
IPv6 検証ツール Users Manual

Rev. 2.3

このドキュメントは、IPv6 検証ツール Release 1.0 を対象
としています。

1 テスタ概要	1
1.1 テスト環境	1
1.2 動作環境	1
1.2.1 ハードウェア環境	1
1.2.2 ソフトウェア環境	1
1.3 インストール	2
1.4 ディレクトリ構成	3
2 設定ファイルフォーマット	4
2.1 Tester Node Config File	4
2.1.1 目的	4
2.1.2 文法	4
2.1.3 例	4
2.1.4 設定項目	5
2.2 Node under Test Config File	6
2.2.1 目的	6
2.2.2 考慮すべき点	6
2.2.3 文法	6
2.2.4 例	6
2.2.5 設定項目	7
3 コマンドシンタックス	8
3.1 テスト自動実行プログラム (autorun)	8
3.1.1 概要	8
3.1.2 コマンドライン	8
3.1.3 オプション	8
3.1.4 出力	9
3.1.5 返り値	9
3.1.6 INDEX ファイルのフォーマット	9
3.2 パケット定義文法チェッカー (pktlint)	10
3.2.1 オプション	10
3.2.2 返り値	10
3.3 リモート制御プログラム	11
3.3.1 概要	11
3.3.2 コマンド	11
3.3.3 返り値	11
4 パケットの定義	12
4.1 概略	12
4.1.1 定義の記述法	12
4.2 型	13
4.2.1 参照型	13
4.2.2 オプション型	14
4.2.3 データ型	15
4.2.4 crypt 型	16
4.2.5 auth 型	17
4.2.6 esppad 型	18
4.2.7 ether 型	19
4.2.8 IPv6 Addr 型	19
4.2.9 IPv4 Addr 型	20
4.2.10 payload 型	20
4.2.11 uint 型	21
4.3 アドレス変換関数について	22

4.3.1Ether アドレス	22
4.3.2IPv6 アドレス	24
4.3.3IPv4 アドレス	26
4.4 コメントの記述	26
4.5 パケット構造の定義	27
4.5.1 概要	27
4.5.2Frame_Ether 定義	27
4.5.3Packet_XX 定義	28
4.5.4Payload 定義	28
4.6 その他のタイプ定義	29
4.7 オプションの定義と比較	30
4.7.1 オプションの定義 (パケット送信時)	30
4.7.2 オプション比較の定義	30
4.8 アライメントについて	30
4.9IPsec について	31
4.9.1ESP	31
4.9.2AH	31
4.10FAQ	32
4.10.1 フラグメントされたパケットの定義	32
5 テストスクリプト用 Perl ライブラリ	33
5.1 スクリプトのファイル名	33
5.2 初期化	33
5.3 コマンドラインオプション	33
5.4 ステータスコード	34
5.5 ライブラリ関数	34
5.5.1vSend	35
5.5.2vRecv	36
5.5.3vCapture	37
5.5.4vStop	37
5.5.5vClear	38
5.5.6vLog	38
5.5.7vRemote	39
5.5.8vRoundoff	39
5.5.9vErrmsg	39
5.5.10vMAC2Addr	40
5.5.11vSleep()	40
5.6 送受信パケットの各フィールド値参照	41
5.6.1 概要	41
5.6.2 フィールドとキー	41
5.7 時刻の関係	48
5.7.1timeout のみが指定された場合	48
5.7.2seektime を指定した場合	48
5.7.3seektime と timeout を指定した場合	48
5.7.4seektime と count を指定した場合	48
5.7.5timeout と count を指定した場合	48
6 ヘッダフォーマット	49
6.1 表記法について	49
6.1.1 表記ルール	49
6.1.2 省略時の設定値基本方針	49
6.2 データリンク	50
6.2.1 イーサネットヘッダ	50

6.3IPv6	52
6.3.1IPv6 ヘッダ	52
6.4IPv6 拡張ヘッダ	54
6.4.1Hop-by-Hop	54
6.4.2Router Alert Option	56
6.4.3Jumbo Payload Option	57
6.4.4Pad1	58
6.4.5PadN	59
6.4.6Type 0 Routing Header	60
6.4.7Fragment Header	62
6.4.8Destination Option Header	63
6.4.9Tunnel Encapsulation Header	64
6.5IPsec	65
6.5.1Authenticatio Headedr	65
6.5.2Encapsulating Security Payload	66
6.5.3AHAlgorithm	67
6.5.4ESPAlgorithm	67
6.6ICMPv6 (ND) (RFC2461)	68
6.6.1Router Solicitation	68
6.6.2Router Advertisement	69
6.6.3Neighbor Solicitation	70
6.6.4Neighbor Advertisement	71
6.6.5Redirect	72
6.6.6Source Link Layer Address Option	73
6.6.7Tagret Link Layer Address option	74
6.6.8Prefix Information Option	75
6.6.9MTU Option	76
6.6.10Redirected Header (Option)	77
6.7Information Messages	78
6.7.1Multicast Listener Query	78
6.7.2Multicast Listener Report	79
6.7.3Multicast Listener Done	80
6.7.4ICMP Echo Request	81
6.7.5Echo Reply	82
6.8Error Message	83
6.8.1Packet Too Big	83
6.8.2Destination Unreachable	84
6.8.3Time Exceeded Messages	85
6.8.4Parameter Problem Messages	86
6.9IPv4	87
6.9.1IPv4 ヘッダ	87
6.9.2End Of Option List オプション	88
6.9.3No Operation オプション	88
6.9.4Loose Source Route オプション	88
6.9.5Strict Source Route オプション	89
6.9.6Record Route オプション	89
6.9.7Timestamp オプション	89
6.10ARP	90
6.10.1ARP パケット	90
6.10.2RARP パケット	91
6.11ICMPv4	92
6.11.1Destination Unreachable Message	92
6.11.2Time Exceeded Message	92
6.11.3Parameter Probelem Message	93
6.11.4Source Quench Message	94
6.11.5Redirect Message	94

6.11.6Echo Request Message	95
6.11.7Echo Reply Message	96
6.11.8Packet Too Big	97
6.12TCP	98
6.12.1TCP ヘッダ	98
6.12.2End Of Option List オプション	99
6.12.3No Operation オプション	99
6.12.4Maximum Segment Size オプション	99
6.13UDP	100
6.13.1Any UDP	100

1 テスタ概要

1.1 テスト環境

想定するテスト環境は、テスト対象を「複数のインタフェースをもったルータおよびマルチホームホスト」とし、ほかのターゲットは、そのサブセットとして扱う。

テストは、完全に自動で行われ、途中で人手が必要にならないようにする。基本的にテストは、パケットを投げ、その反応をみることでおこなう。しかし、設定の変更、ターゲットから能動的にパケットを出さなければならない場合に、ターゲットコンソールから入出力が必要となるため、シリアルラインからの入出力に対応する。

