IPv6 Verification Tool Users Manual

Rev.2.3

This document applies to IPv6 Verification Tool Release 1.0.

Contents

1. Summary of the Tester

1.1 Test Environment

With the presumed test environment, tests shall be conducted for "routers and multi-home hosts with multiple interfaces," and other targets are handled as a subset.

Tests shall be fully automated, and manual work shall not be required halfway. Basically, the test will be conducted by throwing a packet and checking the response. However, if setup is changed, and a packet is actively sent from the Target, there will be a need for input and output from the Target Console. Therefore, input and output from the Serial Line must be supported.

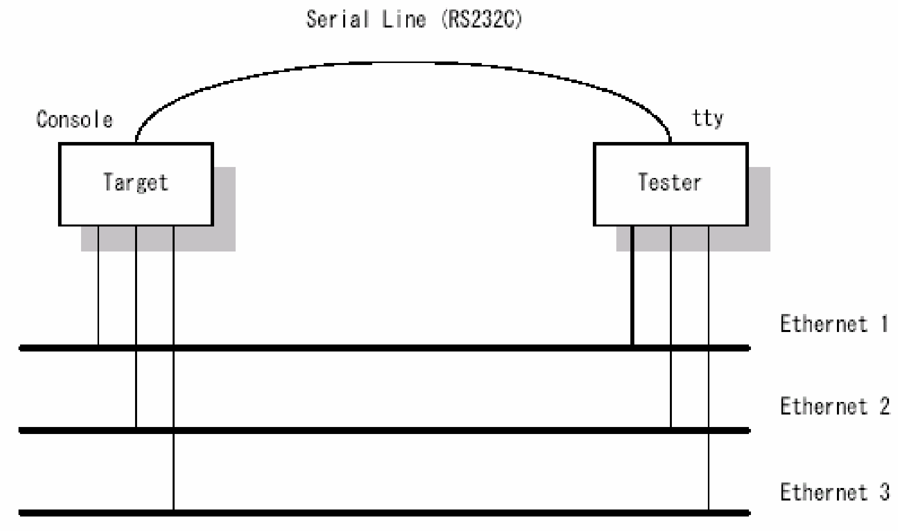


Figure 1 Test Environment Image

1.2 Operation Requirements

1.2.1 Hardware Requirements

Either of the following OS shall be operated:

- FreeBSD2.2.8
- FreeBSD3.4

More than one Ethernet network interface is required.

One serial interface is required.

1.2.2 Software Requirements

The software for implementing the Tester shall be as follows:

Table 1 Operation Requirement (Software)

Items	Version, etc.	Notes
OS	Patch* is attached to FreeBSD2.2.8 or 3.4	IPv4 only
OpenSSL Library	0.9.2b	For IPsec
Perl	5.005_02	For Sequencer
Perl Lib-		
- Expert	Expect Module	For remote control
- IO-Stty	IO-Stty	For remote control
- IO-Tty	IO-Tty	For remote control

* Patch to change the Ether Frame source address freely

1.3 Install

1) Install the FreeBSD
2) Attach a Patch on OS
3) Install related libraries
4) Install tools
5) Install the test script
6) Connect the Tester and Target
7) Set the config file

1.4 Directory Configuration

This tool is installed under "/usr/local/v6eval" (hereafter called SV6EVALROOT), and the following directory configuration is taken:

```
$V6EVALROOT/bin/
            /ct/
            /etc/
            /doc/
```

bin directory

- Directory in which a binary for execution is stored except for the Sequencer
- pkt{send, recv, buf, ctl}, remote control file, auto execution

ct directory

- Directory in which a set of the conformance test is stored
- Under this directory a directory is created by the major item, and respective test scripts are placed in it.

doc directory

- Directory in which documents such as README, INSTALL, and TODO are stored

etc directory

- Directory in which various config files are stored
- A sample is put in immediately after installation

2. Config File Format

2.1 Tester Node Config File

2.1.1 Purpose

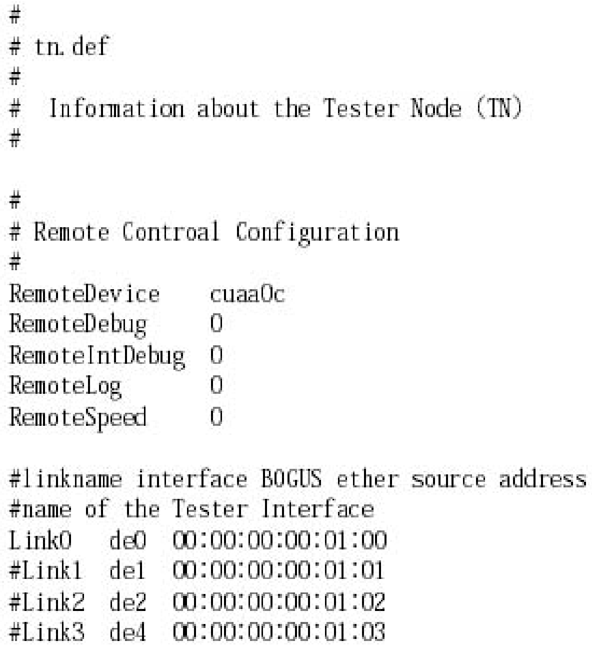
The purpose of this file is to describe the following tester-specific configuration.

- Combination of the tester's I/F name and virtual I/F name
- File path for execution
- Information to respond to auto set for sending and receiving a packet
- Information for report output

2.1.2 Syntax

- Items may be in any order.
- One item per line. A line feed cannot be used in the middle of a line.
- If an entry is duplicated, the latter entry overrides the former one.
- A line beginning with a hash symbol (#) can be used as a comment. (The system ignores lines beginning with a hash symbol and blank lines.)

2.1.3 Example



```
#
# tn.def
#
#  Information about the Tester Node (TN)
#

#
# Remote Controal Configuration
#
RemoteDevice    cuaa0c
RemoteDebug     0
RemoteIntDebug  0
RemoteLog       0
RemoteSpeed     0

#linkname interface BOGUS ether source address
#name of the Tester Interface
Link0   de0  00:00:00:00:01:00
#Link1  de1  00:00:00:00:01:01
#Link2  de2  00:00:00:00:01:02
#Link3  de4  00:00:00:00:01:03
```

2.1.4 Setup items

The following table shows the configurable items in this file.

Table 2 Setup Items

Entry Name	Omission	Description
socketpath	Possible	· A directory is specified for creating the UNIX domain socket, with which the buffer daemon and the program for sending and receiving data communicate. · This directory shall be created before the test program is executed. · If it is omitted, make it /tmp.
LinkN	Impossible	· Numerals are used for N. · The entry "Link0" is required. · After that, the system writes the I/F name and the MAC address. · The items I/F name, MAC Addr, … are mandatory.
filter	Possible	· "IPv6" is specified here only if the IPv6 packet is used for the received packet in the test.
RemoteDevice	Possible	· Tip device name · If it is omitted, make it cuaa0c.
RemoteDebug	Possible	· If it is omitted, make it 0. · Details are unclear.
RemoteIntDebug	Possible	· If it is omitted, make it 0. · Details are unclear.
RemoteLog	Possible	· If it is omitted, make it 0. · If 1 is specified, the details of communication are recorded in the log.
RemoteSpeed	Possible	· If it is omitted, make it 0. · The interval of characters to be sent is specified in seconds. · Decimal points may be used.
RemoteLogout	Possible	

2.2 Node under Test Config File

2.2.1 Purpose

The purpose of this file is to describe the following target-specific configuration.

- Combination of the tester's I/F name and virtual I/F name
- A clue concerning which test to be conducted
- Information to respond to auto set for sending and receiving a packet
- Information for report output

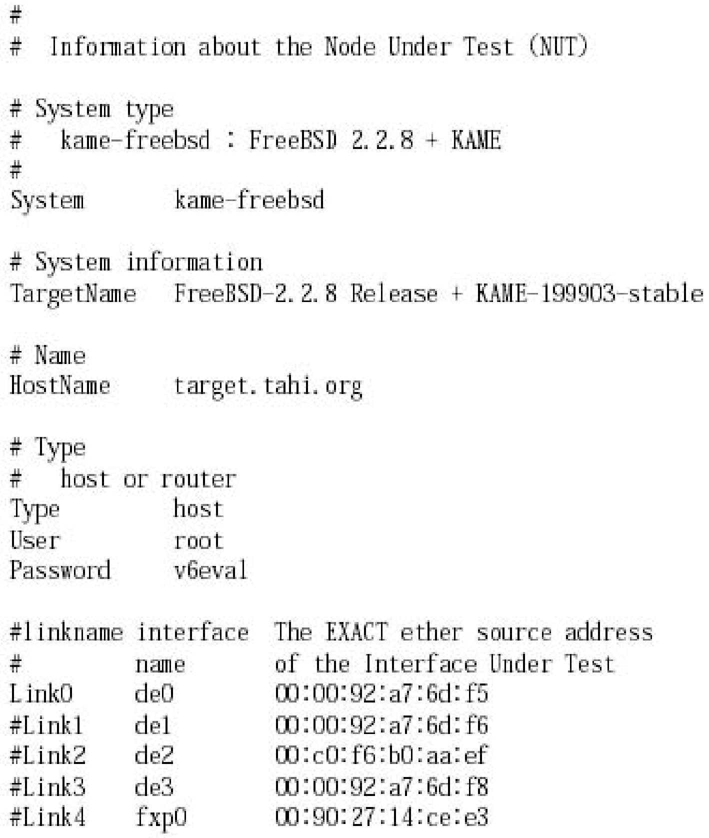
2.2.2 Points to be considered

- The details of this file must not be changed even if the tester environment changes.
- One target file for each target
- The IPv4 address for the target of the test shall be fixed, and it is embedded in the packet definition file.

2.2.3 Syntax

- The items may be specified in any order.
- One item per line. A line feed cannot be used in the middle of a line.
- A line beginning with a hash symbol (#) can be used as a comment. (The system ignores lines beginning with a hash symbol and blank lines.)

2.2.4 Example



```
#
#  Information about the Node Under Test (NUT)

# System type
#   kame-freebsd : FreeBSD 2.2.8 + KAME
#
System       kame-freebsd

# System information
TargetName   FreeBSD-2.2.8 Release + KAME-199903-stable

# Name
HostName     target.tahi.org

# Type
#   host or router
Type         host
User         root
Password     v6eval

#linkname interface  The EXACT ether source address
#         name       of the Interface Under Test
Link0     de0        00:00:92:a7:6d:f5
#Link1    de1        00:00:92:a7:6d:f6
#Link2    de2        00:c0:f6:b0:aa:ef
#Link3    de3        00:00:92:a7:6d:f8
#Link4    fxp0       00:90:27:14:ce:e3
```

2.2.5 Setup items

The following table shows the configurable items in this file.

Table 3 Setup Items

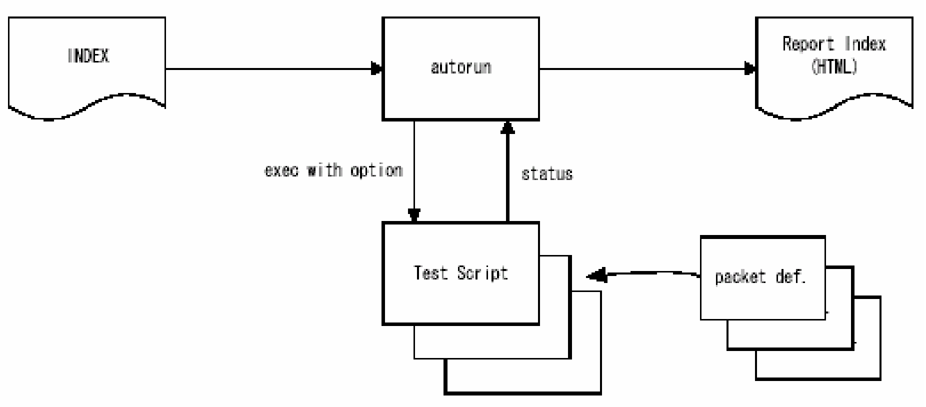
Entry	Omission	Description
System	Impossible	· The system name is specified. Valid entries are: kame-freebsd linux-v6 · The serial control method is changed according to this value.
TargeName	Impossible	· An identifier such as the version of the target is specified.
HostName	Impossible	· The host name of the target is specified. · The value must not contain spaces.
Type	Impossible	· "host" or "router" is specified.
User	Possible	· The user name at the time of remote control is specified. · If it is omitted, make it root.
Passward	Possible	· The password at the time of remote control is specified. · If it is omitted, make it V6eval.
LinkN	Impossible	· It is the same as that of the Tester Node Config file.

3. Command Syntax

3.1 Test Auto Execution Program (autorun)

3.1.1 Outline

- Conducts all the required tests and reports the results.
- The called test program looks inside the Target Config File and Tester Config File, and decides whether to conduct the test. This program does not take care of anything else.



3.1.2 Command line

autorun [-t] [-g] [-s num] [-e num] [-f] [index …]

The test specified in the INDEX file, as specified by index, is conducted. The test results are output to ./ /index.html. However, if ./index.html already exists, the test is not conducted.

3.1.3 Option

-t.

Conducts only the test of the packet definition file specified in INDEX is conducted. The test Script is not executed.

-g.

Creates an HTML document using the test script file specified in INDEX. The test script is not executed. ./index.html is output if the program passes all the tests.

* It is possible to specify options -t and -g simultaneously.

-s num.

Does not conduct tests with a number smaller than that of num.

-e num.

Does not conduct tests with a number larger than that of num.

* It is possible to specify the options -e and -s simultaneously.

-f.

If index.html exists at the time of test execution, the test is conducted, and index.html is overwritten with the new results.

3.1.4 Output

If the test script is executed, the HTML file containing the list of results (index.html) is output.

3.1.5 Returned values

0: No error during program execution

1: An error occurred during program execution.

3.1.6 INDEX File format

3.1.6.1 Format

- Information for one test is specified on each line.
- A line beginning with a hash symbol (#) can be used as a comment. (The system ignores lines beginning with a hash symbol and blank lines.)
- If you wish to display the test version in the results list, specify the following in the comment line:

  # $Name hogehoge$
- When a line starts with "&print:", the portion after the colon is output in the HTML test results document.
- The line starting with "&section:" is … (yet to be implemented)
- The following format applies to other lines:

  <seq>: Specifies the path name of the test sequence. Mandatory.

  <def>: Specifies the path name of the packet definition file. Mandatory.

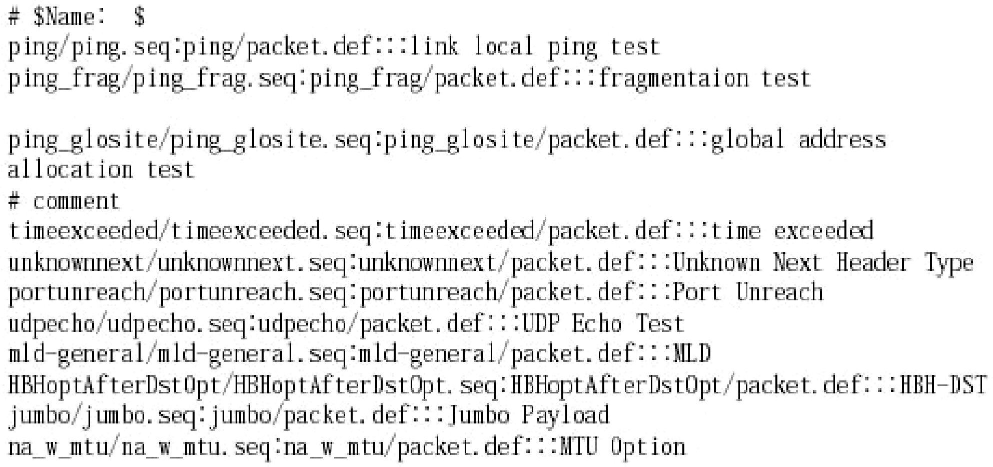
  <opts>: Specifies the argument to be passed to the test sequence. Mandatory. If the argument is omitted, the parameter is discarded.

  <html doc>: Specifies the file name of the HTML document to be used to create the data, instead of creating the data from perldoc attached to the conformance test sequence file.

  <dsc>: Specifies the test name. Optional. If omitted, the system takes ".seq" from the test sequence name.

  <links>: Specifies the number of network devices for the script. For example, if Link0 and Link1 are used, specify 2. If it is omitted, make it 1.

3.1.6.2 Example



```
# $Name:  $
ping/ping.seq:ping/packet.def:::link local ping test
ping_frag/ping_frag.seq:ping_frag/packet.def:::fragmentaion test

ping_glosite/ping_glosite.seq:ping_glosite/packet.def:::global address
allocation test
# comment
timeexceeded/timeexceeded.seq:timeexceeded/packet.def:::time exceeded
unknownnext/unknownnext.seq:unknownnext/packet.def:::Unknown Next Header Type
portunreach/portunreach.seq:portunreach/packet.def:::Port Unreach
udpecho/udpecho.seq:udpecho/packet.def:::UDP Echo Test
mld-general/mld-general.seq:mld-general/packet.def:::MLD
HBHoptAfterDstOpt/HBHoptAfterDstOpt.seq:HBHoptAfterDstOpt/packet.def:::HBH-DST
jumbo/jumbo.seq:jumbo/packet.def:::Jumbo Payload
na_w_mtu/na_w_mtu.seq:na_w_mtu/packet.def:::MTU Option
```

* See Chapter 5. Perl Library of Test Scripts for each of the test programs.

3.2 Packet Definition Syntax Checker (pktlint)

Checks to see if there is any error in the definition specified in the packet definition file. This check can be done by creating a packet to be sent using the packet defined in Frame XX.

3.2.1 Option

The same options as for the test sequence file (XXX)

3.2.2 Returned values

0: No error

Values other than 0: An error occurred.

3.3 Remote Control Program

3.3.1 Outline

A group of programs to provide remote control for devices to be verified, as well as test automation. The extension shall be rmt, and it will be called through Perl Library V6evalTool::vRemote().

