and terrestrial systems, for fixed and mobile phone users. The family of standards is a framework comprising a mix/blend of systems providing global roaming. URL: http://www.itu.int/home