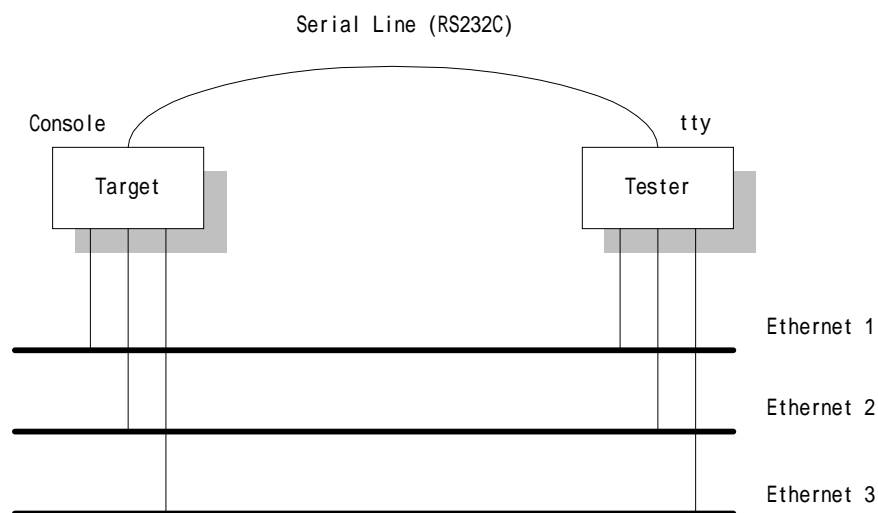


図 1. テスト環境イメージ

1.2 動作環境

1.2.1 ハードウェア環境

つぎのどれかの OS が動作すること

- FreeBSD 2.2.8
- FreeBSD 3.4

1 つ以上のイーサネットワークインタフェースがあること

1 つシリアルインタフェースがあること

1.2.2 ソフトウェア環境

テストの実行に必要なソフトウェアは、次のとおりとする。

表 1. 動作環境 (ソフトウェア)

項目	バージョン等	備考
OS	FreeBSD 2.2.8 or 3.4 にパッチ () をあてたもの	IPv4 Only
OpenSSL Library	0.9.2b	IPsec 用
Perl	5.005_02	シーケンサ用
Perl Lib -		
- Expect	Expect Module	リモート制御用
- IO-Stty	IO-Stty	リモート制御用
- IO-Tty	IO-Tty	リモート制御用

Ether Frame source address を自由に変更できるようにするパッチ。

1.3 インストール

- 1) FreeBSD のインストール
- 2) OS へパッチをあてる
- 3) 関連ライブラリのインストール
- 4) ツールのインストール
- 5) テストスクリプトのインストール
- 6) テスタとターゲットの接続
- 7) config ファイルの設定

1.4 ディレクトリ構成

このツールは '/usr/local/v6eval' (以降 \$V6EVALROOT と表現する) 以下にインストールされ、次のディレクトリ構成となる

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

bin ディレクトリ .

- シーケンサを除き実行に必要なバイナリが入るディレクトリ
- pkt{send,recv,buf,ctl}、リモート制御ファイル、自動実行

ct ディレクトリ .

- コンフォーマンステスト一式が入るディレクトリ
- このディレクトリの下に、大項目ごとにディレクトリを作成し、その中に各テストスクリプトをいれる

doc ディレクトリ .

- README、INSTALL、TODO 等のドキュメントが入るディレクトリ

etc ディレクトリ .

- 各種設定ファイルを置くディレクトリ
- インストール直後はサンプルが入っている

2 設定ファイルフォーマット

2.1 Tester Node Config File

2.1.1 目的

このファイルでは、次にあげるテスト固有の設定を記述する。

- テスタの I/F 名と仮想 I/F 名の対応
- 実行に必要なファイルのパス
- パケット送受信の auto 指定に対応するために必要な情報
- レポートの出力に必要な情報

2.1.2 文法

- 項目は順不同
- 1行1項目。途中での改行は不可
- 同じエントリを2度指定した場合、後で指定したものが優先される
- #で始まる行および空行はコメント

2.1.3 例

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

#
# Remote Control 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 設定項目

このファイル中で設定可能な項目は、次の表のとおりである。

表 2. 設定項目

エン트리名	省略	内容
socketpath	可	<ul style="list-style-type: none">• buffer daemon と送受信プログラムが通信する UNIX Domain Socket を作成するディレクトリを指定する• テストプログラム実行前に、このディレクトリは作成されていること• 省略した場合は、/tmp とみなす
LinkN	不可	<ul style="list-style-type: none">• N の部分には数字がはいる• 必ず "Link0" のエン트리があること• その後に I/F 名、MAC Address を書く• 項目 (I/F name ,MAC Addr,...) の省略は不可
filter	可	<ul style="list-style-type: none">• IPv6 パケットのみをテストの受信パケットとする場合、ここに "IPv6" を指定する。
RemoteDevice	可	<ul style="list-style-type: none">• tip デバイス名• 省略した場合は、cuaa0c とする
RemoteDebug	可	<ul style="list-style-type: none">• 省略した場合は 0 とみなす• 詳細不明
RemoteIntDebug	可	<ul style="list-style-type: none">• 省略した場合は 0 とみなす• 詳細不明
RemoteLog	可	<ul style="list-style-type: none">• 省略した場合は 0 とみなす• 1 を指定すると、通信内容がログに記録される
RemoteSpeed	可	<ul style="list-style-type: none">• 省略した場合は 0 とみなす• 送信文字間隔を秒で指定する• 小数点可
RemoteLogout	可	<ul style="list-style-type: none">•

2.2 Node under Test Config File

2.2.1 目的

このファイルでは、次にあげる**ターゲット固有**の設定を記述する。

- テスタの I/F 名と仮想 I/F 名の対応
- どのテストを実施するかヒントとなるもの
- パケット送受信の auto 指定に対応するために必要な情報
- レポートの出力に必要な情報

2.2.2 考慮すべき点

- テスタ環境が変化してもこのファイルの内容は変化してはいけない
- 1 ターゲットにつき 1 ターゲットファイル
- テストに必要なターゲットの IPv4 アドレスは、固定とし、パケット定義ファイル中に埋め込む

2.2.3 文法

- 項目は順不同
- 1 行 1 項目。途中での改行は不可
- # で始まる行、および空行はコメント

2.2.4 例

```
#
# 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 設定項目

このファイル中で設定可能な項目は、次の表のとおりである。

表 3. 設定項目

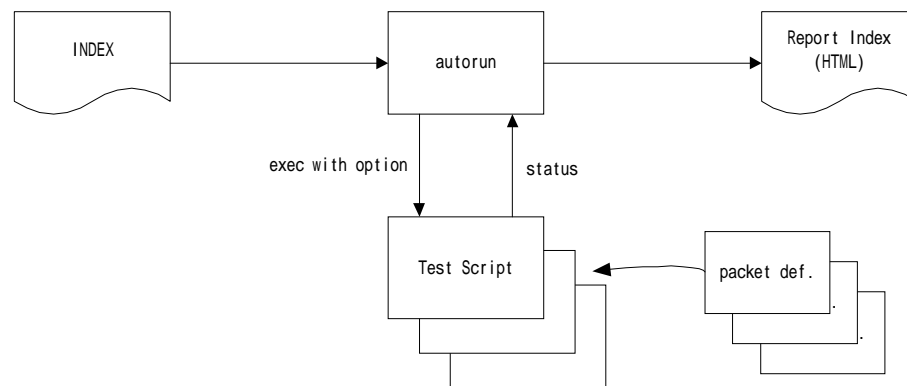
エントリ	省略	内容
System	不可	<ul style="list-style-type: none">システム名を書く。指定できるのは、 kame-freebsd linux-v6これをもとにシリアル制御方法を切り替える
TargetName	不可	<ul style="list-style-type: none">ターゲットのバージョン等識別子を指定
HostName	不可	<ul style="list-style-type: none">ターゲットのホスト名を書くスペースを含んではいけない
Type	不可	<ul style="list-style-type: none">“host” or “router” を指定
User	可	<ul style="list-style-type: none">リモート制御時のユーザ名を指定省略した場合は、root
Password	可	<ul style="list-style-type: none">リモート制御時のパスワードを指定省略した場合は、v6eval
LinkN	不可	<ul style="list-style-type: none">Tester Node Config file の同一項目と同じ

3 コマンドシンタックス

3.1 テスト自動実行プログラム (autorun)

3.1.1 概要

- 必要なテストすべてを実施し、その結果を報告する。
- 呼び出されたテストプログラムは、Target Config File と Tester Config File の中をみて、テストを実施するかどうかを判断する。このプログラムはなにも面倒をみない。



3.1.2 コマンドライン

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

index で指定された INDEX ファイルに記述されたテストを実施し、テスト結果を ./index.html に出力する。ただし、実行時にすでに ./index.html が存在していた場合は、テストを実行しない。

3.1.3 オプション

-t .

INDEX に記述されたパケット定義ファイルのテストのみを行う。テストスクリプトは、実行しない。

-g .

INDEX に記述されたテストスクリプトファイルより、HTML ドキュメントを生成する。テストスクリプトは実行しない。すべてのテストがパスしたものとして、./index.html を出力する。

-t と -g オプションは、同時に指定することができる。

-s num.

num で指定したテスト番号より若い番号のテストは実施しない

-e num.

num で指定したテスト番号より大きい番号のテストは実施しない

-e と -s オプションは、同時に指定することができる

-f.

テスト実行時に index.html が存在していた場合でも、テストを実行し、その結果を、index.html に上書きする

3.1.4 出力

テストスクリプトを実行した場合には、その結果一覧の HTML ファイル (index.html) を出力する。

3.1.5 返り値

0: エラー (プログラムの動作として) なし

1: プログラムの実行になんらかのエラーが発生した

3.1.6 INDEX ファイルのフォーマット

3.1.6.1 フォーマット

- 1行毎に1つのテストに必要な情報を記述する
 - '#' で始まる行、または空行はコメントとみなす
 - 結果一覧中にテストのバージョンを表示させたい場合は、コメント中につきのような行があること
- ```
$Name hoge$
```
- "&print:" ではじまる行は、コロン以降をテスト結果の HTML ドキュメント中に出力する。
  - "&section:" ではじまる行は、.... (未実装)
  - それ以外の行は次のフォーマットに従うこと。