3.3.2 Command

The following remote commands can be used:

Table 4 Remote Control Commands

Command Name	Operation
cleardefr.rmt	Clears the default router list.
clearnc.rmt	Clears the neighbor cache.
clearprefix.rmt	Clear the prefix list.
clearroute.rmt	Clears the routing table.
loginout.rmt	login, logout
manualaddrconf.rmt	Sets the address manually in I/F.
ping6.rmt	Pings the target.
reboot.rmt	After reboot, the program waits for the command prompt.
reboot_async.rmt	After reboot, the program terminates immediately.
route.rmt	Sets the routing.

3.3.3 Returned values

0: No error

Values other than 0: An error occurred.

4. Packet Definition

4.1 Outline

4.1.1 The method to describe definition

4.1.1.1 The method for specifying the packet definition

All definitions are specified in the following format:

```
Type name Tag name (
Element name = Value;
Element name = Value;
…
)
```

The element name to be specified internally is determined by the header name. In addition, the type of the value to be assigned is determined by the type name and element name.

- If the element name with an invalid order appears several times, priority is given to the last one.
- If the element name directly has a value, it starts with a capital letter.
- The element name referred to is specified in lower case (header, upper, payload ...)

4.1.1.2 Differences between the packets sent and those received

- The same means of specification are used for packets to be sent and those received.
- Only (=) can be defined in the packet to be sent.
- In addition to the definition of the packet to be sent, it is possible to specify a comparison of the size and the selection definition for a packet to be received.

4.2 Type

The configurable value can be classified into the following ten types:

- Reference type
- Option type
- Data type
- crypt type
- auth type
- ether type
- IPv6 Addr type
- IPv4 Addr type
- payload type
- uint type

4.2.1 Reference type

- A type to define the packet structure
- The tag defined in another place is specified.
- Backward reference is possible in the definition file.
- The operations specified in the following table are valid.

Table 5 Operation for the Reference type

Specified Symbols	Send Operation	Receive Operation	Remarks
=ref			

4.2.2 Option type

- A type to define the option in the header.
- Specifies the tag (option) defined in another place.
- Backward reference is possible in the definition file.
- The operations specified in the following table are valid.

Table 6 Operations for the Option type

Specified Symbols	Send Operation	Receive Operation	Remarks
=ref			
=oneof(ref,ref[,ref]…)	It is not possible to specify the data.	or	
=comb(ref,ref[,ref]…)	It is not possible to specify the data.	The data can be specified in any order is possible and everything is included.	
=stop	It is not possible to specify the data.	No comparison is made with the following options	

4.2.3 Data type

- A type for the element to be defined as a string of bytes.
- It is used when a string of bytes is specified independently of the format, such as payload data.
- Complete match in comparison (only Equal can be specified for sending and receiving data. = any cannot be specified.)
- One-byte data can be specified. If a value greater than 256 is specified, an error will occur.
- repeat () is used for defining repeat.
- The operations specified in the following table are valid.

Table 7 Operations for the Data type

Specified Symbols	Send Operation	Receive Operation	Remarks
=val			One-byte data. val must be in the range of 0-255.
={val[,val…]}			One-byte data x n val must be in the range of 0-255.
=file('file-path')			A string of bytes loaded from a file. The file name must be enclosed in single quotation marks ('). (Not yet implemented)
=repeat(val,times)			A string of bytes wherein val is repeated the specified number of times.
=substr(ref.offset,len)		*1	A data array taken out len byte from offset of ref, which is defined as a packet in another place.
=left(ref,len)		*1	A data array taken out len byte from the beginning of ref, which is defined as a packet in another place. It is equal to =substr (ref, (), len)
=right(ref,offset)		*1	A data array taken out from the offset-byte location to the end of ref, which is defined as a packet at another place. It is equal to =substr (ref, offset, sizeof(ref)-offset)
=patch(ref,offset,val)		*1	A data array wherein the value of the offset-byte location of ref, which is defined as a packet at another place, is replaced with val. val must be in the range of 0-255.

***1:** Even when a packet is received, the referred packet definition in ref is regarded as a packet to be sent, and is processed accordingly. This fact should be taken into consideration if the address and other values are embedded with auto.

4.2.4 crypt type

- A type for IPsec ESP.
- Specifies the encryption and decryption information.
- A comparison is not performed for the ICV value when a packet is received, and decryption is performed on the ICV included in the received packet.

Table 8 Operations for the crypt type

Specified Symbols	Send Operation	Receive Operation	Remarks
=null_crypt(alignment)			
=descbc(key[,ivec[,padlen]])			
=blowfish(key[,ivec[,padlen]])			
=rc5(key[,ivec[,padlen]])			
=cast128(key[,ivec[,padlen]])			
=des3cbc(key[,ivec[,padlen]])			

4.2.4.1 null_crypt encryption algorithm

- Encryption is not performed.
- Alignment length is specified.

4.2.4.2 descbc algorithm

- Encryption and decryption are performed in accordance with RFC2405.
- The key, initial vector and padding length are specified.

4.2.4.3 blowfish algorithm

- Encryption and decryption are performed in accordance with RFC2451.
- The key, initial vector and padding length are specified.

4.2.4.4 rc5 encryption algorithm

- Encryption and decryption are performed in accordance with RFC2451.
- The key, initial vector and padding length are specified.

4.2.4.5 cast128 encryption algorithm

- Encryption and decryption are performed in accordance with RFC2451.
- The key, initial vector and padding length are specified.

4.2.4.6 des3cbc encryption algorithm

- Encryption and decryption are specified in accordance with RFC2451.
- The key, initial vector and padding length are specified.

4.2.5 auth type

- A type for IPsec ESP & AH.
- Specifies the calculation algorithm information of the hash value.

Table 9 Operations for the auth type

Specified Symbols	Send Operation	Receive Operation	Remarks
=null_auth(alignment)			
=hmacmd5(key)			
=hmacsha1(key)			

4.2.5.1 null.auth

- Specifies the size of alignment.

4.2.5.2 hmacmd5 authentication algorithm

- Calculates the authentication value in accordance with RFC2403.
- Specifies the key.

4.2.5.3 hmacshal authentication algorithm

- Calculates the authentication value in accordance with RFC2404.
- Specifies the key.

4.2.6 esppad type

- A type specifying the rules for creating detailed data in the padded area of ESP.

Table 10 Operations for the esppad type

Specified Symbols	Send Operation	Receive Operation	Remarks
=sequential()		Comparison does not apply.	
=allzero()		Comparison does not apply.	
=random()		Comparison does not apply.	

* Even if the padded area is specified in the definition of the received packet, no comparison is made with the area.

4.2.6.1 sequential()

A value is padded in the padded data area sequentially from the beginning, such as "0, 1, 2, 3…"

4.2.6.2 allzero()

0 is padded in the overall padded data area.

4.2.6.3 random()

Random numbers are padded in the padded data area.

4.2.7 ether type

- A type to describe the Ethernet MAC address.

Table 11 Operations for the ether type

Specified Symbols	Send Operation	Receive Operation	Remarks
=ether(MAC Addr description)			
=v62ehtermulti(v6addr type)			
=tnether(ifname)			
=ethersrc(ifname)			
=etherdst(ifname)			
=nutether(ifname)			
=auto			Differs depending on where it is defined.
=any	Cannot be specified.		

4.2.8 IPv6 Addr type

- A type to describe the IPv6 address.

Table 12 Operations for the IPv6 Addr type

Specified Symbols	Send Operation	Receive Operation	Remarks
=v6(v6 addr description)			
=tnv6(ifname)			
=nutv6(ifname)			
=v6src(ifname)			
=v6dst(ifname)			
=v6merge(prefix,len,v6 addr)			
=v6ether(MAC);			
=auto			Differs depending on where it is defined.
=any	Cannot be specified.		

4.2.9 IPv4 Addr type

- A type to describe the IPv4 Addr address.

Table 13 Operations for the IPv4 Addr type

Specified Symbols	Send Operation	Receive Operation	Remarks
=v4(v4 addr description)			
=auto			Differs depending on where it is defined.
=any	Cannot be specified.		

4.2.10 payload type

- A type to refer to the payload portion of the packet
- Specifies the tag specified in another place.
- Backward reference is possible in the definition file.
- The operations specified in the following table are possible.

Table 14 Operations for payload

Specified Symbols	Send Operation	Receive Operation	Remarks
=ref			
=any	Cannot be specified.	No comparison is made.	

4.2.11 uint type

- It is used for other elements with values.

Table 15 Operations for the uint type

Specified Symbols	Send Operation	Receive Operation	Remarks
=val			
=within(val1,val2)	Cannot be specified.		val1<member<val2
=oneof(val,val[,val]...)	Cannot be specified.		
>val	Cannot be specified.		
<val	Cannot be specified.		
>=val	Cannot be specified.		
<=val	Cannot be specified.		
=any	Cannot be specified.		
=auto			

4.3 Address Change Functions

To take out the tester and target-dependent definitions from the packet definition file and test the sequence file, the following functions for creating a value using the test condition definition file can be used in the packet definition.

4.3.1 Ether address

4.3.1.1 ether()

Format

```
ether("MAC Address")
```

- Accepts the MAC address as an argument, and returns an ether-type value.
- The argument is mandatory.

Example

```
ether("00:11:22:33:44:55");
```

4.3.1.2 tnether()

Format

```
tnether(["ifname"])
```

- Accepts the interface name as an argument.
- Returns an ether-type value corresponding to the tester MAC address with the interface specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the -i option of pktsend and pktrecv.

Example

```
tnether("Link5");
tnether();
```

4.3.1.3 nutether()

Format

```
nutether(["ifname"])
```

- Accepts the interface name as an argument.
- Returns the ether-type value corresponding to the target MAC address with the interface specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the -i option of pktsend and pktrecv.

Example

```
nutether("Link5");
nutether();
```

4.3.1.4 ethersrc()

Format

ethersrc(["ifname"])

- Accepts the interface name as an argument.
- When a packet is sent, returns the ether-type value corresponding to the tester MAC address with the interface specified by the argument.
- When a packet is received, returns the ether-type value corresponding to the target MAC address with the interface specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the -i option of pktsend and pktrecv.

Example

```
ethersrc("Link5");
ethersrc();
```

4.3.1.5 etherdst()

Format

etherdst(["ifname"])

- Accepts the interface name as an argument.
- When a packet is sent, returns the ether-type value corresponding to the target MAC address with the interface specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the -i option of pktsend and pktrecv.

Example

```
etherdst("Link5");
etherdst();
```

4.3.1.6 v62ethermulti()

Format

v62ethermulti(IPv6type)

- Accepts the IPv6-type value, in other words, the function for returning the IPv6 type as an argument.
- Returns the ether-type value of the Ethernet Multicast address, which corresponds to the IPv6 Multicast address specified by the argument.
- The argument is mandatory.

Example

```
v62ethermulti(v6("ff02::1"));
```

4.3.2 IPv6 address

4.3.2.1 v6()

Format

v6("RFC2373 description")

- Accepts the IPv6 address specified in RFC2373 as an argument.
- Returns the IPv6 type corresponding to the argument.

Example

v6(“ff02::1”);

4.3.2.2 tnv6()

Format

tnv6([“ifname”])

- Accepts the interface name as an argument.
- Returns the IPv6 link local address of the tester, which is created from the interface name specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the command line option, the -i option.

Example

```
tnv6("Link5");
tnv6();
```

4.3.2.3 nutv6()

Format

nutv6([“ifname”])

- Accepts the interface name as an argument.
- Returns the IPv6 link local address of the target, which is created from the interface name specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the command line option, the -i option.
- If it is omitted, i-option

Example

```
nutv6("Link3");
nutnv6();
```

4.3.2.4 v6src()

Format

v6src([“ifname”])

- Accepts the interface name as an argument.
- When a packet is sent, returns the IPv6 link local address of the tester, which is created from the interface name specified by the argument.
- When a packet is received, returns the IPv6 link local address of the Target, which is created from the interface name specified by the argument.
- If the argument is omitted, the value defaults to the interface name specified by the command line option, the -i option.

Example

```
v6src("Link3");
v6src();
```

4.3.2.5 v6dst()

Format

v6dst([“ifname”])

- Accepts the interface name as an argument.
- When a packet is sent, returns the IPv6 link local address of the target, which is created from the interface name specified by the argument.
- When a packet is received, returns the IPv6 link local address of the tester, which is created from the interface name specified by the argument.
- If it is omitted, i-option

Example

```
v6dst("Link3");
v6dst();
```

4.3.2.6 v6merge()

Changes the prefix of the IPv6 address.

Format

v6merge("RFC2372", prefixlength, v6 address type)

Example

```
v6merge("ff02::",64,tnv6());
```

4.3.2.7 v6ether()

Created the IPv6 Link Local Address from the Ether address.

Format

v6ether(“MAC Address”)

Example

v6ether("00:01:02:03:04:05");

4.3.3 IPv6 address

4.3.3.1 v4()

Format

v4 ("IPv4 address description")

- Accepts the IPv4 address as an argument.
- Returns the IPv4 type.
- The argument cannot be omitted.

Example

v4(“127.0.0.1”);

4.4 Comment Description

Comments shall be specified in the C++ format.

Example

```
// comment1
/*
comment2
*/
```

4.5 Definition of the Packet Structure

4.5.1 Outline

The packet to be sent and received is created by defining each component, such as the header and payload, and by configuring it. The components are specified in the following type of definition:

- frame
- packet
- payload

4.5.2 Frame_Ether definition

Table 16 Frame_Ether Definition

Element	Type	
header	ref	Refers to the Hdr_Ehter definition.
packet	ref	Refers to the Packet_XX, or Payload definition.

- Each one appears only once.
- The order is irrelevant.
- The definition is mandatory.
- Only this definition can be specified as a packet to be sent and received.

4.5.3 Packet_XX definition

The packet definitions are as follows:

- Pacekt_IPv6
- Packet_IPv4
- Packet_ARP
- Packet_RARP

The following rules apply to these definitions.

Table 17 Packet_XX Definitions

Element	Type	
header	ref	
exthdr	ref	Only Packet_IPv6 and Packet_IPv4 can be omitted. Refers to the extension header.
upper	ref	Refers to the definition of packet, upper, and payload. Only Packet_IPv6 and Packet_IPv4 can be omitted.

- upper appears only once. (Currently, it is mandatory.)
- It is possible to specify exthdr multiple times, and the header is configured in the order specified.
- If the program refers to the packet definition for upper, a tunneled packet is created.
- upper definition means TCP, UDP, ICMP, and Packet_XX.
- It is possible to refer to the first header for Hdr_IPv4 and Hdr_IPv6 only.
- It is possible to refer to the following headers except for Hdr_IPv4 and Hdr_IPv6.

4.5.4 Payload definition

Table 18 Payload Definitions

Element	Type	
data	data	

- At least one value must be specified.
- It is possible to specify multiple values.
- Comparison is made for the data defined here as a string of bytes.

4.6 Definition of Other Types

See the list of header formats in Chapter 6. Element names and type names can be created by capitalizing the first character of the words listed therein.

4.7 Definition of Options and Comparisons

4.7.1 Option definition (when a packet is sent)

Option is specified in the header definition as follows:

```
option = tag name;
```

4.7.2 Definition of option comparison

4.7.2.1 OR

If any of the specified options occur, option will be specified as follows:

```
option = oneof(A, B, C);
```

4.7.2.2 Any order

If any order is permitted, but the specified option appears, the option will be specified as follows:

```
option = comb(A, B, C);
```

This means "A, B, C are included in any order for the following three options."

A, B, C shall be tag names. Definitions cannot be nested.

In addition, option can be specified as follows:

```
option = comb(A, A, C);
```

In this case, A appears twice, and C appears once.

4.7.2.3 The indefinite option

If a comparison does not have to be made for the option or the portion following the specified option, the option is specified as follows:

```
option = stop();
```

4.8 Alignment

Each option in the option header must be aligned on the natural boundary according to the data. The option Padl/PadN is available for this purpose. With this tool, padding is not done automatically.

However, if the endpoint of an option header does not match the 8-byte boundary, an alert will be issued to the user, who then must pad the header based on this information.

When a packet is received, it is processed by skipping Padl/PadN. These options, if they exist in the definition of the received packet, are be ignored.

4.9 IPsec

4.9.1 ESP

4.9.1.1 Corresponding encryption/decryption functions and hash functions

The corresponding encryption/decryption functions shall be as follows:

- DES in CBC mode
- NULL Encryption

The corresponding hash function shall be the same as that of AH.

4.9.2 AH

4.9.2.1 Corresponding hash functions

The corresponding hash functions shall be as follows:

- HMAC-MD5-96
- HMAC-SHA-1-96

4.9.2.2 IPsec and fragment

A valid packet, such as:

- IP Fragment ESP …
- IP Fragment AH …

is created using subster(), or suchlike as in the case of a normal fragment packet.

For an invalid packet configuration

- IP AH Fragment

the portion following the fragment header is calculated as a fixed area.

The configuration

- IP AH AH xxxx

will not cause an error and the invalid value will be input.

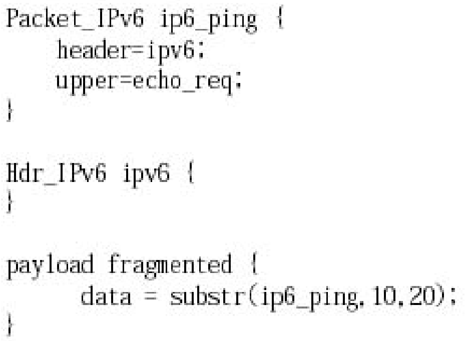
4.10 FAQ

4.10.1 Definition of the fragmented packet

It is impossible to specify a packet and issue the instruction "Create a packet fragmented by MTU = nnn."

Part of the packet defined separately is taken out, and it is processed like payload.

//Define a large packet before it is fragmented.

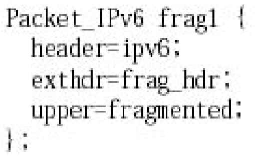


```
Packet_IPv6 ip6_ping {
    header=ipv6;
    upper=echo_req;
}

Hdr_IPv6 ipv6 {
}

payload fragmented {
      data = substr(ip6_ping,10,20);
}
```

//Packet to be sent



```
Packet_IPv6 frag1 {
  header=ipv6;
  exthdr=frag_hdr;
  upper=fragmented;
};
```

* The offset of the fragment header shall be padded by the person who writes the definition.

5. Perl Library for Test Scripts

This chapter describes the perl API for creating a packet, receiving a packet and packet memory control.

5.1 Script File Names

When running the specification conformance test automatically, the following rules apply to the file names for scripts.

- The extension is ".seq."
- The directories are used to classify the files according to RFC and draft. Therefore, these values must not be specified in the file names.

5.2 Initialization

People using this library must specify the following sentence at the beginning of the test sequence script.

```
BEGIN { $V6evalTool::TestVersion = ’$Name: $’; }
use V6evalTool;
```

If an error occurs during initialization, internal error (64) will be returned as the end code.

5.3 Command Line Option

If this library is used, the following command line options can be specified.

-tn: esting node config file

-nut: Node under test config file

-pkt: Packet definition file

-log: File name of the log to be output

-v: Details output in the log are also displayed on the screen.

-l: Log Level (0,1,2)

-h: Help

-nostd: Specifies that the standard Include File is not loaded.

-remote: Specifies the remote option.

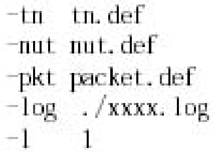
-noremote: Specifies that remote control is not performed (*).

-keepImd: Specifies that the intermediate file is not deleted after being processed with CPP.

-trace: Detailed trace information is output in the standard output.

* The status vRemote() is returned to indicate that processing completed normally. (Is this specification okay?)

If the options are omitted, the program will operate according to the following default values.



```
-tn  tn.def
-nut nut.def
-pkt packet.def
-log  ./xxxx.log
-l    1
```

If a character string not starting with a hyphen (-) is specified in the command line, users can refer to @ARGV for the string. If the option starts with a hyphen and an option other than those above are specified, the program terminates in error. The end code is Internal Error.

5.4 Status Code

The test script shall return the following status codes to indicate the test results.

Table 19 Status Codes

Code	Constant	Description
0	$exitPass	PASS
1	$exitIgnore	Normal termination with the script, such as test preparation
2	$exitNS	Not yet supported
3	$exitWarn	Cannot be identified as a failure because the specification is unclear.
4	$exitHostOnly	The host-dedicated test was conducted for the router.
5	$exitRouterOnly	The router-dedicated test was conducted for the host.
32	$exitFail	FAIL
64	$exitFatal	Internal Error

If the auto execution program (see 3.1) receives $exitFatal, the following procedures are stopped, and a report is output for the tests executed up to that point.

5.5 Library Functions

The following functions are available for users.

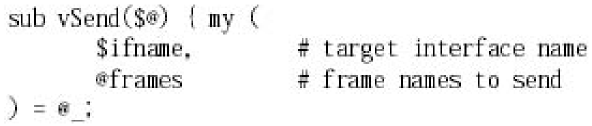
vStop();
vClear();
vCapture();
vSend();
vRecv();
vLog();
vRemote();
vErrmsg();
vRoundoff();
vSleep();

5.5.1 vSend

5.5.1.1 Outline

The specified packet is sent from the specified interface. (Multiple packets can be specified.)

5.5.1.2 Argument



```
sub vSend($@) { my (
        $ifname,        # target interface name
        @frames         # frame names to send
) = @_;
```

5.5.1.3 Returned value

The hash value is returned. The key and value are as follows:

sentTimeN

Time when the packet specified for the N number was sent. For example, if the user wishes to determine the time the packet specified at first was sent, the format is $retval{'sentTime1'}. The format is stored in the character string as follows.

%ld[=epochtime].%06d[=microsecond]

However, if an error occurs during this process, the test script terminates, and $exitFatal (64) is returned as the end code.

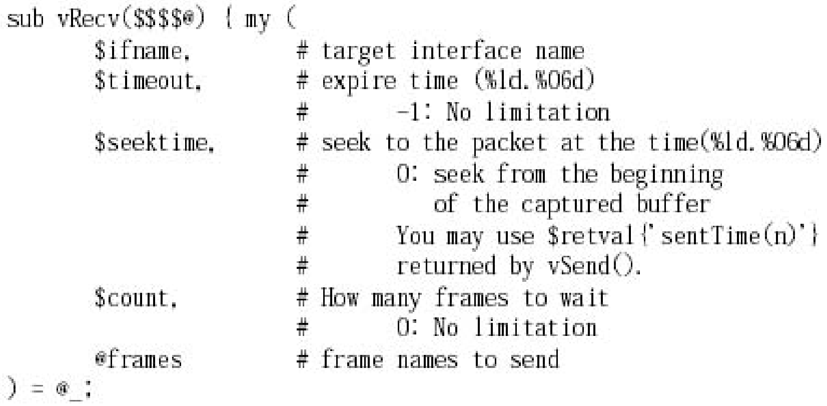
See 5.6 for furtehr details.

5.5.2 vRecv

5.5.2.1 Outline

A test is conducted to determine whether the expected packet can be received from the specified interface.

5.5.2.2 Argument



```
sub vRecv($$$$@) { my (
        $ifname,        # target interface name
        $timeout,       # expire time (%ld.%06d)
                        #       -1: No limitation
        $seektime,      # seek to the packet at the time(%ld.%06d)
                        #       0: seek from the beginning
                        #          of the captured buffer
                        #       You may use $retval{'sentTime(n)'}
                        #       returned by vSend().
        $count,         # How many frames to wait
                        #       0: No limitation
        @frames         # frame names to send
) = @_;
```

5.5.2.3 End status

The following three cases can be considered when this routine terminates:

(1) The specified packet was received.

(2) Time specified in Timeout elapsed.

(3) The number of packets specified in Count was received.

If one of these conditions is met, the routine terminates.

5.5.2.4 Returned values

The hash value is returned.
The key and value are as follows:

status:

=0: Normal end
1: End due to timeout
2: The specified packet was received, but the expected packet could not be received.
>=3: Error

recvCount;

The number of packets received before the expected packet was received (including the expected packet)

recvTimeN:

Time when the packet specified by the N number was received. For example, if the user wishes to know when the first packet was received, the format shall be $retval{'recvTime1'}
The format is stored in the character string as follows:

%ld[=epochtime].%06d[=microsecond]

recvFrame:

If this routine terminates normally, the packet name will be entered as information to indicate which of the packets specified as expected values were received. If an errors occur (including timeout), the hash value is not defined.

See 5.6 for further details.

5.5.3 vCapture

5.5.3.1 Outline

This function specifies storing the received packets in the packet memory with the specified interface.

5.5.3.2 Argument

sub vCapture($)

Accepts the link name as an argument.

5.5.3.3 Returned values

Returns the scalar value, 0.

Normal termination is assumed, even if the Capture mode is already launched.

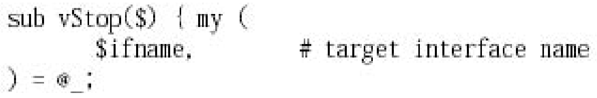
However, if an error occurs during processing, the test script terminates and $exitFatal(64) is returned as the end status.

5.5.4 vStop

5.5.4.1 Outline

Stops the Capture mode in the packet memory with the specified interface.

5.5.4.2 Argument



```
sub vStop($) { my (
        $ifname,        # target interface name
) = @_;
```

5.5.4.3 Returned values

Returns the scalar value, 0.

Normal termination is assumed, even if the Capture mode is already stopped.

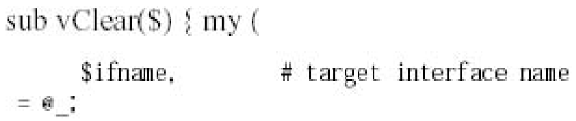
However, if an error occurs during processing, the test script terminates and $exitFatal(64) is returned as the end status.

5.5.5 vClear

5.5.5.1 Outline

Clears the buffer of the packet memory with the specified interface.

5.5.5.2 Argument



```
sub vClear($) { my (
      $ifname,        # target interface name
 = @_;
```

5.5.5.3 Returned values

Returns the scalar value, 0.

Normal termination is assumed, even if the buffer is already blank. (The same specifications for the returned values of pktctl.)

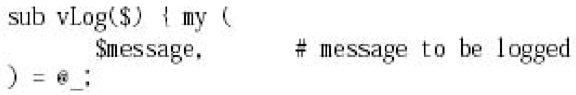
However, if an error occurs during processing, the test script terminates and $exitFatal(64) is returned as the end status.

5.5.6 vLog

5.5.6.1 Outline

Writes the specified character string in the log.

5.5.6.2 Argument



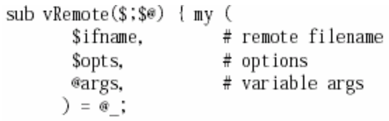
```
sub vLog($) { my (
      $message,       # message to be logged
) = @_;
```

5.5.7 vRemote

5.5.7.1 Outline

Calls the remote control program.

5.5.7.2 Argument



```
sub vRemote($;$@) { my (
       $ifname,        # remote filename
       $opts,          # options
       @args,          # variable args
      ) = @_;
```

5.5.7.3 Returned values

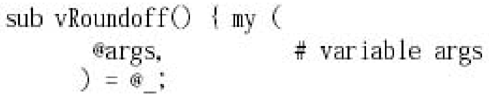
0: No error
Values other than 0: An error occurred

5.5.8 vRoundoff

5.5.8.1 Outline

Rounds off the value to the tenth decimal place.

5.5.8.2 Argument



```
sub vRoundoff() { my (
       @args,          # variable args
      ) = @_;
```

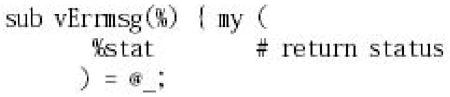
5.5.8.3 Returned values

Rounded value

5.5.9 vErrmsg

Returns the corresponding error message when the returned value of vRecv is given.

5.5.9.1 Argument



```
sub vErrmsg(%) { my (
       %stat        # return status
      ) = @_;
```

5.5.9.2 Returned values

Table 20

Values for $stat{status}	Message
0	NULL
1	1 “Time-out :”+ *1
2	2 “Count-out”+ *1
Values larger than 3	“ERROR: Internal error”

However, *1 means "Could not get any packet" if $stat{status} is 0, and "Received unexpected packet" in other cases.

5.5.10 vMAC2Addr

This function creates the link local address from the MAC address.

5.5.10.1 Argument

sub vSleep($)

5.5.10.2 Returned values

Link local address

5.5.11 vSleep()

Used instead of "sleep," and the common log is output.

5.6 Reference of Each Field of the Packet Sent and Received

5.6.1 Outline

The values returned by vRecv() and vSend() have variables for referring to the values of each field of the packet sent and received. It is possible to read the values by assigning the key specifying the following fields to the hash value obtained as return values.

Example of referring to the IPv6 header/source address in the packet received

```
$status = vRecv(......);
$source_address = $status{"Frame_Ether.Packet_IPv6.SourceAddress"}
```

5.6.2 Filed and Key

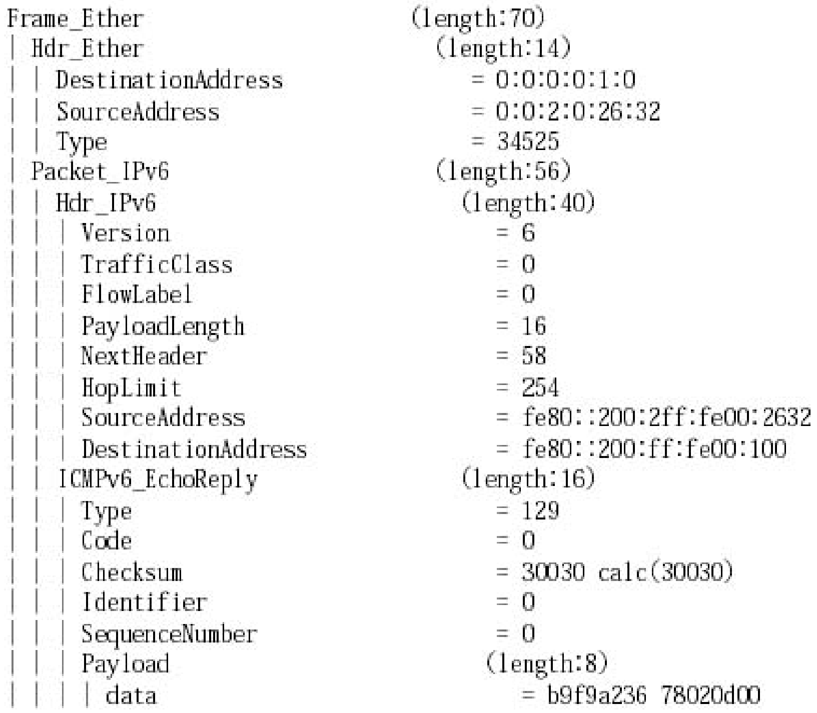
The following rules apply when combining the field and the hash key.

5.6.2.1 Basic rules

- It shall be possible to refer to the field values of the packet sent and to the packet corresponding to the expected values.
- A key shall be what connects the block name indicating the structure with a period "."

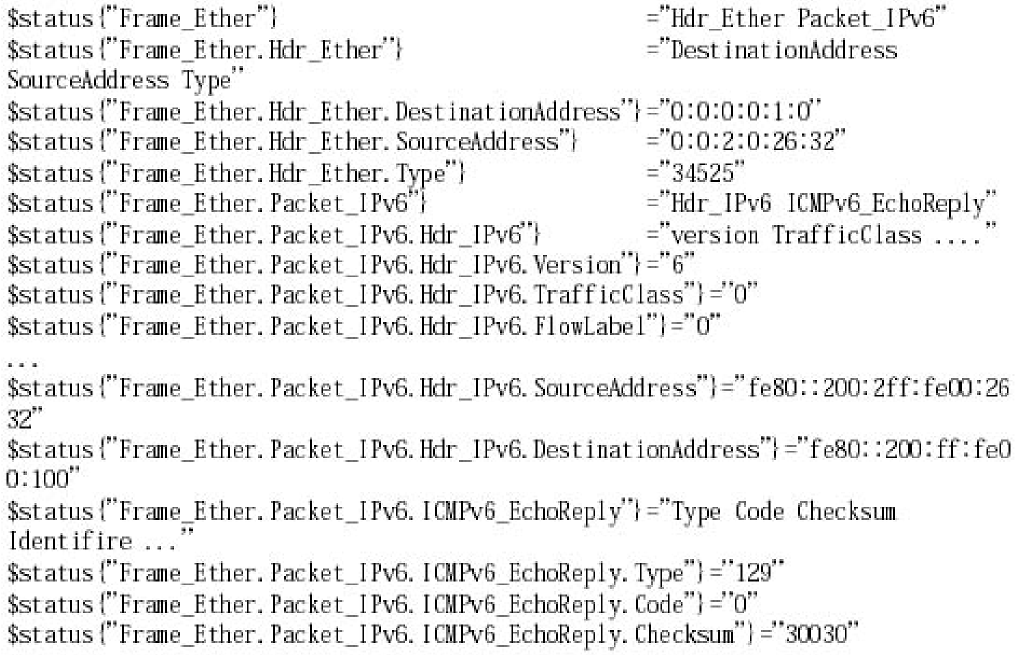
Example

Structure of the packet received



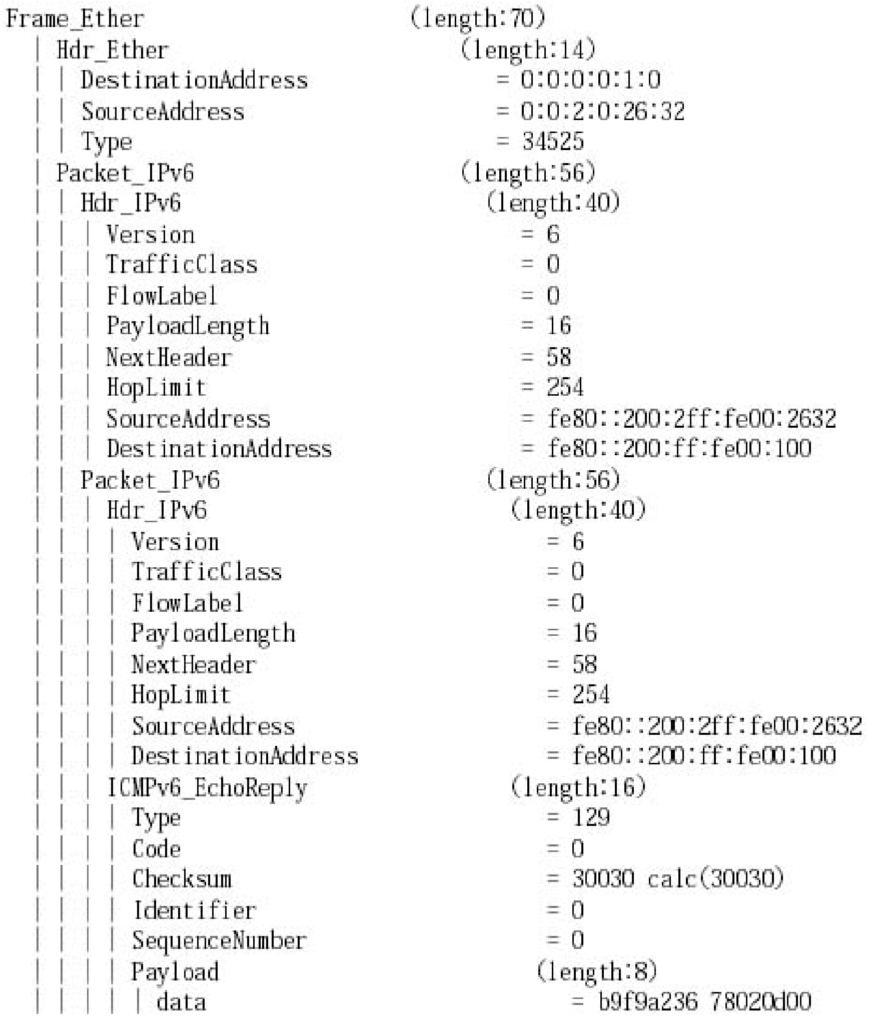
```
Frame_Ether                     (length:70)
| Hdr_Ether                       (length:14)
| | DestinationAddress                = 0:0:0:0:1:0
| | SourceAddress                     = 0:0:2:0:26:32
| | Type                              = 34525
| Packet_IPv6                     (length:56)
| | Hdr_IPv6                        (length:40)
| | | Version                           = 6
| | | TrafficClass                      = 0
| | | FlowLabel                         = 0
| | | PayloadLength                     = 16
| | | NextHeader                        = 58
| | | HopLimit                          = 254
| | | SourceAddress                     = fe80::200:2ff:fe00:2632
| | | DestinationAddress                = fe80::200:ff:fe00:100
| | ICMPv6_EchoReply                (length:16)
| | | Type                              = 129
| | | Code                              = 0
| | | Checksum                          = 30030 calc(30030)
| | | Identifier                        = 0
| | | SequenceNumber                    = 0
| | | Payload                         (length:8)
| | | | data                              = b9f9a236 78020d00
```