```
<seq>:<def>:<opts>:<htmldoc>:<dsc>:<links>
```

<seq> : テストシーケンスのパス名を記述する。省略は不可。  
<def> : パッケージ定義ファイルのパス名を記述する。省略は不可  
<opts>: テストシーケンスへわたす引数を記述する。省略可能で、省略した場合は、引数無しとみます

<htmldoc>: コンフォーマンステストシーケンスファイルに付属の perldoc から生成せず  
に、独自の HTML ドキュメントを用いる場合は、ここにファイル名を記述する。  
<dsc> : テスト名称を記述する。省略可能で、省略時は、テストシーケンス名から  
".seq" を除いたものとする。  
<links>: そのスクリプトで利用するネットワークデバイスの数を指定する。たとえば、  
Link0、Link1 を使用する場合は 2 を指定する。省略時は、1 とする。

### 3.1.6.2 例

```
$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
```

それぞれのテストプログラムについては、5章テストスクリプト用 Perl ライブラリを参照すること

## 3.2 パケット定義文法チェッカー (pktlint)

パケット定義ファイルに記述された定義に、エラーがないか調べる。Frame\_XX で定義されているパケットを、送信用に生成することでチェックする。

### 3.2.1 オプション

テストシーケンスファイルと同じオプション (XXX)

### 3.2.2 返り値

0 : エラー無し  
0 以外 : エラー有り

## 3.3 リモート制御プログラム

### 3.3.1 概要

検証対象機器をリモートで制御し、テストの自動化を可能とするためのプログラム群。拡張子は、rmt とし、Perl ライブラリ V6evalTool::vRemote() を経由して呼び出される。

### 3.3.2 コマンド

リモート制御コマンドとして、次のものがある

表 4. リモート制御コマンド

| コマンド名              | 動作                         |
|--------------------|----------------------------|
| cleardefr.rmt      | Default Router List をクリアする |
| clearnc.rmt        | Neighbor Cache をクリアする      |
| clearprefix.rmt    | Prefix List をクリアする         |
| clearroute.rmt     | ルーティングテーブルをクリアする           |
| loginout.rmt       | login, logout              |
| manualaddrconf.rmt | マニュアルで I/F にアドレスを設定する      |
| ping6.rmt          | ターゲットから ping を出させる         |
| reboot.rmt         | リブート後、コマンドプロンプトを待つ         |
| reboot_async.rmt   | リブート後、直ちに終了                |
| route.rmt          | ルーティングをセットする               |

### 3.3.3 返り値

0 : エラー無し  
0 以外 : エラー有り

## 4 パケットの定義

### 4.1 概略

#### 4.1.1 定義の記述法

##### 4.1.1.1 パケットの定義の記述法

すべての定義は、次のフォーマットで記述する。