Substituted variables and the values



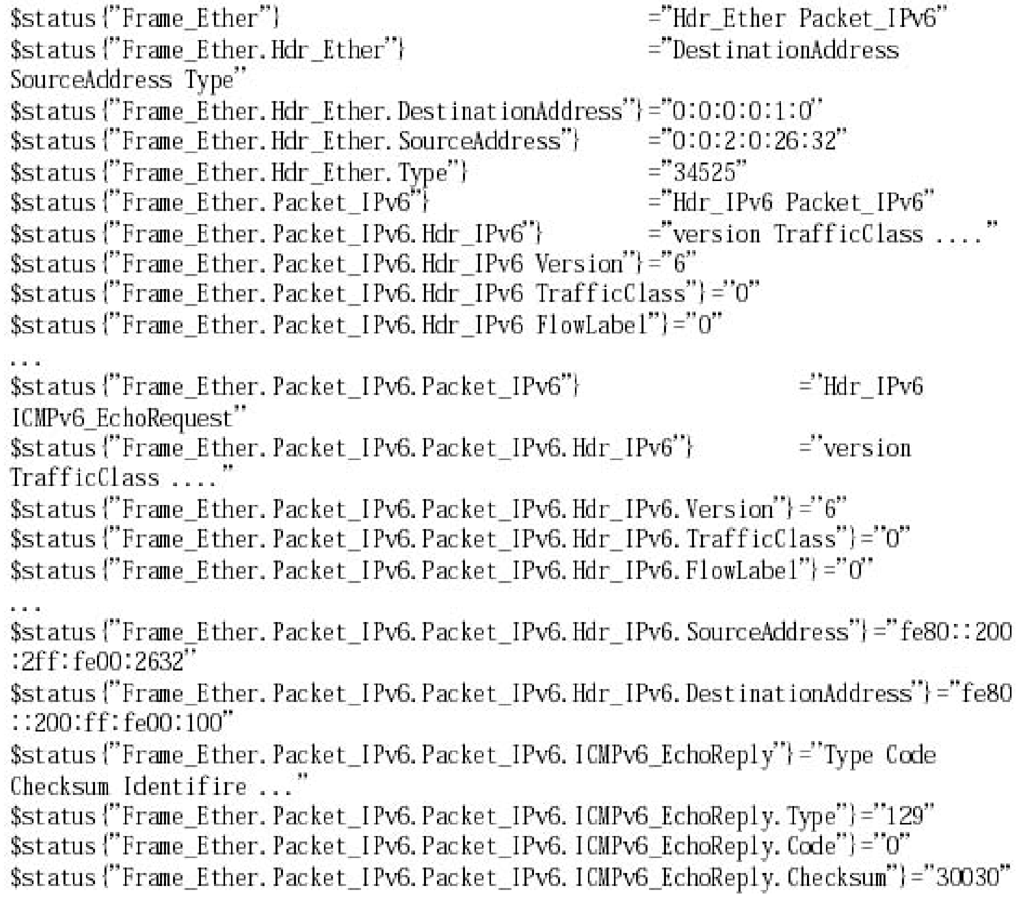
```
$status{"Frame_Ether"}                                   ="Hdr_Ether Packet_IPv6"
$status{"Frame_Ether.Hdr_Ether"}                         ="DestinationAddress
SourceAddress Type"
$status{"Frame_Ether.Hdr_Ether.DestinationAddress"}="0:0:0:0:1:0"
$status{"Frame_Ether.Hdr_Ether.SourceAddress"}           ="0:0:2:0:26:32"
$status{"Frame_Ether.Hdr_Ether.Type"}                    ="34525"
$status{"Frame_Ether.Packet_IPv6"}                       ="Hdr_IPv6 ICMPv6_EchoReply"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6"}              ="version TrafficClass ...."
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6.Version"}="6"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6.TrafficClass"}="0"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6.FlowLabel"}="0"
...
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6.SourceAddress"}="fe80::200:2ff:fe00:26
32"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6.DestinationAddress"}="fe80::200:ff:fe0
0:100"
$status{"Frame_Ether.Packet_IPv6.ICMPv6_EchoReply"}="Type Code Checksum
Identifire ..."
$status{"Frame_Ether.Packet_IPv6.ICMPv6_EchoReply.Type"}="129"
$status{"Frame_Ether.Packet_IPv6.ICMPv6_EchoReply.Code"}="0"
$status{"Frame_Ether.Packet_IPv6.ICMPv6_EchoReply.Checksum"}="30030"
```

Example 2



```
Frame_Ether                         (length:70)
  | Hdr_Ether                         (length:14)
  | | DestinationAddress                 = 0:0:0:0:1:0
  | | SourceAddress                      = 0:0:2:0:26:32
  | | Type                               = 34525
  | Packet_IPv6                       (length:56)
  | | Hdr_IPv6                          (length:40)
  | | | Version                            = 6
  | | | TrafficClass                       = 0
  | | | FlowLabel                          = 0
  | | | PayloadLength                      = 16
  | | | NextHeader                         = 58
  | | | HopLimit                           = 254
  | | | SourceAddress                      = fe80::200:2ff:fe00:2632
  | | | DestinationAddress                 = fe80::200:ff:fe00:100
  | | Packet_IPv6                       (length:56)
  | | | Hdr_IPv6                          (length:40)
  | | | | Version                            = 6
  | | | | TrafficClass                       = 0
  | | | | FlowLabel                          = 0
  | | | | PayloadLength                      = 16
  | | | | NextHeader                         = 58
  | | | | HopLimit                           = 254
  | | | | SourceAddress                      = fe80::200:2ff:fe00:2632
  | | | | DestinationAddress                 = fe80::200:ff:fe00:100
  | | | ICMPv6_EchoReply                  (length:16)
  | | | | Type                               = 129
  | | | | Code                               = 0
  | | | | Checksum                           = 30030 calc(30030)
  | | | | Identifier                         = 0
  | | | | SequenceNumber                     = 0
  | | | | Payload                           (length:8)
  | | | | | data                               = b9f9a236 78020d00
```

Substituted variables and the values



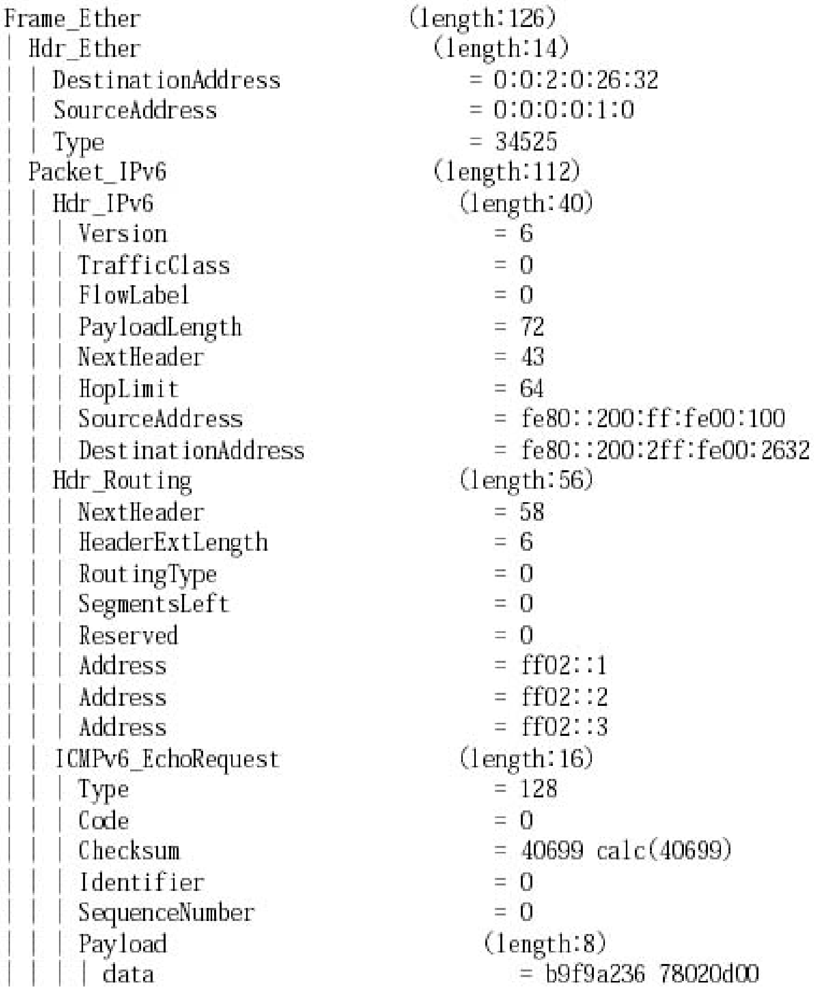
```
$status{"Frame_Ether"}                                   ="Hdr_Ether Packet_IPv6"
$status{"Frame_Ether.Hdr_Ether"}                         ="DestinationAddress
SourceAddress Type"
$status{"Frame_Ether.Hdr_Ether.DestinationAddress"}="0:0:0:0:1:0"
$status{"Frame_Ether.Hdr_Ether.SourceAddress"}      ="0:0:2:0:26:32"
$status{"Frame_Ether.Hdr_Ether.Type"}                    ="34525"
$status{"Frame_Ether.Packet_IPv6"}                       ="Hdr_IPv6 Packet_IPv6"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6"}          ="version TrafficClass ...."
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6 Version"}="6"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6 TrafficClass"}="0"
$status{"Frame_Ether.Packet_IPv6.Hdr_IPv6 FlowLabel"}="0"
...
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6"}                  ="Hdr_IPv6
ICMPv6_EchoRequest"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.Hdr_IPv6"}         ="version
TrafficClass ...."
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.Hdr_IPv6.Version"}="6"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.Hdr_IPv6.TrafficClass"}="0"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.Hdr_IPv6.FlowLabel"}="0"
...
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.Hdr_IPv6.SourceAddress"}="fe80::200
:2ff:fe00:2632"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.Hdr_IPv6.DestinationAddress"}="fe80
::200:ff:fe00:100"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.ICMPv6_EchoReply"}="Type Code
Checksum Identifire ..."
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.ICMPv6_EchoReply.Type"}="129"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.ICMPv6_EchoReply.Code"}="0"
$status{"Frame_Ether.Packet_IPv6.Packet_IPv6.ICMPv6_EchoReply.Checksum"}="30030"
```

There is no problem since the key is changed.

5.6.2.2 If a field with the same name exists in the same header

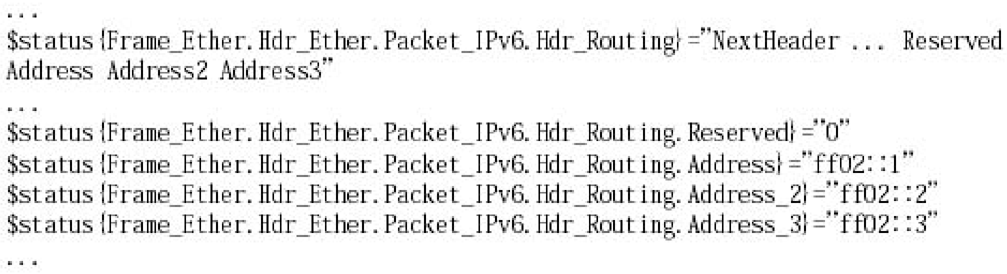
- Assumptions are made regarding the routing header.
- The header is suffixed with numerals.

Example



```
Frame_Ether                     (length:126)
| Hdr_Ether                       (length:14)
| | DestinationAddress               = 0:0:2:0:26:32
| | SourceAddress                    = 0:0:0:0:1:0
| | Type                             = 34525
| Packet_IPv6                     (length:112)
| | Hdr_IPv6                        (length:40)
| | | Version                          = 6
| | | TrafficClass                     = 0
| | | FlowLabel                        = 0
| | | PayloadLength                    = 72
| | | NextHeader                       = 43
| | | HopLimit                         = 64
| | | SourceAddress                    = fe80::200:ff:fe00:100
| | | DestinationAddress               = fe80::200:2ff:fe00:2632
| | Hdr_Routing                     (length:56)
| | | NextHeader                       = 58
| | | HeaderExtLength                  = 6
| | | RoutingType                      = 0
| | | SegmentsLeft                     = 0
| | | Reserved                         = 0
| | | Address                          = ff02::1
| | | Address                          = ff02::2
| | | Address                          = ff02::3
| | ICMPv6_EchoRequest              (length:16)
| | | Type                             = 128
| | | Code                             = 0
| | | Checksum                         = 40699 calc(40699)
| | | Identifier                       = 0
| | | SequenceNumber                   = 0
| | | Payload                         (length:8)
| | | | data                             = b9f9a236 78020d00
```

Substituted Variables and the Values

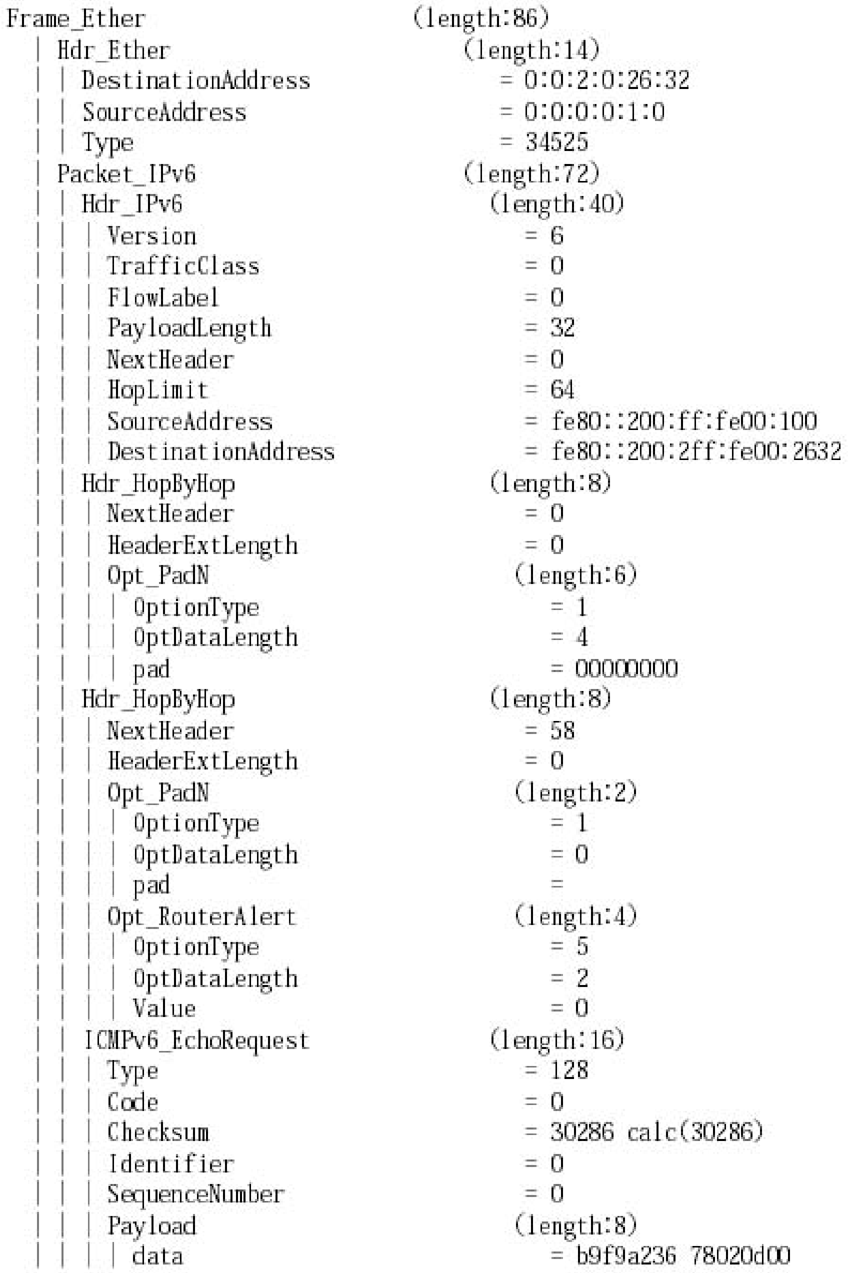


```
...
$status{Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_Routing}="NextHeader ... Reserved
Address Address2 Address3"
...
$status{Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_Routing.Reserved}="0"
$status{Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_Routing.Address}="ff02::1"
$status{Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_Routing.Address_2}="ff02::2"
$status{Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_Routing.Address_3}="ff02::3"
...
```

5.6.2.3 If the header or suchlike having the same name exists at the same level of nesting

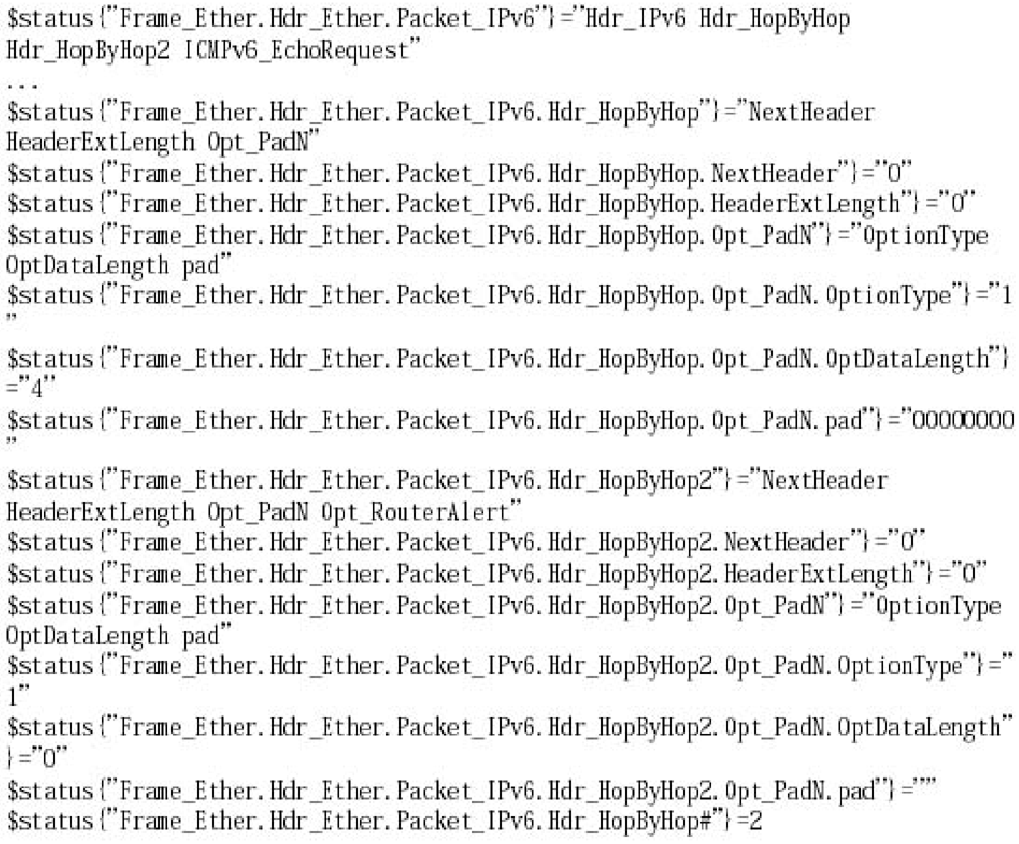
- Assumptions are made when the same option header is attached
- The structure name is appended with numerals.