```
タイプ名 タグ名 {
 要素名 = 値;
 要素名 = 値;
 ...
}
```

ヘッダ名により、その中に記述できる要素名が決まる。また、タイプ名と要素名により、取りうる値の型が決まる。

- 順序が意味を持たない要素名について、複数回現れた場合、後優先とする。
- 直接値を持つ要素名は、先大文字
- 参照する要素名は、すべて小文字 ( header, upper, payload,...)

##### 4.1.1.2 送信、受信パケット定義の違い

- 基本的に送受信で同一の表記法とする。
- 送信パケットで定義できるのは ( = ) のみ。
- 受信では送信パケット定義に加え、大小比較、選択定義が記述できる。

## 4.2 型

定義できる値の型は次の10個に分類される

- 参照型
- オプション型
- データ型
- crypt 型
- auth 型
- ether 型
- IPv6 Addr 型
- IPv4 Addr 型
- payload 型
- uint 型

### 4.2.1 参照型

- パケットの構造を定義するための型
- 他の場所で定義されたタグを指定する
- 定義ファイル中で後方参照可能
- オペレーションは次の表が可能

表 5. 参照型のオペレーション

| 表記記号 | 動作 (送信) | 動作 (受信) | 備考 |
|------|---------|---------|----|
| =ref |         |         |    |

#### 4.2.2 オプション型

- ヘッダ中のオプションを定義するためにある型
- 他の場所で定義されたタグ（オプション）を指定する
- 定義ファイル中で後方参照可能
- オペレーションは次の表が可能

表 6. オプション型のオペレーション

| 表記記号                     | 動作<br>(送信) | 動作<br>(受信)     | 備考 |
|--------------------------|------------|----------------|----|
| =ref                     |            |                |    |
| =oneof(ref,ref[,ref]...) | 指定不可       | or             |    |
| =comb(ref,ref[,ref]...)  | 指定不可       | 順不同ですべてを含む     |    |
| =stop                    | 指定不可       | 以降のオプションを比較しない |    |

### 4.2.3 データ型

- バイト列として定義する要素のためにある型
- ペイロードデータなどフォーマットに関係なくバイト列を記述するとき利用
- 比較は完全一致（送信、受信ともに、イコールのみ指定可能。=any は不可）
- 1 バイトデータのみ指定可能。256 以上の値を指定した場合、エラー。
- 繰り返し定義は repeat() をつかう
- オペレーションは次の表が可能

表 7. データ型のオペレーション

| 表記記号                    | 動作<br>(送信) | 動作<br>(受信) | 備考                                                                                                    |
|-------------------------|------------|------------|-------------------------------------------------------------------------------------------------------|
| =val                    |            |            | 1 バイトのデータ。<br>val は 0-255 の範囲                                                                         |
| ={val[,val...]}         |            |            | 1 バイトのデータ × n 個。<br>それぞれの val は 0-255 の範囲                                                             |
| =file('file-path')      |            |            | ファイルから読み込んだバイト列<br>ファイル名は ' で囲むこと<br>(未実装)                                                            |
| =repeat(val,times)      |            |            | val を times 回繰り返したバイト列                                                                                |
| =substr(ref,offset,len) |            | 1          | 他の場所でパケット定義された ref の<br>offset から len バイト切り出したデータ<br>列                                                |
| =left(ref,len)          |            | 1          | 他の場所でパケット定義された ref の先<br>頭から len バイト切り出したデータ列<br>=substr(ref,0,len) と等価                               |
| =right(ref,offset)      |            | 1          | 他の場所でパケット定義された ref の<br>offset バイト目から最後までを切り出し<br>たデータ列<br>=substr(ref,offset,sizeof(ref)-offset) と等価 |
| =patch(ref,offset,val)  |            | 1          | 他の場所でパケット定義された ref の<br>offset バイト目の値を val に書き換えた<br>データ列<br>val は、0-255 の範囲                          |

1：受信時であっても、ref で参照されるパケット定義は、送信時とみなされて処理される。auto でアドレス等を埋めている場合は、注意すること

## 4.2.4 crypt 型

- IPsec ESP のための型
- 暗号化、復号化の情報を記述する
- 受信時では、ICV の値は比較対象外とし、受信パケット中に含まれている ICV をもとに、暗号の復号化を行う

表 8. crypt 型のオペレーション

| 表記記号                           | 動作<br>(送信) | 動作<br>(受信) | 備考 |
|--------------------------------|------------|------------|----|
| =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 暗号アルゴリズム

- 暗号化しない
- アライメント長を指定する

### 4.2.4.2 descbc アルゴリズム

- RFC2405 にしたがって暗号化、復号化をおこなう
- 鍵、初期ベクタ、パディング長を指定する

### 4.2.4.3 blowfish アルゴリズム

- RFC2451 にしたがって暗号化、復号化をおこなう
- 鍵、初期ベクタ、パディング長を指定する

### 4.2.4.4 rc5 暗号アルゴリズム

- RFC2451 にしたがって暗号化、復号化をおこなう
- 鍵、初期ベクタ、パディング長を指定する

### 4.2.4.5 cast128 暗号アルゴリズム

- RFC2451 にしたがって暗号化、復号化をおこなう
- 鍵、初期ベクタ、パディング長を指定する

### 4.2.4.6 des3cbc 暗号アルゴリズム

- RFC2451 にしたがって暗号化、復号化をおこなう
- 鍵、初期ベクタ、パディング長を指定する

## 4.2.5 auth 型

- IPsec ESP & AH のための型
- ハッシュ値の計算アルゴリズム情報を記述する

表 9. auth 型のオペレーション

| 表記記号                  | 動作<br>(送信) | 動作(受信) | 備考 |
|-----------------------|------------|--------|----|
| =null_auth(alignment) |            |        |    |
| =hmacmd5(key)         |            |        |    |
| =hmacsha1(key)        |            |        |    |

### 4.2.5.1 null\_auth

- アライメントのサイズを指定する

### 4.2.5.2 hmacmd5 認証アルゴリズム

- RFC2403 にしたがって認証値を計算する
- 鍵を指定する

### 4.2.5.3 hmacsha1 認証アルゴリズム

- RFC2404 にしたがって認証値を計算する
- 鍵を指定する

## 4.2.6 esppad 型

- ESP のパディング領域のデータ内容生成ルールを指定するための型

表 10. auth 型のオペレーション

| 表記記号          | 動作<br>(送信) | 動作(受信) | 備考 |
|---------------|------------|--------|----|
| =sequential() |            | 比較対象外  |    |
| =allzero()    |            | 比較対象外  |    |
| =random()     |            | 比較対象外  |    |

受信パケット定義に、パディング領域の指定があったとしても、その領域は比較されない。

### 4.2.6.1 sequential()

パディングデータ領域を先頭から順に、「0、1、2、3、…」と順に値を埋める

### 4.2.6.2 allzero()

パディングデータ領域をすべて0で埋める

### 4.2.6.3 random()

パディングデータ領域を乱数で埋める

## 4.2.7 ether 型

- Ethernet MAC アドレス表記のための型

表 11. ether 型のオペレーション

| 表記記号                     | 動作<br>(送信) | 動作<br>(受信) | 備考         |
|--------------------------|------------|------------|------------|
| =ether(MAC Addr 表記)      |            |            |            |
| =v62ehtermulti(v6addr 型) |            |            |            |
| =tnether(ifname)         |            |            |            |
| =ethersrc(ifname)        |            |            |            |
| =etherdst(ifname)        |            |            |            |
| =nutether(ifname)        |            |            |            |
| =auto                    |            |            | 定義場所により異なる |
| =any                     | 指定不可       |            |            |

## 4.2.8 IPv6 Addr 型

- IPv6 アドレス表記のための型

表 12. IPv6 Addr 型のオペレーション

| 表記記号                         | 動作<br>(送信) | 動作<br>(受信) | 備考         |
|------------------------------|------------|------------|------------|
| =v6(v6 addr 表記)              |            |            |            |
| =tnv6(ifname)                |            |            |            |
| =nutv6(ifname)               |            |            |            |
| =v6src(ifname)               |            |            |            |
| =v6dst(ifname)               |            |            |            |
| =v6merge(prefix,len,v6 addr) |            |            |            |
| =v6ether(MAC);               |            |            |            |
| =auto                        |            |            | 定義場所により異なる |
| =any                         | 指定不可       |            |            |

## 4.2.9 IPv4 Addr 型

- IPv4 アドレス表記のための型

表 13. IPv4 Addr 型のオペレーション

| 表記記号            | 動作<br>(送信) | 動作<br>(受信) | 備考         |
|-----------------|------------|------------|------------|
| =v4(v4 addr 表記) |            |            |            |
| =auto           |            |            | 定義場所により異なる |
| =any            | 指定不可       |            |            |

## 4.2.10 payload 型

- パケットの payload 部を参照する型
- 他の場所で定義されたタグを指定する
- 定義ファイル中で後方参照可能
- オペレーションは次の表が可能

表 14. payload のオペレーション

| 表記記号 | 動作 (送信) | 動作 (受信) | 備考 |
|------|---------|---------|----|
| =ref |         |         |    |
| =any | 指定不可    | 比較しない   |    |

#### 4.2.11 uint 型

- そのほか、値を持つ要素用

表 15. uint 型のオペレーション

| 表記記号                     | 動作<br>(送信) | 動作<br>(受信) | 備考               |
|--------------------------|------------|------------|------------------|
| =val                     |            |            |                  |
| =within(val1,val2)       | 指定不可       |            | val1<member<val2 |
| =oneof(val,val[,val]...) | 指定不可       |            |                  |
| >val                     | 指定不可       |            |                  |
| <val                     | 指定不可       |            |                  |
| >=val                    | 指定不可       |            |                  |
| <=val                    | 指定不可       |            |                  |
| =any                     | 指定不可       |            |                  |
| =auto                    |            |            |                  |

## 4.3 アドレス変換関数について

パケット定義ファイル、テストシーケンスファイルから、テスター、ターゲット依存の定義を取り除くため、テスト条件定義ファイルより、値を生成する次の関数がパケット定義中で利用できる。

### 4.3.1 Ether アドレス

#### 4.3.1.1 ether()

書式 .

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

- MAC アドレスを引数にとり、Ether 型の値を返す
- 引数の省略不可

例 .

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

#### 4.3.1.2 tnether()

書式 .

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

- インターフェース名を引数にとる
- 引数で指定されたインタフェースのテスト MAC アドレスに対応した Ether 型の値を返す
- 引数を省略した場合は、pktsend、pktrecv の -i option で指定されたインタフェース名を指定したものとして処理される。

例 .

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

#### 4.3.1.3 nutether()

書式 .

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

- インターフェース名を引数にとる
- 引数で指定されたインタフェースのターゲット MAC アドレスに対応した Ether 型の値を返す
- 引数を省略した場合は、pktsend、ktrecv の -i option で指定されたインタフェース名を指定したものとして処理される。

例 .

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

#### 4.3.1.4 ethersrc()

書式 .

```
ethersrc(["ifname"])
```

- インターフェース名を引数にとる
- 送信時は、引数で指定されたインタフェースのテスト MAC アドレスに対応した Ether 型の値を返す
- 受信時は、引数で指定されたインタフェースのターゲット MAC アドレスに対応した Ether 型の値を返す
- 引数を省略した場合は、pktsend、pktrecv の -i option で指定されたインタフェース名を指定したものと処理される。

例 .

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

#### 4.3.1.5 etherdst()

書式 .

```
etherdst(["ifname"])
```

- インターフェース名を引数にとる
- 送信時は、引数で指定されたインタフェースのターゲット MAC アドレスに対応した Ether 型の値を返す
- 受信時は、引数で指定されたインタフェースのテスト MAC アドレスに対応した Ether 型の値を返す
- 引数を省略した場合は、pktsend、pktrecv の -i option で指定されたインタフェース名を指定したものと処理される。

例 .

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

#### 4.3.1.6 v6ethermulti()

書式 .

```
v6ethermulti(IPv6 型)
```

- IPv6 型の値、すなわち IPv6 型を返す関数を引数にとる
- 引数で指定された IPv6 Multicast アドレスに対応した、Ethernet Multicast アドレスの Ether 型の値を返す
- 引数の省略は不可

例 .

```
v6ethermulti(v6("ff02::1"));
```

## 4.3.2 IPv6 アドレス

### 4.3.2.1 v6()

書式 .

```
v6(“RFC2373 表記 ”)
```

- RFC2373 で記述された IPv6 アドレスを引数にとる
- 引数に対応した IPv6 型を返す

例 .

```
v6(“ff02::1”);
```

### 4.3.2.2 tnv6()

書式 .

```
tnv6([“ifname”])
```

- インタフェース名を引数にとる
- 引き数に指定したインタフェース名から生成されるテストの IPv6 link local アドレスを返す
- 省略時は、コマンドラインオプション `-i option` で指示されたインタフェース名が指定されたものとして処理する

例 .

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

### 4.3.2.3 nutv6()

書式 .

```
nutv6([“ifname”])
```

- インタフェース名を引数にとる
- 引き数に指定したインタフェース名から生成されるターゲットの IPv6 link local アドレスを返す
- 省略時は、コマンドラインオプション `-i option` で指示されたインタフェース名が指定されたものとして処理する
- 省略時は、`-i option`

例 .

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

#### 4.3.2.4 v6src()

書式 .

```
v6src(["ifname"])
```

- インタフェース名を引数にとる
- 送信時は、引き数に指定したインタフェース名から生成されるテストの IPv6 link local アドレスを返す
- 受信時は、引き数に指定したインタフェース名から生成されるターゲットの IPv6 link local アドレスを返す
- 省略時は、コマンドラインオプション `-i option` で指示されたインタフェース名が指定されたものとして処理する

例 .

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

#### 4.3.2.5 v6dst()

書式 .

```
v6dst(["ifname"])
```

- インタフェース名を引数にとる
- 送信時は、引き数に指定したインタフェース名から生成されるターゲットの IPv6 link local アドレスを返す
- 受信時は、引き数に指定したインタフェース名から生成されるテストの IPv6 link local アドレスを返す
- 省略時は、`-i option`

例 .

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

#### 4.3.2.6 v6merge()

IPv6 アドレスのプレフィックスを変化させる

書式 .

```
v6merge("RFC2372",prefixlength,v6 アドレス型)
```

例 .

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

#### 4.3.2.7 v6ether()

Ether アドレスより IPv6 Link Local Address を生成する

書式 .

```
v6ether("MAC Address")
```

例 .

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

### 4.3.3 IPv4 アドレス

#### 4.3.3.1 v4()

書式 .

```
v4("IPv4 アドレス表記")
```

- 引数に IPv4 アドレスをとる
- IPv4 型を返す
- 引数の省略は不可

例 .

```
v4("127.0.0.1");
```

## 4.4 コメントの記述

コメントの記述法は、C++ 形式とする。

例 .

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

## 4.5 パケット構造の定義

### 4.5.1 概要

送受信するパケットは、ヘッダ、ペイロード等の部分部分を定義し、それを構造で意義で、送受信可能なパケットとして生成する。これらは、つぎにあげるタイプの定義で記述する。

- frame
- packet
- payload

### 4.5.2 Frame\_Ether 定義

表 16. Frame\_Ether 定義

| 要素     | 型   |                             |
|--------|-----|-----------------------------|
| header | ref | Hdr_Ether 定義を参照             |
| packet | ref | Packet_XX または、Payload 定義を参照 |

- それぞれ 1 度のみ出現
- 順序は意味を持たない
- 省略不可
- この定義のみ送受信パケットとして指定可能

### 4.5.3 Packet\_XX 定義

Packet 定義には、つぎのものがある

- Pakekt\_IPv6
- Packet\_IPv4
- Packet\_ARP
- Packet\_RARP

これらの定義は、次のルールに従う

表 17. Packet\_XX 定義

| 要素     | 型   |                                                                    |
|--------|-----|--------------------------------------------------------------------|
| header | ref |                                                                    |
| exthdr | ref | Packet_IPv6、Packet_IPv4 のみ<br>省略可能<br>拡張ヘッダを参照                     |
| upper  | ref | packet, upper, payload 定義を参照<br>Pakekt_IPv6、Packet_IPv4 のみ<br>省略不可 |

- upper は必ず 1 度のみ出現。(現状は、省略不可)
- exthdr は複数指定可能で、その指定順にヘッダが構成される
- upper に packet 定義を参照するとトンネルされたパケットが生成できる
- upper 定義とは、TCP、UDP、ICMP、Packet\_XX
- 最初の header は Hdr\_IPv4、Hdr\_IPv6 定義のみ参照可
- その後の header は Hdr\_IPv4、Hdr\_IPv6 定義以外が参照可

### 4.5.4 Payload 定義

表 18. Payload 定義

| 要素   | 型    |  |
|------|------|--|
| data | data |  |

- data は最低 1 つ必要
- data は複数指定可能
- ここで定義されたデータは、バイト列として比較される。

## 4.6 その他のタイプ定義

6章、ヘッダフォーマット一覧を参照。ここにあげている単語の先頭文字を大文字に接続したものが要素名、タイプ名となる

## 4.7 オプションの定義と比較

### 4.7.1 オプションの定義（パケット送信時）

ヘッダ定義中で、  
option = タグ名 ;  
のように書く。

### 4.7.2 オプション比較の定義

#### 4.7.2.1 OR

指定するオプションのどれか1つがくればよい場合は、  
option = oneof(A,B,C);  
と記述する。

#### 4.7.2.2 順不同

順序は問わないが、指定のオプションがきてほしい場合は、  
option = comb(A,B,C);

で、「つぎの3つのオプション中で、A、B、Cを**順不同**で含む」の意味をもつ。  
A、B、Cは、タグ名とする。入れ子で定義できない。

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

という指定も可能。この場合は、Aが2回、1が1回出現する。

#### 4.7.2.3 不定

オプションについては、比較の必要ない場合や、指定のオプション以降を比較する必要がない場合は、

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

と記述する。

## 4.8 アライメントについて

オプションヘッダ内の個々のオプションは、それぞれが保持するデータにあわせ、natural Boundry 境界にアライメントされなければならない。これらのため Pad1/PadN オプションがある。このツールでは、自動的にパディングはしない。

ただし、ユーザへの警告として、あるオプションヘッダの終了点が8バイト境界に一致していない場合のみそのむね表示する。ユーザはこの情報をもとにパディングすること。

受信時は、Pad1/PadN は飛ばして処理する。受信パケットの定義中にこれらのオプションがあっても無視する。

## 4.9 IPsec について

### 4.9.1 ESP

#### 4.9.1.1 対応する暗号、復号化関数とハッシュ関数

対応する暗号、復号化関数は、次のものとする

- DES in CBC mode
- NULL Encryption

対応するハッシュ関数は、AH のものと同一とする。

### 4.9.2 AH

#### 4.9.2.1 対応するハッシュ関数

対応するハッシュ関数は、次のものとする

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

#### 4.9.2.2 IPsec とフラグメント

正しいパケットである

- IP Fragment ESP ...
- IP Fragment AH ...

のようなパケットは生成は、通常のフラグメントパケットと同様に、`subster()` 等を利用して作成する。

不正なパケット構成である

- IP AH Fragment

の場合は、フラグメントヘッダ以降を不変領域として計算する。

- IP AH AH xxxx

の構成は、エラーにはならず、不正な値が入る。

## 4.10 FAQ

### 4.10.1 フラグメントされたパケットの定義

あるパケットを指定して、「MTU = nnn でフラグメントしたパケットをつくれ」という指定方法は用意されていない。

別途定義してあるパケットの一部を切り出して、payload のように処理する。