Example



```
Frame_Ether                       (length:86)
  | Hdr_Ether                         (length:14)
  | | DestinationAddress                  = 0:0:2:0:26:32
  | | SourceAddress                       = 0:0:0:0:1:0
  | | Type                                = 34525
  | Packet_IPv6                       (length:72)
  | | Hdr_IPv6                            (length:40)
  | | | Version                             = 6
  | | | TrafficClass                        = 0
  | | | FlowLabel                           = 0
  | | | PayloadLength                       = 32
  | | | NextHeader                          = 0
  | | | HopLimit                            = 64
  | | | SourceAddress                       = fe80::200:ff:fe00:100
  | | | DestinationAddress                  = fe80::200:2ff:fe00:2632
  | | Hdr_HopByHop                        (length:8)
  | | | NextHeader                          = 0
  | | | HeaderExtLength                     = 0
  | | | Opt_PadN                            (length:6)
  | | | | OptionType                          = 1
  | | | | OptDataLength                       = 4
  | | | | pad                                 = 00000000
  | | Hdr_HopByHop                        (length:8)
  | | | NextHeader                          = 58
  | | | HeaderExtLength                     = 0
  | | | Opt_PadN                            (length:2)
  | | | | OptionType                          = 1
  | | | | OptDataLength                       = 0
  | | | | pad                                 =
  | | | Opt_RouterAlert                     (length:4)
  | | | | OptionType                          = 5
  | | | | OptDataLength                       = 2
  | | | | Value                               = 0
  | | ICMPv6_EchoRequest                  (length:16)
  | | | Type                                = 128
  | | | Code                                = 0
  | | | Checksum                            = 30286 calc(30286)
  | | | Identifier                          = 0
  | | | SequenceNumber                      = 0
  | | | Payload                             (length:8)
  | | | | data                                = b9f9a236 78020d00
```

Substituted Variables and the Values



```
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6”}=”Hdr_IPv6 Hdr_HopByHop
Hdr_HopByHop2 ICMPv6_EchoRequest”
...
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop”}=”NextHeader
HeaderExtLength Opt_PadN”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop.NextHeader”}=”0”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop.HeaderExtLength”}=”0”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop.Opt_PadN”}=”OptionType
OptDataLength pad”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop.Opt_PadN.OptionType”}=”1
”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop.Opt_PadN.OptDataLength”}
=”4”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop.Opt_PadN.pad”}=”00000000
”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2”}=”NextHeader
HeaderExtLength Opt_PadN Opt_RouterAlert”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2.NextHeader”}=”0”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2.HeaderExtLength”}=”0”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2.Opt_PadN”}=”OptionType
OptDataLength pad”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2.Opt_PadN.OptionType”}=”
1”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2.Opt_PadN.OptDataLength”
}=”0”
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop2.Opt_PadN.pad”}=””
$status{”Frame_Ether.Hdr_Ether.Packet_IPv6.Hdr_HopByHop#”}=2
```

5.7 Time Relationship

This section describes the interrelation between parameters concerning time, such as timeout, seektime, and count, when a packet is received.

5.7.1 If only timeout is specified

When pktrecv is forked by vRecv() and the argument is evaluated, the time shall be t1. If the time specified by the -e option is added to t1, a timeout shall occur. T1 is calculated before the packet definition is evaluated.

If a packet is received in the buffer by time t2, the packet is returned. If it is not received, the program will end in a timeout. The time is measured to the nearest second.

5.7.2 If seektime is specified

All the data received before time t0 specified by seektime will be read and discarded. After that, the normal recv operation will begin.

5.7.3 If seektime and timeout are specified

Timeout t2 shall be the time calculated by adding timeout x to seektime t0.

5.7.4 If seektime and count are specified

The program ends when the end conditions for earlier program are satisfied. However, the packets received before the seektime are read and discarded, and they are not counted.

5.7.5 If timeout and count are specified

The program ends when the end conditions for program are met.

6. Header Format

6.1 Description Method

6.1.1 Description rules

Table 21 Description Rules

Fields	Symbols	Description
Type	*	Parameters which do not appear in the packet, but are required when a packet is created.
Header name	(option)	A packet cannot be configured with this definition only.
Option	(opt)	Options that meet the standard and can be added
Option	(bad-opt)	Options that can be added but that invalidate the packet
default	auto	Calculation is done automatically, and the values are padded.
default (Receive)	check	If it is specified, it will be auto. It checks whether the values are correct.
default	no symbol	Mandatory

6.1.2 Basic policy for setup values when omitted

- The flag members are 0.
- The reserve area members are 0.
- NextHeader is auto.
- The length field is auto.
- The CheckSum field is auto.

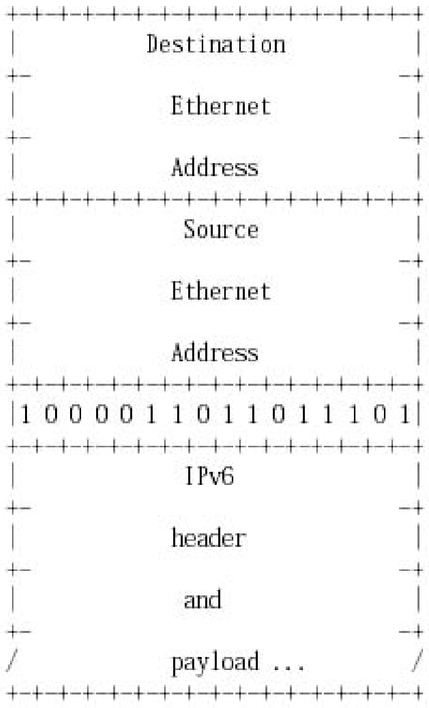
6.2 Data Link

6.2.1 Ethernet Header

6.2.1.1 Reference RFC

RFC2464,Transmission of IPv6 Packets over Ethernet Networks,December 1998

6.2.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Destination          |
+-                             -+
|            Ethernet           |
+-                             -+
|            Address            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Source            |
+-                             -+
|            Ethernet           |
+-                             -+
|            Address            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1 0 0 0 0 1 1 0 1 1 0 1 1 1 0 1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             IPv6              |
+-                             -+
|            header             |
+-                             -+
|             and               |
+-                             -+
/           payload ...         /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.2.1.3 Default

Table 22 Ethernet Header

Name	Type	Size	default (Send)	default (Receive)
Destination Address	Ether Address	48bit	Target Address	Tester Address
Source Address	Ether Address	48bit	Tester Address	Target Address
Type	uint	16bit	auto	any

6.2.1.4 Auto setup when a packet is sent

Auto creation rules for type

The following values are substituted according to the payload following this header.

Table 23 Auto Creation Rules for Type, Ethernet Header

The Subsequent Header Type	Values of Type	Remarks
IPv4 Header	0x0800	
ARP	0x0806	
RARP	0x8035	
IPv6 Header	0x86dd	[V6ETHER]

6.2.1.5 Other

Restriction in the location of definition

Specification is possible only at the beginning of the frame definition.

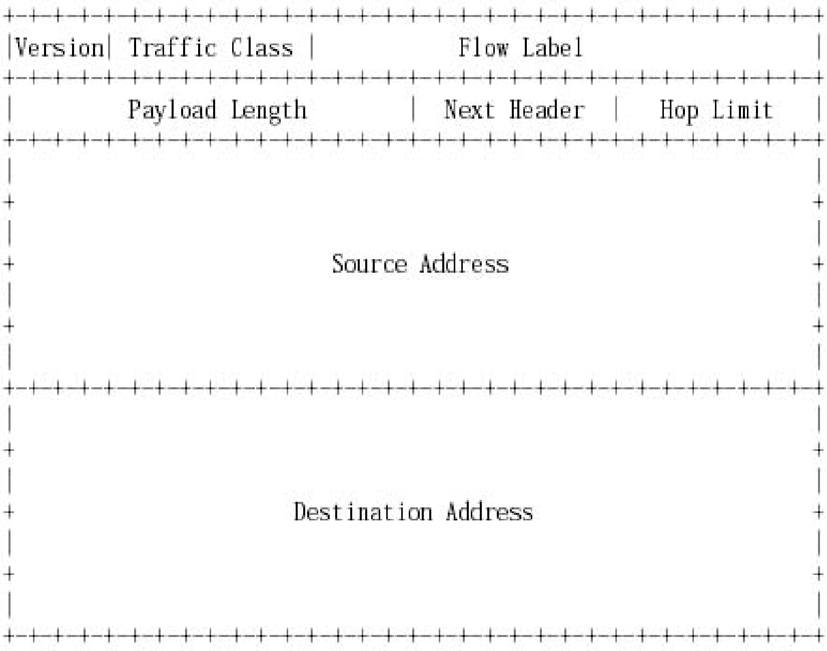
6.3 IPv6

6.3.1 IPv6 Header

6.3.1.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.3.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Traffic Class |           Flow Label                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Payload Length        |  Next Header  |   Hop Limit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                         Source Address                        +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                      Destination Address                      +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.3.1.3 Alignment

8N

6.3.1.4 Default

Table 24 IPv6 Header

Name	Type	Size	default (Send)	default (Receive)
Version	uint	4bit	6	6
Traffic Class	uint	8bit	0	any
Flow Lavel	uint	20bit	0	any
Payload Length	uint	16bit	auto	any
Next Header	uint	8bit	auto	any
Hop Limit	uint	8bit	64	any
Source Address	IPv6 Address	128bit	Tester IP Address	Target IP Address
Destination Address	IPv6 Address	128bit	Target IP Address	Tester IP Address

6.3.1.5 Auto setup when a packet is sent

Auto creation rules for payload length

- The length of the payload following the IPv6 header
- The units are octets.

Auto creation rules for the next header

The following values are input according to the subsequent header and the upper level. (Refer to IANA protocol numbers: rfc/iana/assignments/protocol-numbers.)

Table 25 Auto Creation Rules for Next Header

The Subsequent Header	Next Header
Hop-by-Hop Options	0
IPv6	41
Routing	43
Fragment	44
Authentication	51
Destination Option	60
Encapsulating Security Optio	50
ICMPv6	58
No Next Header	59
TCP	6
UDP	17
ICMP	1

6.3.1.6 Other

Can only be specified at the beginning of a packet definition.

6.4 IPv6 Extension Header

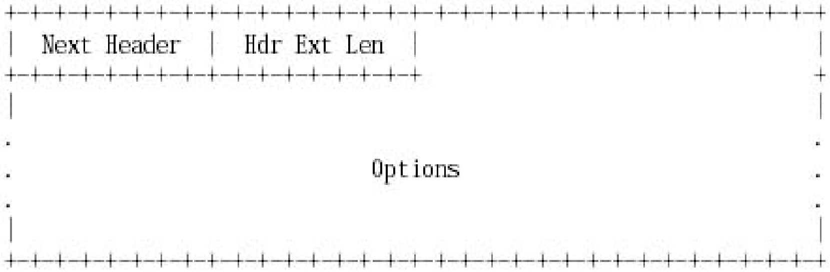
* Alignment is 8N for all. (Options are defined separately.)

6.4.1 Hop-by-Hop

6.4.1.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.4.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                                                               |
.                                                               .
.                            Options                            .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.4.1.3 Default

Table 26 Hop-by-Hop Header

Name	Type	Size	default (Send)	default (Receive)
Next Header	uint	8bit	auto	any
Header Ext Length	uint	8bit	auto	any
(Router Alert)				
(Jumbo payload)				
(Pad1)				
(Padn)				

6.4.1.4 Auto setup when a packet is sent

Next Header.

- The structure is found and the values are padded.

Header Ext Length.

- Length excluding the first 8 octets
- In units of 8 octets.

6.4.1.5 Auto setup when a packet is received

Next Header.

Any

Header Ext Length.

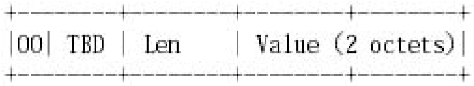
The structure is analyzed based on the received values.

6.4.2 Router Alert Option

6.4.2.1 Reference RFC

draft-ietf-ipngwg-ipv6rour-alert-04.txt, IPv6 Router Alert Option, February 1998

6.4.2.2 Format



```
+--------+--------+--------+--------+
|00| TBD | Len    | Value (2 octets)|
+--------+--------+--------+--------+
```

6.4.2.3 Default

Table 27 Router Alert (Option)

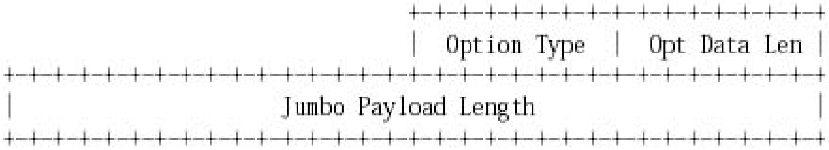
Name	Type	Size	Default (Send)	Default (Receive)
Option Type	uint	8bit	0x05	0x05
Opt Data Length	uint	8bit	2	2
Value	uint	16bit	Required	Required

6.4.3 Jumbo Payload Option

6.4.3.1 Reference RFC

draft-ietf-ipngwg-jumbograms-00.txt,IPv6 Jumbograms, Feburary 1999

6.4.3.2 Format



```
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                |  Option Type  |  Opt Data Len |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Jumbo Payload Length                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.4.3.3 Default

Table 28 Jumbo Payload (Option)

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8	0xC2	0xC2
Opt Data Length	uint	8	4	4
Jumbo Payload Length	uint	32	auto	any

6.4.3.4 Auto setup when a packet is sent

Jumbo Payload Length.

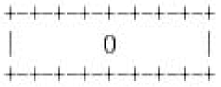
- Packet length excluding the IPv6 Header
- The units are octets

6.4.4 Pad1

6.4.4.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.4.4.2 Format



```
+-+-+-+-+-+-+-+-+
|       0       |
+-+-+-+-+-+-+-+-+
```

6.4.4.3 Default

Table 29 Pad1 (Optional)

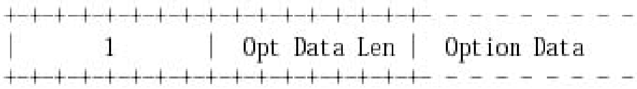
Name	Type	Size	Default (Send)	Default (Receive)
Option Type	uint	8bit	0	0

6.4.5 PadN

6.4.5.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.4.5.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
|       1       | Opt Data Len |  Option Data
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- - - - - - - - -
```

6.4.5.3 Default

Table 30 PadN (Option)

Name	Type	Size	Default (Send)	Default (Receive)
Option Type	uint	8bit	1	1
Opt Data Length	uint	8bit	auto	any
Pad	data		0(?)	0(?)

6.4.5.4 Auto setup when a packet is sent

The procedure for calculating Opt Data Length

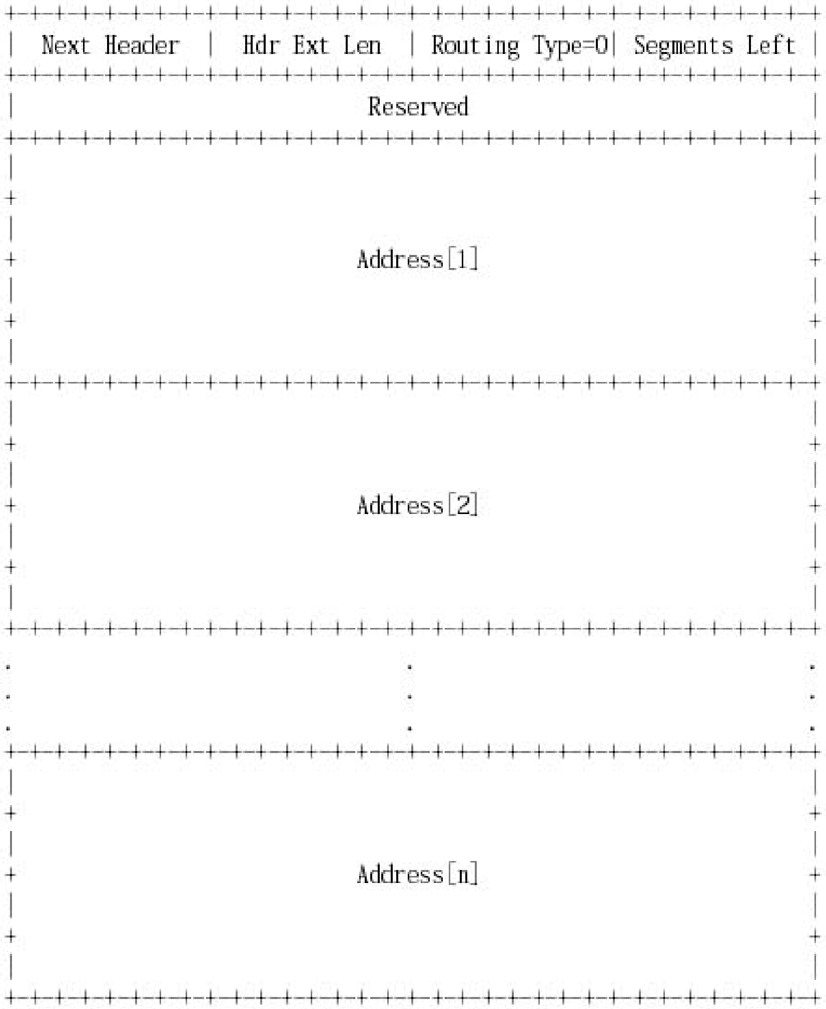
- Padded length
- Zero is valid.