```
// フラグメントされる前の大きなパケットを定義
Packet_IPv6 ip6_ping {
 header=ipv6;
 upper=echo_req;
}

Hdr_IPv6 ipv6 {
}

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

// 送信パケット
Packet_IPv6 frag1 {
 header=ipv6;
 exthdr=frag_hdr;
 upper=fragmented;
};
```

フラグメントヘッダのオフセットは、定義を書く人が埋めること。

## 5 テストスクリプト用 Perl ライブラリ

ここでは、パケット生成、受け取り、パケットメモリコントロールの perl API について記述する。

### 5.1 スクリプトのファイル名

仕様適合テスト自動化のため、作成するスクリプトのファイル名は、次のルールにしたがうこと。

- 拡張子は、”.seq”
- RFC、ドラフト単位の分類は、ディレクトリでおこなう。よって、ファイル名には、含めてはいけない

### 5.2 初期化

このライブラリを利用するユーザーは、テストシーケンススクリプトの頭に、次の文を記述しなければならない。

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

初期化の途中で、エラーが起きた場合、終了コードは内部エラー (64) を返す。

### 5.3 コマンドラインオプション

このライブラリを利用した場合、つぎのコマンドラインオプションが指定できる。

```
-tn Testing node 設定ファイル
-nut Node under test 設定ファイル
-pkt パケット定義ファイル
-log 出力するログのファイル名
-v log に出力した内容を画面上にも表示する
-l Log Level (0,1,2...)
-h Help

-nostd 標準インクルードファイルを読み込まない
-remote リモートオプションを指定する
-noremote リモート制御を行わない()
-keep1md CPP で処理した後の中間ファイルを削除しない
-trace 詳細なトレース情報を標準出力に出力する
```

すべて成功したものとして vRemote() のステータスをかえず (この仕様でいいの?)

省略された場合は、つぎの値が設定されたものとして動作する

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

ハイフン ( - ) 以外で始まる文字列がコマンドラインに指定された場合、ユーザーはその文字列を @ARGV から参照できる。ハイフンで始まるオプションで、上記以外のオプションが指定された場合、エラー終了する。終了コードは内部エラー。

## 5.4 ステータスコード

作成したテストスクリプトは、そのテストの結果を示す次の値をステータスコードとして返すこと

表 19. ステータスコード

| コード | 定数               | 意味                      |
|-----|------------------|-------------------------|
| 0   | \$exitPass       | PASS                    |
| 1   | \$exitIgnore     | テストの準備等のスクリプトで正常終了した    |
| 2   | \$exitNS         | Not yet supported       |
| 3   | \$exitWarn       | 仕様が不明確なので FAIL とは断定できない |
| 4   | \$exitHostOnly   | ホスト専用テストをルータに対して実行した    |
| 5   | \$exitRouterOnly | ルータ専用テストをホストに対して実行した    |
| 32  | \$exitFail       | FAIL                    |
| 64  | \$exitFatal      | Internal Error          |

自動実行プログラム (3.1 章参照) は、\$exitFatal を受けた場合、それ以降の手順を停止し、それまでに実行したテストのレポートを出力する。

## 5.5 ライブラリ関数

ユーザには、次の関数を提供する。

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

## 5.5.1 vSend

### 5.5.1.1 概要

指定したインタフェースから、指定したパケット（複数指定可）を送信する。

### 5.5.1.2 引数

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

### 5.5.1.3 返り値

ハッシュ値をかえす。キーと値は、次のとおり。

sentTimeN

N 番目に指定したパケットを送信した時刻。たとえば、最初に指定したパケットの送信時刻が知りたい場合、\$retval{'sentTime1'} とする。そのフォーマットは、文字列で次のように格納されている。

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

ただし、この処理中にエラーが生じた場合は、テストスクリプトを終了し \$exitFatal(64) を終了ステータスとして返す

そのほか、

5.6 章を参照のこと

## 5.5.2 vRecv

### 5.5.2.1 概要

指定したインタフェースから、期待するパケットが受信できたかどうかを調べる。

### 5.5.2.2 引数

```
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 終了条件

このルーチンからもどる場合には、次の3通りの場合がある

- (1) 指定したパケットを受信した
- (2) タイムアウトで指定した時間がたった
- (3) カウントで指定した個数のパケットを受信した

これらのうち、どれかひとつの条件が満たされた場合、このルーチンから復帰する。

### 5.5.2.4 返り値

ハッシュ値を返す。そのキーと値は、次のとおり。

status:

```
=0: 正常終了
1: タイムアウトによる終了
2: 指定のパケットを受信しても期待するパケットが受信できなかった。
>=3: エラー
```

recvCount;

期待するパケットを受信するまでに受信したパケット数。(期待したパケットもその数に含む)

recvTimeN:

N 番目に受信したパケットの受信時刻。たとえば、最初に受信したパケットの受信時刻を知りたい場合、\$retval{"recvTime1"} とする。そのフォーマットは、文字列で次のように格納されている。

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

recvFrame:

正常終了の場合、期待値として指定したパケット名のうち、どのパケットを受信したかの情報として、パケット名が入る。エラー（タイムアウト等を含む）の場合、このハッシュ値は、未定義となる。

そのほか .

5.6 章を参照のこと

## 5.5.3 vCapture

### 5.5.3.1 概要

指定したインタフェースのパケットメモリへ受信パケットの格納を開始する。

### 5.5.3.2 引数

```
sub vCapture($)
```

リンク名を引数に取る

### 5.5.3.3 返り値

スカラー値、0 を返す

すでにキャプチャモードだった場合も正常終了

ただし、この処理中にエラーが生じた場合は、テストスクリプトを終了し \$exitFatal(64) を終了ステータスとして返す。

## 5.5.4 vStop

### 5.5.4.1 概要

指定したインタフェースのパケットメモリへのキャプチャを停止する。

### 5.5.4.2 引数

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

### 5.5.4.3 返り値

スカラー値、0 を返す

すでに停止していた場合も正常終了

ただし、この処理中にエラーが生じた場合は、テストスクリプトを終了し \$exitFatal(64) を終了ステータスとして返す

## 5.5.5 vClear

### 5.5.5.1 概要

指定したインタフェースの、パケットメモリのバッファを空にする

### 5.5.5.2 引数

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

### 5.5.5.3 返り値

スカラー値、0 を返す

すでにバッファが空でも正常終了 (pktctl の返り値も同じ仕様)

ただし、この処理中にエラーが生じた場合は、テストスクリプトを終了し \$exitFatal(64) を終了ステータスとして返す

## 5.5.6 vLog

### 5.5.6.1 概要

指定した文字列を log に書き出す。

### 5.5.6.2 引数

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

## 5.5.7 vRemote

### 5.5.7.1 概要

リモート制御プログラムを呼び出す

### 5.5.7.2 引数

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

### 5.5.7.3 返回值

0 : エラーなし

0 以外 : エラーあり

## 5.5.8 vRoundoff

### 5.5.8.1 概要

1/10 の桁に四捨五入する

### 5.5.8.2 引数

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

### 5.5.8.3 返回值

四捨五入した値

## 5.5.9 vErrMsg

vRecv の返回值を与えると、対応したエラーメッセージをかえす。

### 5.5.9.1 引数

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

### 5.5.9.2 返り値

表 20.

| \$stat{status} の値 | メッセージ                   |
|-------------------|-------------------------|
| 0                 | NULL                    |
| 1                 | “Time-out :”+ 1         |
| 2                 | “Count-out”+ 1          |
| 3 以上              | “ERROR: Internal error” |

ただし、 1 は、\$stat{status} が 0 の場合は、”Could not get any packet”、それ以外の場合は、”Got unexpected packet”。

### 5.5.10 vMAC2Addr

MAC アドレスからリンクローカルアドレスを生成する

#### 5.5.10.1 引数

sub vSleep(\$)

#### 5.5.10.2 返り値

リンクローカルアドレス

### 5.5.11 vSleep()

sleep の代わりに利用し、共通の Log を出力する。

## 5.6 送受信パケットの各フィールド値参照

### 5.6.1 概要

vRecv()、vSend() の戻り値には、送受信したパケットの各フィールドの値を参照するための変数がある。戻り値として得られるハッシュ値に、次のようなフィールドを特定するキーを与えることで、その値を読み出すことができる。

受信したパケットの IPv6 ヘッダ・ソースアドレスを参照する例

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

### 5.6.2 フィールドとキー

フィールドとハッシュのキーとの対応は、次のルールに従う

#### 5.6.2.1 基本ルール

- 参照できるフィールド値は、送信パケット、および期待値と一致したパケット中のフィールド値とする。
- 構造をあらわすブロック名を "." (ピリオド) で接続したものをキーとする

例 .

受信パケットの構造

```
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
```

## 代入される変数とその値

```
$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"
```

## 例 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
```

## 代入される変数とその値

```
$status{"Frame_Ether.Hdr_Ether.Type"}="Hdr_Ether Packet_IPv6"
$status{"Frame_Ether.Hdr_Ether.DestinationAddress"}="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 Identifyre ..."
$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"
```

key が代わるので問題無し

### 5.6.2.2 同一ヘッダ中に同じ名前のフィールドがある場合

- ルーティングヘッダを想定
- 最後に数字をつける

例 .

```
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
```

代入される変数とその値

```
...
$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 同一ネストレベルに同じ名前のヘッダ等がある場合

- 同じオプションヘッダがつく場合を想定
- 構造名に数字を加える

例 .

```
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
```

## 代入される変数とその値

```
$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 時刻の関係

ここでは、受信時で `timeout`、`seektime`、`count` 等の時刻に関するパラメータの相互関係について記述する

### 5.7.1 `timeout` のみが指定された場合

`vRecv()` により `pktrecv` が fork され、引数の評価を行う時刻を  $t_1$  とする。 $t_1$  に `-e` オプションで指定した時間を加えた時刻が `timeout` 時刻となる。 $t_1$  の計算は、パケット定義の評価の前におこなわれる。

時刻  $t_2$  までにバッファにパケットが受信されれば、そのパケットを返し、受信されなければ、`timeout` 終了する。時間の精度は秒。

### 5.7.2 `seektime` を指定した場合

`seektime` により指定した時刻  $t_0$  より前に受信されたデータは、すべて読み捨てる。その後、通常の `recv` 動作に入る。

### 5.7.3 `seektime` と `timeout` を指定した場合

`timeout` 時刻  $t_2$  は、`seektime`  $t_0$  に `timeout`  $x$  を加えた時刻とする。

### 5.7.4 `seektime` と `count` を指定した場合

どちらか早い方の終了条件に達した時点で終了する。ただし `seektime` より前に受信したパケットは、読み捨て、`count` しない。

### 5.7.5 `timeout` と `count` を指定した場合

どちらか早い方の終了条件に達した時点で終了する。

## 6 ヘッダフォーマット

### 6.1 表記法について

#### 6.1.1 表記ルール

表 21. 表記ルール

| フィールド        | 記号        | 意味                                 |
|--------------|-----------|------------------------------------|
| 種類           |           | パケット中には現れないが、生成時に必要となるパラメータ        |
| ヘッダ名         | (option)  | この定義単独ではパケットを構成できない                |
| 種類           | (opt)     | 規格を満たして、追加可能なオプション                 |
| 種類           | (bad-opt) | 追加可能だが追加した場合は、不正なパケットとなるオプション      |
| default      | auto      | 自動的に計算し値を生める                       |
| default (受信) | check     | 実際指定する場合は、auto。<br>正しい値かどうかをチェックする |
| default      | 無印        | 省略不可                               |

#### 6.1.2 省略時の設定値基本方針

- Flag のメンバは、0
- reserve 領域のメンバは、0
- NextHeader は、auto
- 長さに関するフィールドは、auto
- チェックサムに関するフィールドは、auto

## 6.2 データリンク

### 6.2.1 イーサネットヘッダ

#### 6.2.1.1 参照 RFC

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

#### 6.2.1.2 フォーマット

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

#### 6.2.1.3 デフォルト

表 22. イーサネットヘッダ

| 名前                  | 種類            | size  | default (送信)   | default (受信)   |
|---------------------|---------------|-------|----------------|----------------|
| 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 送信時自動設定

Type の自動生成ルール .

このヘッダの次に続くペイロードにより次の値を代入する

表 23. イーサネットヘッダ Type の自動生成ルール

| 後続ヘッダタイプ    | Type の値 | 備考        |
|-------------|---------|-----------|
| IPv4 Header | 0x0800  |           |
| ARP         | 0x0806  |           |
| RARP        | 0x8035  |           |
| IPv6 Header | 0x86dd  | [V6ETHER] |

### 6.2.1.5 その他

定義場所の制限 .

frame 定義の先頭でのみ指定可能

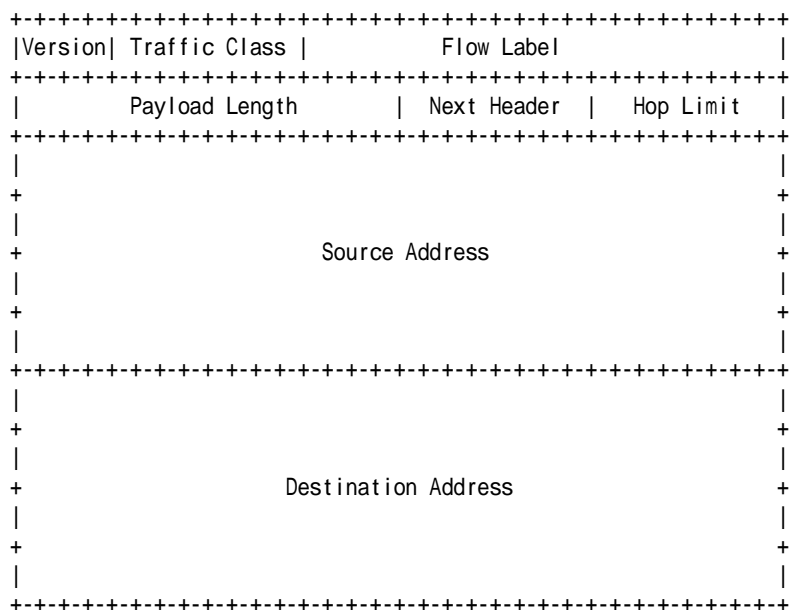
## 6.3 IPv6

### 6.3.1 IPv6 ヘッダ

#### 6.3.1.1 参照 RFC

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

#### 6.3.1.2 フォーマット



#### 6.3.1.3 アライメント

8N

#### 6.3.1.4 デフォルト

表 24. IPv6 ヘッダ

| 名前                  | 種類           | size   | default (送信)      | default (受信)      |
|---------------------|--------------|--------|-------------------|-------------------|
| Version             | uint         | 4bit   | 6                 | 6                 |
| Traffic Class       | uint         | 8bit   | 0                 | any               |
| Flow Level          | 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 送信時自動設定

Payload Length の自動生成ルール .

- IPv6 ヘッダの後ろに続くペイロードの長さ ( RFC2460 pp.4)
- 単位は octet

Next Header の自動生成ルール .

あとに続くヘッダ、上位層により次の値を入れる (IANA protocol-numbers を参照 : rfc/iana/assignments/protocol-numbers)

表 25. Next Header 生成ルール

| 後続ヘッダ                        | 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 その他

packet 定義の先頭でのみ指定可能



Header Ext Length.

受信した値を元に構造を解析する。

## 6.4.2 Router Alert Option

### 6.4.2.1 参照 RFC

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

### 6.4.2.2 フォーマット

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

### 6.4.2.3 デフォルト

表 27. Router Alert (Option)

| 名前              | 種類   | size  | default (送信) | default (受信) |
|-----------------|------|-------|--------------|--------------|
| Option Type     | uint | 8bit  | 0x05         | 0x05         |
| Opt Data Length | uint | 8bit  | 2            | 2            |
| Value           | uint | 16bit | 省略不可         | 省略不可         |

## 6.4.3 Jumbo Payload Option

### 6.4.3.1 参照 RFC

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

### 6.4.3.2 フォーマット