6.4.6 Type 0 Routing Header

6.4.6.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.4.6.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  | Routing Type=0| Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                           Address[1]                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                           Address[2]                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                               .                               .
.                               .                               .
.                               .                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                           Address[n]                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.4.6.3 Default

Table 31 Type 0 Routing Header

Name	Type	Size	Default (Send)	Default (Receive)
Next Header	uint	8bit	auto	any
Hdr Ext Length	uint	8bit	auto	any
Routing Type	uint	8bit	0	0
Segment Left	uint	8bit	0	0
Rserved	uint	32bit	0	0
Address	IPv6 Address	128bit	Required	Required
Repeat as needed				

6.4.6.4 Auto setup when a packet is sent

Next Header.
Hdr Ext Len.

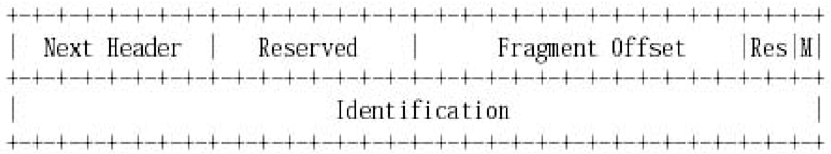
- Length of the routing header excluding the first eight octets.
- In units of 8 octets.

6.4.7 Fragment Header

6.4.7.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.4.7.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |   Reserved    |      Fragment Offset    |Res|M|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Identification                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.4.7.3 Default

Table 32 Fragment Header

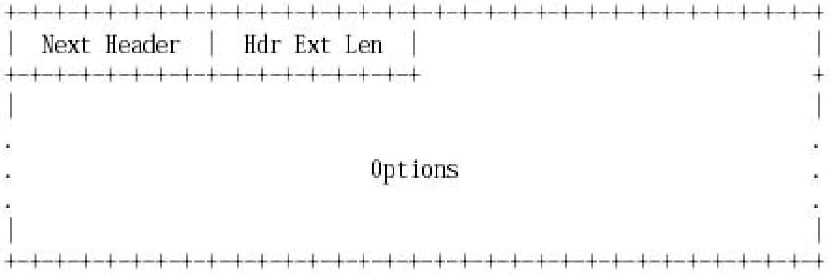
Name	Type	Size	Default (Send)	Default (Receive)
Next Header	uint	8bit	Required	any
Reserved1	uint	8bit	0	0
Fragment Offset	uint	13bit	0	0
Reserved2	uint	2bit	0	0
M Flag	uint	1bit	0	0
Identification	uint	32bit	0	0

6.4.8 Destination Option Header

6.4.8.1 Reference RFC

RFC2460, Internet Protocol, Version 6 (IPv6) Specification, December 1998

6.4.8.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
|                                                               |
.                                                               .
.                            Options                            .
.                                                               .
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.4.8.3 Default

Table 33 Destination Option Header

Name	Type	Size	Default (Send)	Default (Receive)
Next Header	uint	8bit	auto	any
Header Ext Len	uint	8bit	auto	any
(tunnel encapslation)				
(pad1)				
(padN)				

6.4.8.4 Auto setup when a packet is sent

Next Header.

Header Ext Len.

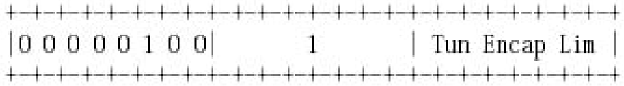
- Length of the destination option header excluding the first eight octets.
- In units of 8 octets.

6.4.9 Tunnel Encapsulation Header

6.4.9.1 Reference RFC

RFC2473, Generic Packet Tunneling in IPv6 Specification, December 1998

6.4.9.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0 0 0 0 0 1 0 0|       1       | Tun Encap Lim |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.4.9.3 Default

Table 34 Tunnel Encapsulation Header

Name	Type	Size	Default (Send)	Default (Receive)
Option Type	uint	8bit	0x04	0x04
Opt Data Len	uint	8bit	1	1
Limit	uint	8bit	0	0

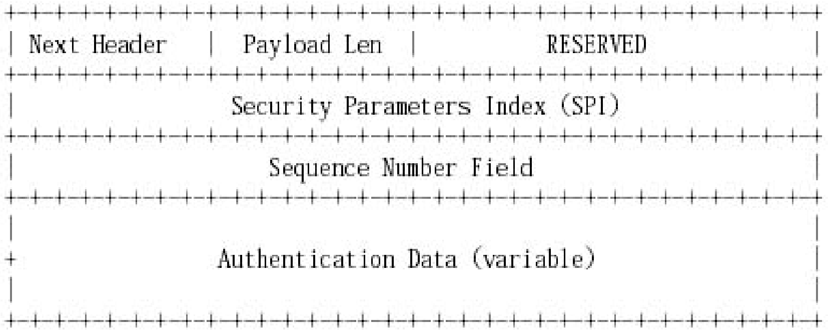
6.5 IPsec

6.5.1 Authentication Header

6.5.1.1 Reference RFC

RFC2402, IP Authentication Header, November 1998

6.5.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Payload Len  |          RESERVED             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Security Parameters Index (SPI)               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Sequence Number Field                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                Authentication Data (variable)                 |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.5.1.3 Default

Table 35 Authentication

Name	Type	Size	Default (Send)	Default (Receive)
Next Header	uint	8bit	auto	any
Payload Length	uint	8bit	auto	any
Reserved	uint	16bit	0	0
SPI	uint	32bit	0	0
Sequence Number	uint	32bit	0	0
Algorithm	ref			

6.5.1.4 Auto setup when a packet is sent

Payload Length.

- Length of the Authentication Header 2
- In units of 4 bytes.

6.5.1.5 Other

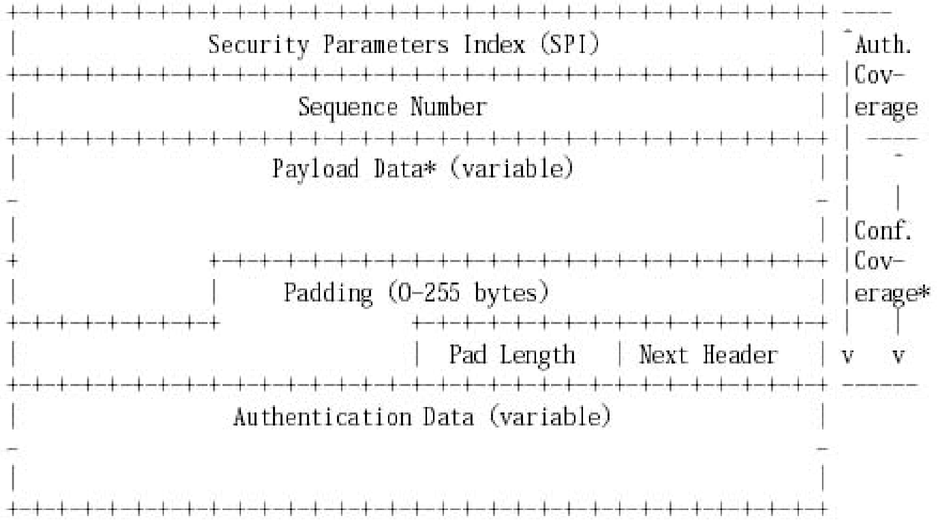
- Because the size of the hash value differs according to the hash function, it must be adjusted to the 8 byte boundary by padding. Therefore a padded member is required.

6.5.2 Encapsulating Security Payload

6.5.2.1 Reference RFC

RFC2406, IP Encapsulating Security Payload (ESP), November 1998

6.5.2.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ----
|               Security Parameters Index (SPI)                 | ˉAuth.
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|                      Sequence Number                          | |erage
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | ----
|                    Payload Data* (variable)                   | |   ˉ
-                                                               - |   |
|                                                               | |Conf.
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|               |     Padding (0-255 bytes)                     | |erage*
+-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |   |
|                               |  Pad Length   | Next Header   | v   v
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
|                 Authentication Data (variable)                |
-                                                               -
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.5.2.3 Default

Table 36 Encapsulating Security Payload

Name	Type	Size	Default (Send)	Default (Receive)
SPI	uint	32bit	0	0
Sequence Number	uint	32bit	0	0
Pad Length	uint	8bit	auto	any
Next Header	uint	8bit	auto	any
Algorithm	ref			

6.5.3 AHAlgorithm

6.5.3.1 Reference RFC

RFC2403, The Use of HMAC-MD5-96 within ESP and AH, November 1998

RFC2404, The Use of HMAC-SHA-1-96 within ESP and AH

6.5.3.2 Default

Table 37 AH Algorithm Block

Name	Type	Size	Default (Send)	Default (Receive)
auth	auth		Required	Required

6.5.4 ESPAlgorithm

6.5.4.1 Reference RFC

RFC2403, The Use of HMAC-MD5-96 within ESP and AH, November 1998

RFC2404, The Use of HMAC-SHA-1-96 within ESP and AH, November 1998

RFC2451, The ESP CBC-Mode Cipher Algorithms, November 1998

RFC2410, The NULL Encryption Algorithm and Its Use With IPsec, November 1998

RFC2405, The ESP DES-CBC Cipher Algorithm With Explict IV, November 1998

6.5.4.2 Default

Table 38 ESP Algorithm Block

Name	Type	Size	Default (Send)	Default (Receive)
pad	esppad		auto	any
crypt	crypt		Required	Required
auth	auth		null_auth()	null_auth()

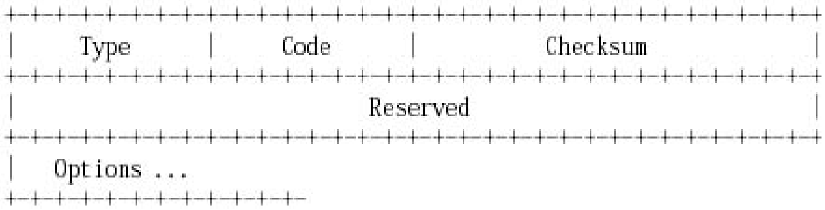
6.6 ICMPv6 (ND) (RFC2461)

6.6.1 Router Solicitation

6.6.1.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Reserved                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

6.6.1.3 Default

Table 39 Router Solicitation

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	133	133
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Reserved	uint	32bit	0	0
(SSL option)				
(TLL option)	bad-opt			
(MTU option)	bad-opt			
(Prefix option)	bad-opt			
(redirected hd option)	bad-opt			

6.6.1.4 Auto setup when a packet is sent

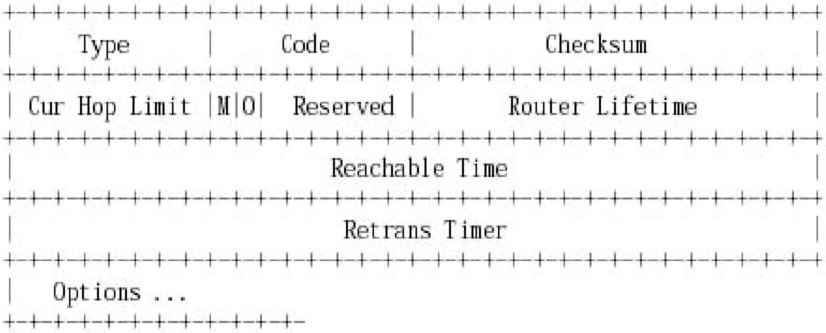
checksum.

6.6.2 Router Advertisement

6.6.2.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.2.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Cur Hop Limit |M|O|  Reserved |       Router Lifetime         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reachable Time                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Retrans Timer                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

6.6.2.3 Default

Table 40 Router Advertisement

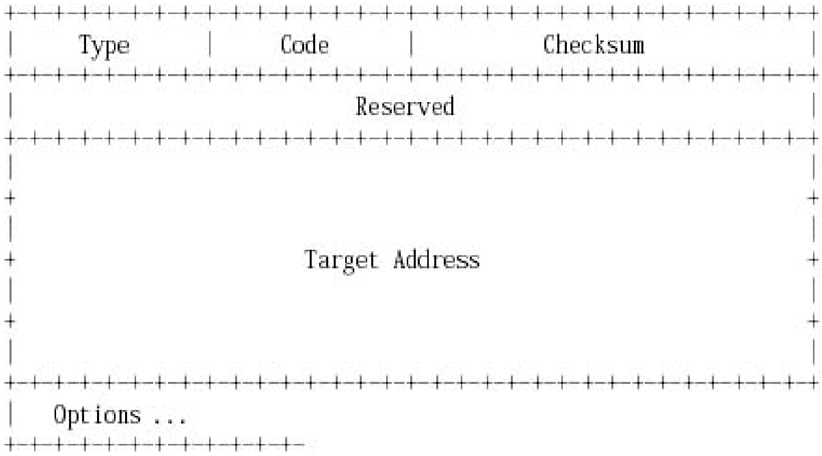
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	134	134
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
CurHopLimit	uint	8bit	0	0
M Flag	uint	1bit	0	0
O Flag	uint	1bit	0	0
Reserved	uint	6bit	0	0
Life Time	uint	16bit	0	0
Reachable Time	uint	32bit	0	0
Retrans Timer	uint	32bit	0	0
(SLL option)				
(MTU Option)				
(Prefix Option)				
(TLL option)	bad-opt			
(redirected hd optoin)	bad-opt			

6.6.3 Neighbor Solicitation

6.6.3.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.3.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Target Address                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

6.6.3.3 Default

Table 41 Neighbor Solicitation

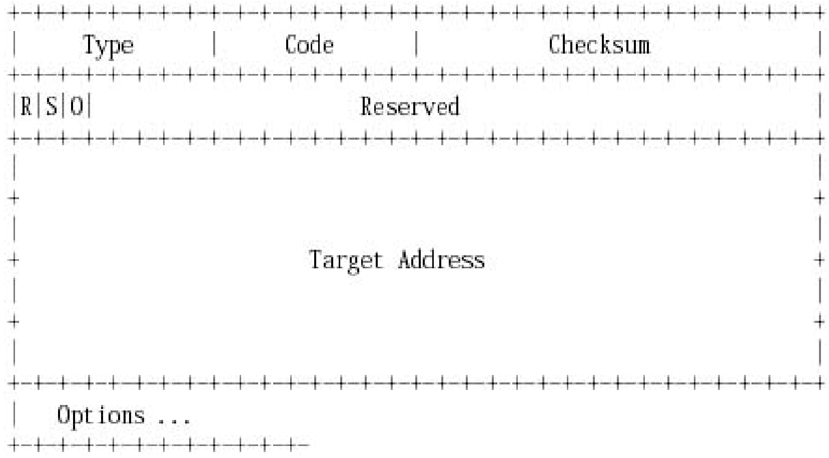
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	135	135
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Reserved	uint	32bit	0	0
Target Address	IPv6 Address	128bit	Target Address	Tester Address
(SLL option)				
(TLL option)	bad-opt			
(MTU option)	bad-opt			
(Prefix option)	bad-opt			
(redirected hd option)	bad-opt			

6.6.4 Neighbor Advertisement

6.6.4.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.4.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|R|S|O|                     Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Target Address                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

6.6.4.3 Default

Table 42 Neighbor Advertisement

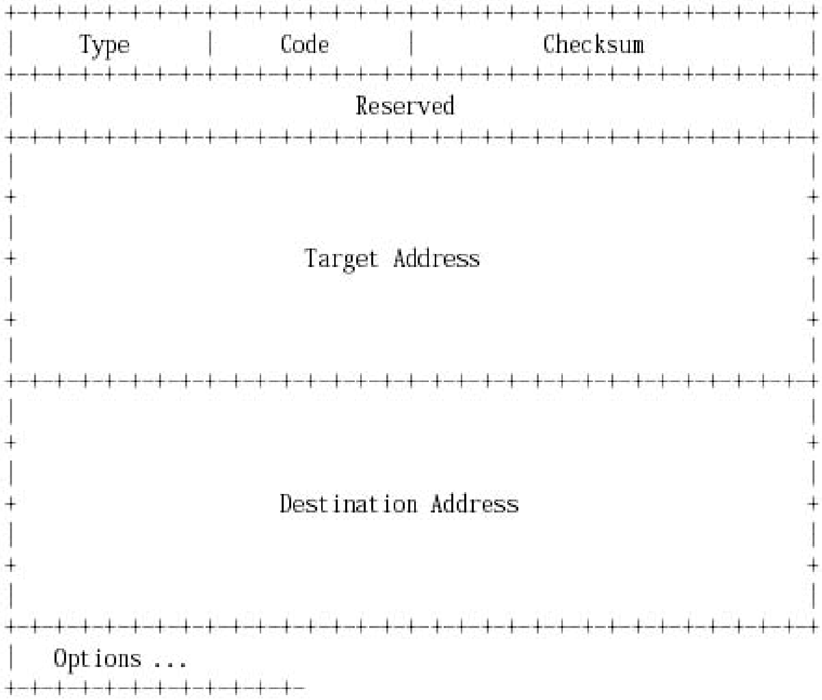
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	136	136
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
R Flag	uint	1bit	0	0
S Flag	uint	1bit	0	0
O Flag	uint	1bit	0	0
Reserved	uint	32bit	0	0
Target Address	IPv6 Address	128bit	Tester Address	Targ Address
(TLL option)				
(SLL option)	bad-opt			
(Prefix option)	bad-opt			
(MTU option)	bad-opt			
(redirected hd option)	bad-opt			

6.6.5 Redirect

6.6.5.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.5.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Target Address                          +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                     Destination Address                       +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Options ...
+-+-+-+-+-+-+-+-+-+-+-+-
```

6.6.5.3 Default

Table 43 Redirect

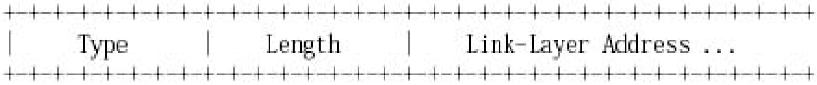
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	137	137
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Reserved	Reserved	32bit	0	0
Target Address	IPv6 Address	128bit	Required	Required
Destination Address	IPv6 Address	128bit	Required	Required
(TLL option)				
(Redirect Hd Option)				
(SLL option)	bad-opt			
(MTU option)	bad-opt			
(Prefix option)	bad-opt			

6.6.6 Source Link Layer Address Option

6.6.6.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.6.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |    Link-Layer Address ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.6.6.3 Default

Table 44 Source Link Layer Address Option (option)

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	1	1
Length	uint	8bit	auto	any
Link Layer Address	Ether Address	48bit	Target Ether Addr	Tester Ether Addr

6.6.6.4 Other

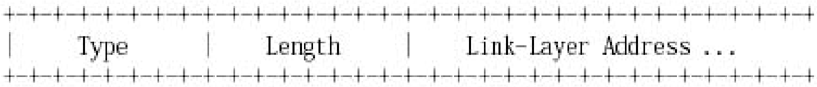
Originally, the length is variable to accommodate a link layer address other than that of Ether. However, as only Ether is applies, the length is fixed this time.

6.6.7 Tagrget Link Layer Address option

6.6.7.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.7.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |    Link-Layer Address ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.6.7.3 Default

Table 45 Target Link Layer Address Option (option)

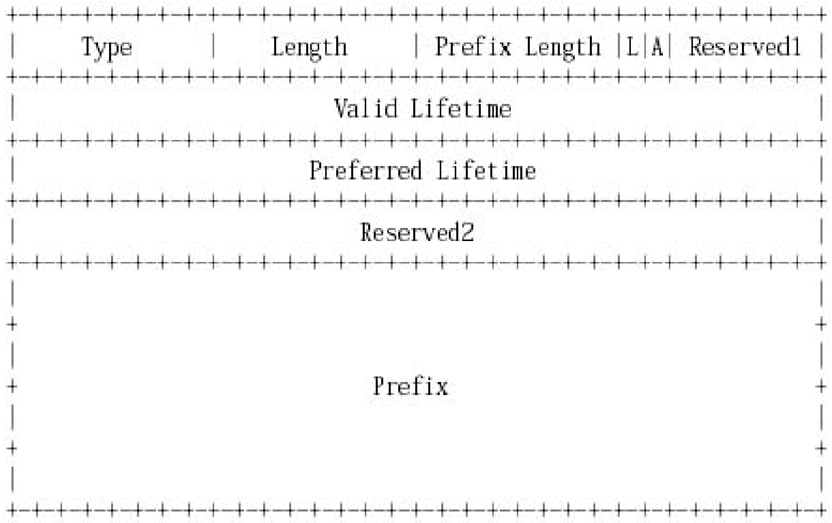
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	2	2
Length	uint	8bit	auto	use
Link Layer Address	Ether Addr	48bit	Target Ether Addr	Tester Ether Addr

6.6.8 Prefix Information Option

6.6.8.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.8.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Prefix Length |L|A| Reserved1 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Valid Lifetime                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Preferred Lifetime                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved2                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                            Prefix                             +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.6.8.3 Default

Table 46 Prefix Information (Option)

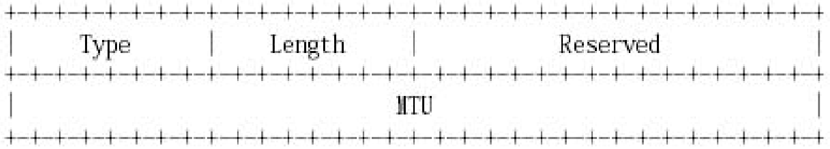
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	3	3
Length	uint	8bit	4	4
Prefix Length	uint	8bit	64	64
L Flag	uint	1bit	0	0
A Flag	uint	1bit	0	0
Reserved1	uint	6bit	0	0
Valid Lifetime	uint	32bit	0	0
Preferred Lifetime	uint	32bit	0	0
Reserved2	uint	32bit	0	0
Prefix	IPv6 Address	128bit	Required	Required

6.6.9 MTU Option

6.6.9.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.9.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              MTU                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.6.9.3 Default

Table 47 MTU (Option)

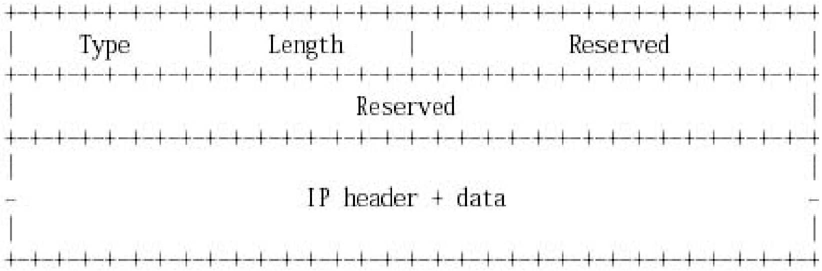
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	5	5
Length	uint	8bit	1	1
Resereved	uint	16bit	0	0
MTU	uint	32bit	1500	1500

6.6.10 Redirected Header (Option)

6.6.10.1 Reference RFC

RFC2461,Neighbor Discovery for IP Version 6(IPv6), December 1998

6.6.10.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
-                       IP header + data                        -
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.6.10.3 Default

Table 48 Redirected Header (option)

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	4	4
Length	uint	8bit	auto	any
Reserved	uint	48bit	0	0
payload	payload		Required	Required

6.6.10.4 Auto setup when a packet is sent

Length

- Length of the overall option
- In units of 8 octets.

6.6.10.5 Other

Using 8 octets for unit means that the length of the payload portion must be a multiple of 8 octets.

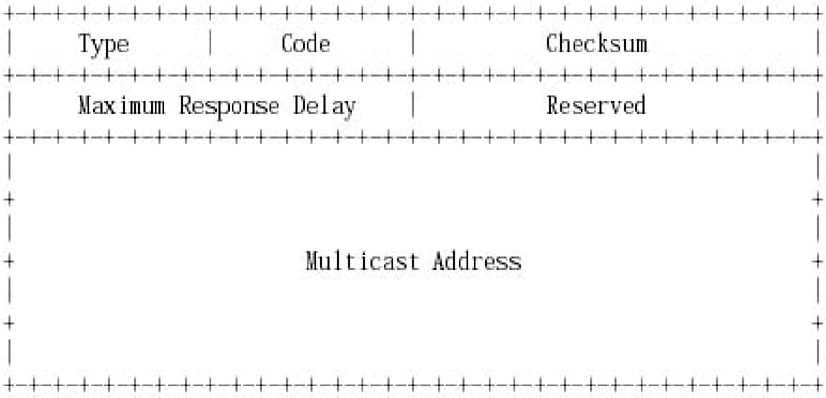
6.7 Information Messages

6.7.1 Multicast Lister Query

6.7.1.1 Reference RFC

draft-ietf-ipngwg-mld-01.txt, Multicast Listener Discovery (MLD) for IPv6, February 1999

6.7.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Maximum Response Delay    |          Reserved             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Multicast Address                       +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.7.1.3 Default

Table 49 Multicast Listener

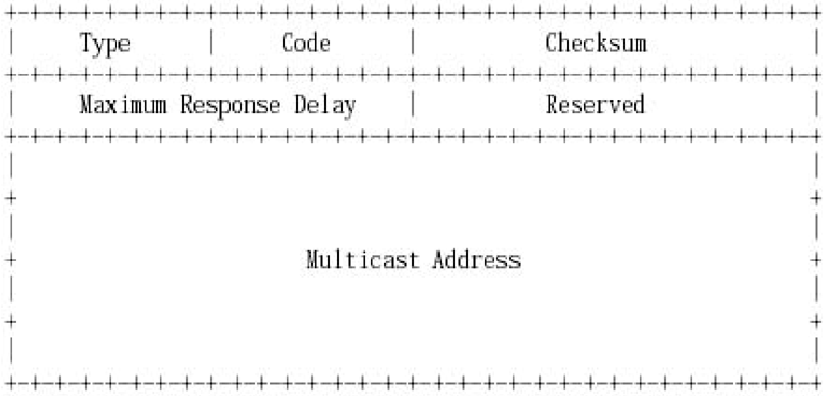
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	130	130
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Max Response Delay	uint	16bit	0	0
Reserved	uint	16bit	0	0
Multicast Address	IPv6 Address	128bit	Required	Required

6.7.2 Multicast Listener Report

6.7.2.1 Reference RFC

draft-ietf-ipngwg-mld-01.txt, Multicast Listener Discovery (MLD) for IPv6, February 1999

6.7.2.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Maximum Response Delay    |          Reserved             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Multicast Address                       +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.7.2.3 Default

Table 50 Multicast Listener Report

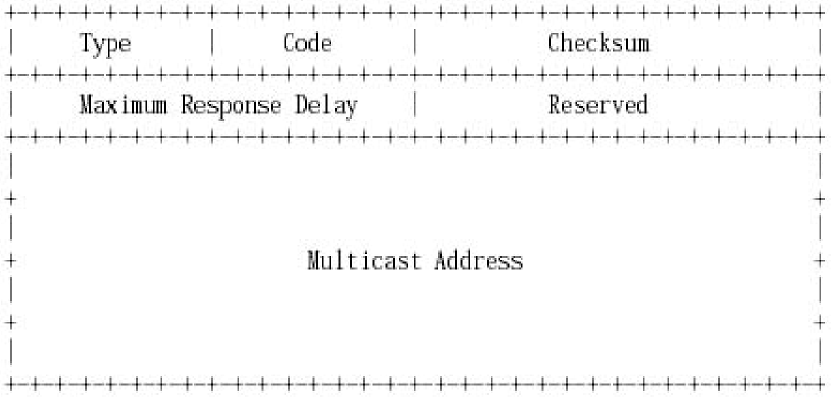
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	131	131
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Max Response Delay	uint	16bit	0	0
Reserved	uint	16bit	0	0
Multicast Address	IPv6 Address	128bit	Required	Required

6.7.3 Multicast Listener Done

6.7.3.1 Reference RFC

draft-ietf-ipngwg-mld-01.txt, Multicast Listener Discovery (MLD) for IPv6, February 1999

6.7.3.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Maximum Response Delay    |          Reserved             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                       Multicast Address                       +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.7.3.3 Default

Table 51 Multicast Listener Done

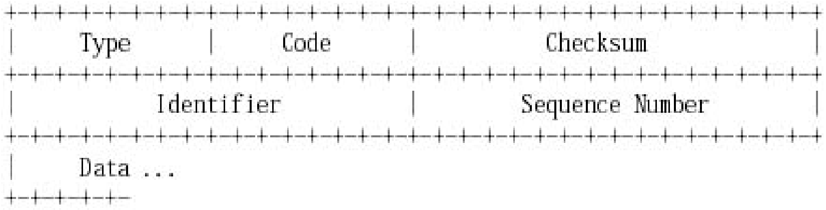
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	132	132
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Max Response Delay	uint	16bit	0	0
Reserved	uint	16bit	0	0
Multicast Address	IPv6 Address	128bit	Required	Required

6.7.4 ICMP Echo Request

6.7.4.1 Reference RFC

RFC2463, Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6(IPv6) Specification, December 1998

6.7.4.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Identifier          |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Data ...
+-+-+-+-+-
```

6.7.4.3 Default

Table 52 ICMP Echo Request

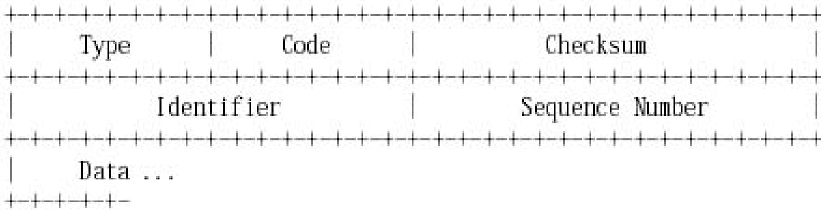
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	128	128
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Identifier	uint	16bit	0	0
Sequence Number	uint	16bit	0	0
payload	payload		Required	Required

6.7.5 Echo Reply

6.7.5.1 Reference RFC

RFC2463, Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6(IPv6) Specification, December 1998

6.7.5.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Identifier          |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Data ...
+-+-+-+-+-
```

6.7.5.3 Default

Table 53 ICMP Echo Reply

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	129	129
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Identifier	uint	16bit	0	0
Sequence Number	uint	16bit	0	0
payload	payload		Required	Required

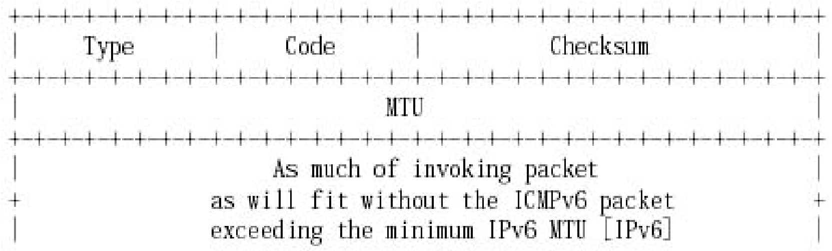
6.8 Error Message

6.8.1 Packet Too Big

6.8.1.1 Reference RFC

RFC2463, Intern Control Message Protocol (ICMPv6) for the Internet Protocol Version 6(IPv6) Specification, December 1998

6.8.1.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             MTU                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    As much of invoking packet                 |
+               as will fit without the ICMPv6 packet           +
|               exceeding the minimum IPv6 MTU [IPv6]           |
```

6.8.1.3 Default

Table 54 Packet Too Big

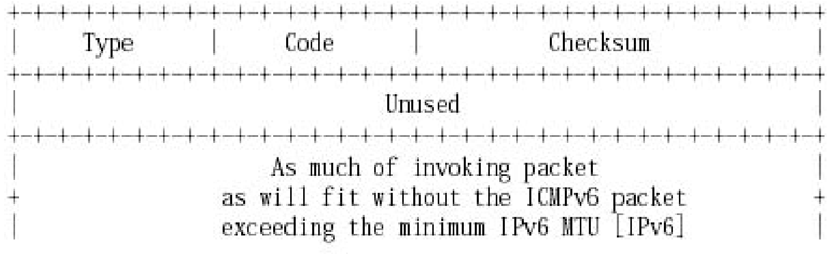
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	2	2
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
MTU	uint	32bit	Required	Required
payload	payload		Required	Required

6.8.2 Destination Unreachable

6.8.2.1 Reference RFC

RFC2463, Intern Control Message Protocol (ICMPv6) for the Internet Protocol Version 6(IPv6) Specification, December 1998

6.8.2.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Unused                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    As much of invoking packet                 |
+                as will fit without the ICMPv6 packet          +
|               exceeding the minimum IPv6 MTU [IPv6]           |
```

6.8.2.3 Default

Table 55 Destination Unreachable Message

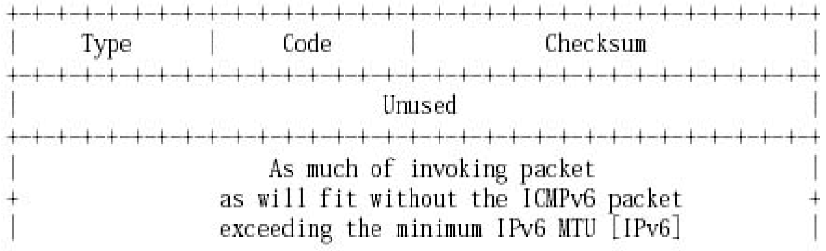
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	1	1
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Unused	uint	32bit	0	0
payload	payload		Required	Required

6.8.3 Time Exceeded Messages

6.8.3.1 Reference RFC

RFC2463, Intern Control Message Protocol (ICMPv6) for the Internet Protocol Version 6(IPv6) Specification, December 1998

6.8.3.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Unused                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    As much of invoking packet                 |
+                as will fit without the ICMPv6 packet          +
|               exceeding the minimum IPv6 MTU [IPv6]           |
```

6.8.3.3 Default

Table 56 Time Exceeded Message

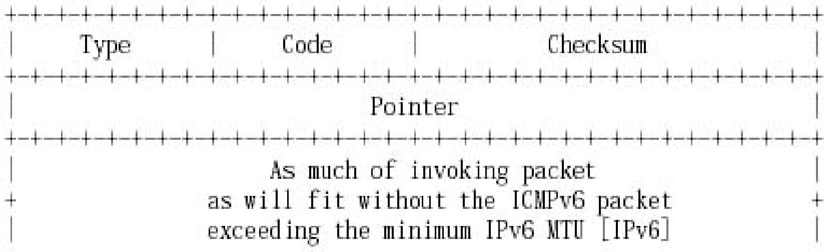
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	3	3
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Unused	uint	32bit	0	0
payload	payload		Required	Required

6.8.4 Parameter Problem Messages

6.8.4.1 Reference RFC

RFC2463, Intern Control Message Protocol (ICMPv6) for the Internet Protocol Version 6(IPv6) Specification, December 1998

6.8.4.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Pointer                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    As much of invoking packet                 |
+               as will fit without the ICMPv6 packet           +
|               exceeding the minimum IPv6 MTU [IPv6]           |
```

6.8.4.3 Default

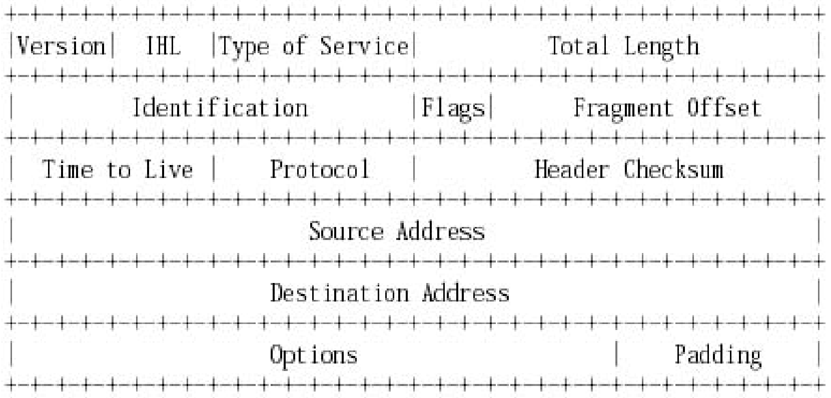
Table 57 Parameter Problem Messages

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	4	4
Code	uint	8bit	[12]	[12]
Checksum	uint	16bit	auto	check
Pointer	uint	32bit	Required	Required
payload	payload		Required	Required

6.9 IPv4

6.9.1 IPV4 Header

6.9.1.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.9.1.2 Default

Table 58 IPV4 Header

Name	Type	Size	Default (Send)	Default (Receive)
Version	uint	4bit	4 (IPv4)	4 (IPv4)
IHL	uint	4bit	5	use
Type Of Service	uint	8bit	0	any
Total Length	uint	16bit	auto	use
Identifier	uint	16bit	0	any
Flags	uint	4bit	0	0
Fragment Offset	uint	12bit	0	0
TTL	uint	8bit	255	any
Protocol	uint	8bit	auto	auto
Header Checksum	uint	16bit	auto	check
Source Address	IPv4 Address	32bit	Required	Required
Destination Address	IPv4 Address	32bit	Required	Required
(option)	ref			
(Padding)	date			

6.9.2 End Of Option List Option

6.9.2.1 Default

Table 59 End Of Option List Option

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	0	0

6.9.3 No Operation Option

6.9.3.1 Default

Table 60 IPv4 No Operation Option

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	1	1

6.9.4 Loose Source Route Option

6.9.4.1 Default

Table 61 IPv4 Loose Source Route Option

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	131	131
Length	uint		auto	auto
Pointer				
RouteData	IPv4			
continued				

6.9.5 Strict Source Route Option

6.9.5.1 Default

Table 62 Strict Source Route Option

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	137	137
Length	uint		auto	auto
Pointer				
RouteData	IPv4			
continued				

6.9.6 Record route Option

6.9.6.1 Default

Table 63 IPv4 Record Route Option

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	7	7
Length	uint		auto	auto
Pointer				
RouteData	IPv4			
continued				

6.9.7 Timestamp Option

6.9.7.1 Default

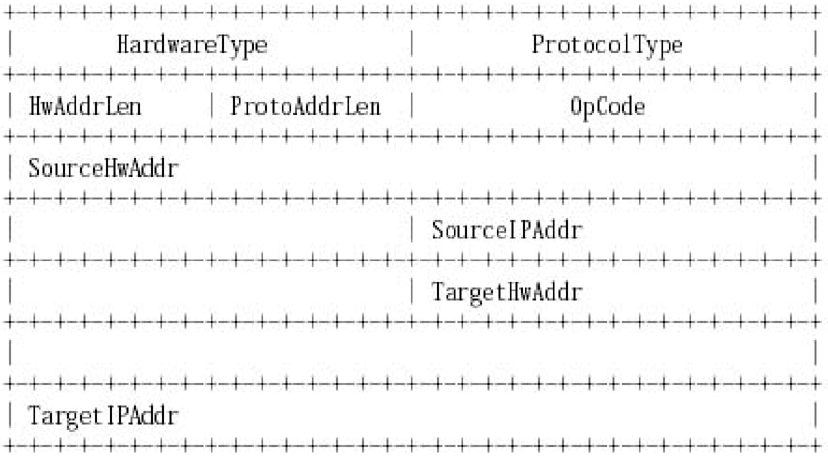
Table 64 IPv4 Time Stamp Option

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	68	68
Length	uint		auto	auto
Pointer				
Overflow				
Flag				
Timestamp				
continued				

6.10 ARP

6.10.1 ARP Packet

6.10.1.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          HardwareType         |          ProtocolType         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| HwAddrLen     | ProtoAddrLen  |             OpCode            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SourceHwAddr                                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | SourceIPAddr                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | TargetHwAddr                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TargetIPAddr                                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

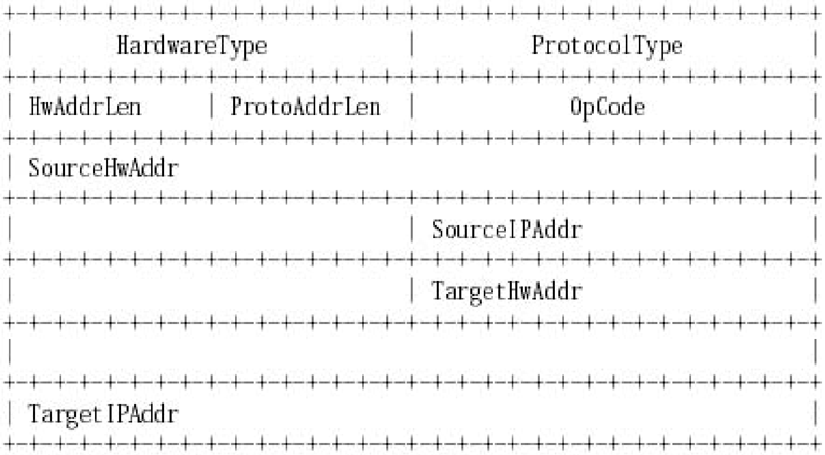
6.10.1.2 Default

Table 65 ARP

Name	Type	Size	Default (Send)	Default (Receive)
Hardware	uint	16bit	1	1
Protocol	uint	16bit	2048	2048
HLEN	uint	8bit	6	6
PLEN	uint	8bit	4	4
Opration	uint	16bit	2	1
SenderHAddr	ether	48bit	Terter Address	Target Address
SenderPAddr	v4	32bit	Required	Required
TargetHAddr	ether	48bit	Target Address	Tester Address
TargetPAddr	v4	32bit	Required	Required

6.10.2 RARP Packet

6.10.2.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        HardwareType           |          ProtocolType         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| HwAddrLen     | ProtoAddrLen  |             OpCode            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SourceHwAddr                                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | SourceIPAddr                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | TargetHwAddr                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| TargetIPAddr                                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.10.2.2 Default

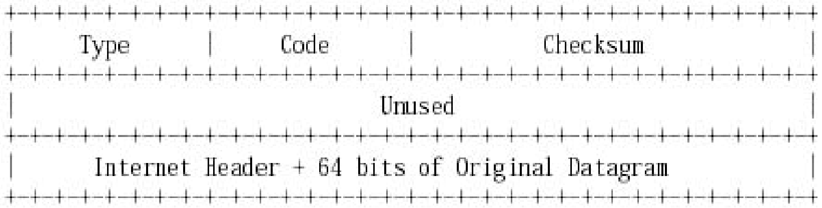
Table 66 RARP

Name	Type	Size	Default (Send)	Default (Receive)
Hardware	uint	16bit	1	1
Protocol	uint	16bit	2048	2048
HLEN	uint	8bit	6	6
PLEN	uint	8bit	4	4
Opration	uint	16bit	3	4
SenderHAddr	ether	48bit	Terter Address	Target Address
SenderPAddr	v4	32bit	Required	Required
TargetHAddr	ether	48bit	Target Address	Tester Address
TargetPAddr	v4	32bit	Required	Required

6.11 ICM Pv4

6.11.1 Destination Unreachable Message

6.11.1.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Unused                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Datagram           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

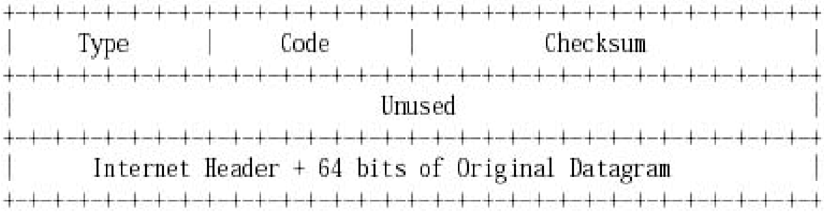
6.11.1.2 Default

Table 67 ICMPv4 Destination Unreachable Message

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	3	3
Code	uint	8bit	Required	Required
Checksum	uint	16bit	auto	check
Unused	uint	32bit	0	0
payload	payload		Required	Required

6.11.2 Time Exceeded Message

6.11.2.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Unused                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Datagram           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

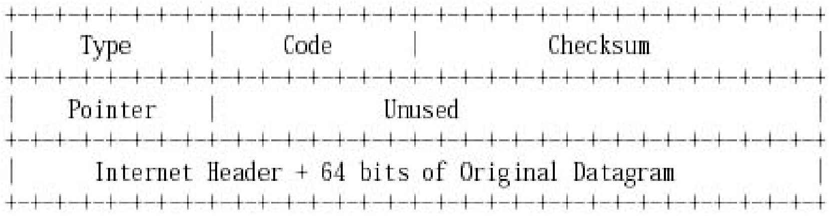
6.11.2.2 Default

Table 68 ICM Pv4 Tune Exceeded Message

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	11	11
Code	uint	8bit	Required	Required
Checksum	uint	16bit	auto	check
Unused	uint	32bit	0	0
payload	payload		Required	Required

6.11.3 Parameter Problem Message

6.11.3.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Pointer    |                   Unused                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Datagram           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

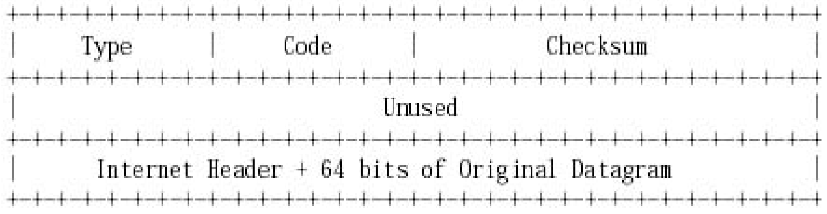
6.11.3.2 Default

Table 69 ICMPv4 Parameter Problem Message

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	12	12
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Pointer	uint	8bit	0	0
Unused	uint	32bit	0	0
payload	payload		Required	Required

6.11.4 Source Quench Message

6.11.4.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Unused                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Datagram           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

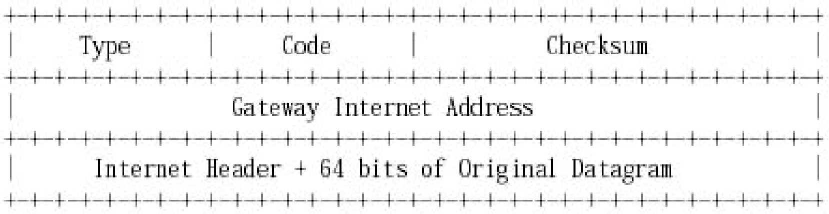
6.11.4.2 Default

Table 70 ICMPv4 Source Quench Message

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	4	4
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Unused	uint	32bit	0	0
payload	payload		Required	Required

6.11.5 Redirect Message

6.11.5.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Gateway Internet Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Internet Header + 64 bits of Original Datagram           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

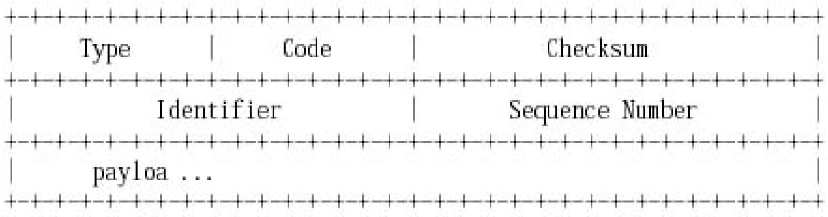
6.11.5.2 Default

Table 71 ICMPv4 Redirect Message

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	5	5
Code	uint	8bit	Required	Required
Checksum	uint	16bit	auto	check
Address	IPv4	32bit	Required	Required
payload	payload		Required	Required

6.11.6 Echo Request Message

6.11.6.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Identifier          |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     payloa ...                                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

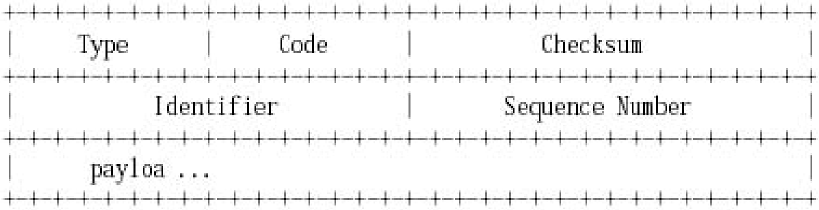
6.11.6.2 Default

Table 72 ICMPv4 Echo Request

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	8	8
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Identifier	uint	16bit	0	0
SequenceNumber	uint	16bit	0	0
payload	payload		Required	Required

6.11.7 Echo Reply Message

6.11.7.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Identifier          |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     payloa ...                                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.11.7.2 Default

Table 73 ICMPv4 Echo Reply

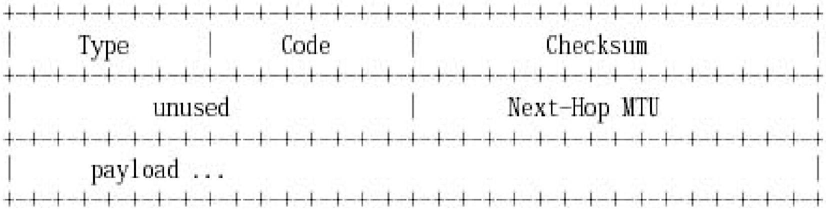
Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	0	0
Code	uint	8bit	0	0
Checksum	uint	16bit	auto	check
Identifier	uint	16bit	0	0
SequenceNumber	uint	16bit	0	0
payload	payload		Required	Required

6.11.8 Packet Too Big

6.11.8.1 Reference RFC

RFC1191

6.11.8.2 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           unused              |         Next-Hop MTU          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     payload ...                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.11.8.3 Default

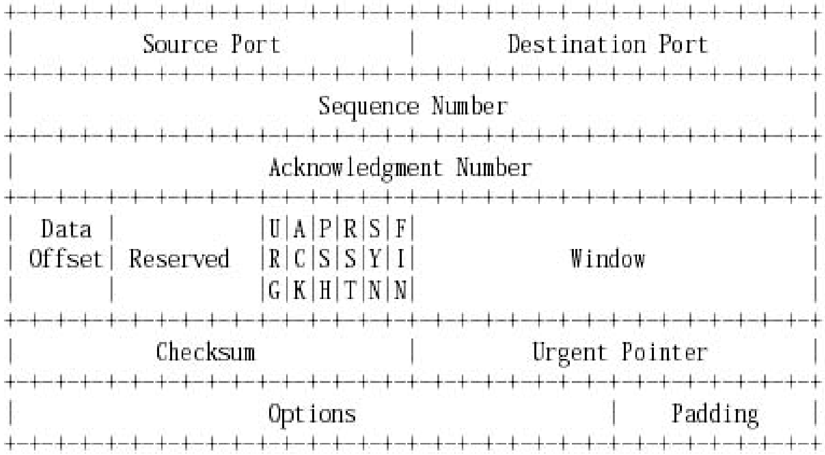
Table 74 ICMPv4 Packet Too Big

Name	Type	Size	Default (Send)	Default (Receive)
Type	uint	8bit	3	3
Code	uint	8bit	4	4
Checksum	uint	16bit	auto	check
Unused	uint	16bit	0	0
NextHopMTU	uint	16bit	0	0
payload	payload		Required	Required

6.12 TCP

6.12.1 TCP Header

6.12.1.1 Format



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgment Number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Data |           |U|A|P|R|S|F|                               |
| Offset| Reserved  |R|C|S|S|Y|I|            Window             |
|       |           |G|K|H|T|N|N|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.12.1.2 Default

Table 75 TCP header

Name	Type	Size	Default (Send)	Default (Receive)
Source Port	uint	16bit	Required	Required
Destinatino Port	uint	16bit	Required	Required
Sequence Number	uint	32bit	0	0
Ack Number	uint	32bit	0	0
Data Offset	uint	4bit	auto	use
Reserved	reserve	6bit	0	0
URG Flag	flag	1bit	0	0
ACK Flag	flag	1bit	0	0
PSH Flag	flag	1bit	0	0
RST Flag	flag	1bit	0	0
SYN Flag	flag	1bit	0	0
FIN Flag	flag	1bit	0	0
Window	uint	16it	0	0
Checksum	uint	16bit	auto	check
Urgent Pointer	uint	16bit	0	0
(Options)	ref			
(Padding)	ref			
payload	payload			

6.12.2 End Of Option List Option

6.12.2.1 Default

Table 76 IPv4 End Of Option List Option

Name	Type	Size	Default (Send)	Default (Receive)
Kind	uint	8bit	0	0

6.12.3 No operation Option

6.12.3.1 Default

Table 77 IPv4 No Operation Option

Name	Type	Size	Default (Send)	Default (Receive)
Kind	uint	8bit	1	1

6.12.4 Maximum Segment Size Option

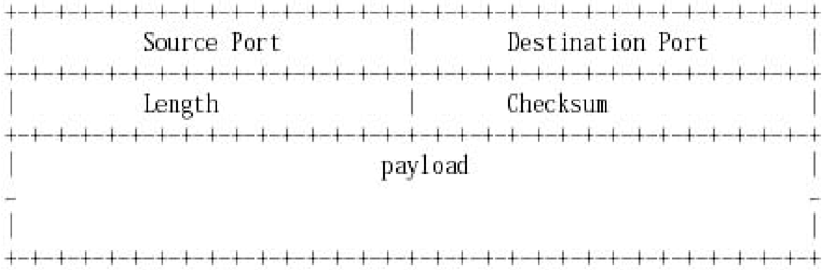
6.12.4.1 Default

Table 78 IPv4 Maximum Segment Size Option

Name	Type	Size	Default (Send)	Default (Receive)
Kind	uint	8bit	2	2
Length				
MaxSegSize				

6.13 UDP

6.13.1 Any UDP



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Length               |       Checksum                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            payload                            |
-                                                               -
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

6.13.1.1 Default

Table 79 UDP

Name	Type	Size	Default (Send)	Default (Receive)
Source Port	uint	16bit	Required	Required
Destinatino Port	uint	16bit	Required	Required
Length	uint	16bit	auto	any
Checksum	uint	16bit	auto	check
payload	payload		Required	Required