```

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

### 6.4.3.3 デフォルト

表 28. Jumbo Payload (Option)

| 名前                   | 種類   | size | default (送信) | default (受信) |
|----------------------|------|------|--------------|--------------|
| Type                 | uint | 8    | 0xC2         | 0xC2         |
| Opt Data Length      | uint | 8    | 4            | 4            |
| Jumbo Payload Length | uint | 32   | auto         | any          |

### 6.4.3.4 送信時自動設定

Jumbo Payload Length.

- IPv6 ヘッダを除いたパケット長
- 単位は octet

## 6.4.4 Pad1

### 6.4.4.1 参照 RFC

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

### 6.4.4.2 フォーマット

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

### 6.4.4.3 デフォルト

表 29. Pad1 (Option)

| 名前          | 種類   | size | default (送信) | default (受信) |
|-------------|------|------|--------------|--------------|
| Option Type | uint | 8bit | 0            | 0            |

## 6.4.5 PadN

### 6.4.5.1 参照 RFC

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

### 6.4.5.2 フォーマット

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

### 6.4.5.3 デフォルト

表 30. PadN (Option)

| 名前              | 種類   | size | default (送信) | default (受信) |
|-----------------|------|------|--------------|--------------|
| Option Type     | uint | 8bit | 1            | 1            |
| Opt Data Length | uint | 8bit | auto         | any          |
| Pad             | data |      | 0(?)         | 0(?)         |

### 6.4.5.4 送信時自動設定

Opt Data Length の計算方法 .

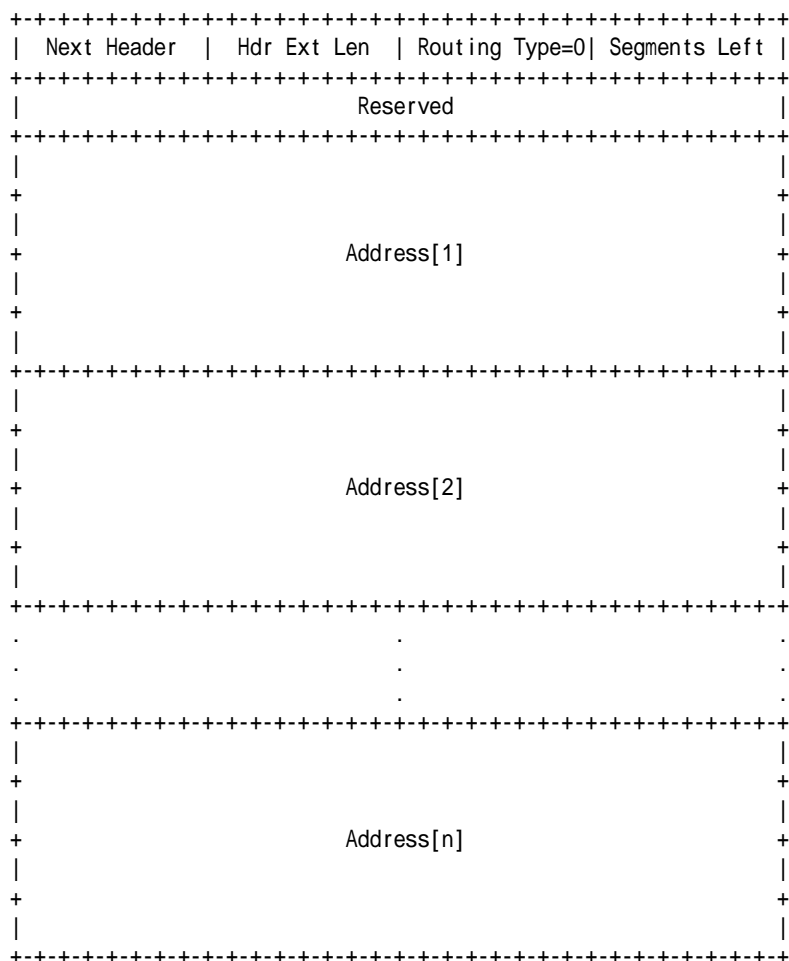
- Pad の長さ
- ゼロも可能

## 6.4.6 Type 0 Routing Header

### 6.4.6.1 参照 RFC

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

### 6.4.6.2 フォーマット



### 6.4.6.3 デフォルト

表 31. Type 0 Routing Header

| 名前             | 種類           | size   | default (送信) | default (受信) |
|----------------|--------------|--------|--------------|--------------|
| 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            |
| Rsaved         | uint         | 32bit  | 0            | 0            |
| Address        | IPv6 Address | 128bit | 省略不可         | 省略不可         |
| 必要な分繰り返し       |              |        |              |              |

#### 6.4.6.4 送信時自動設定

Next Header.

Hdr Ext Len.

- はじめの 8octets を除いたルーティングヘッダの長さ
- 単位は、8octets

## 6.4.7 Fragment Header

### 6.4.7.1 参照 RFC

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

### 6.4.7.2 フォーマット

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

### 6.4.7.3 デフォルト

表 32. Fragment Header

| 名前              | 種類   | size  | default (送信) | default (受信) |
|-----------------|------|-------|--------------|--------------|
| Next Header     | uint | 8bit  | 省略不可         | 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 参照 RFC

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

### 6.4.8.2 フォーマット

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

### 6.4.8.3 デフォルト

表 33. Destination Option Header

| 名前                    | 種類   | size | default (送信) | default (受信) |
|-----------------------|------|------|--------------|--------------|
| Next Header           | uint | 8bit | auto         | any          |
| Header Ext Len        | uint | 8bit | auto         | any          |
| (tunnel encapslation) |      |      |              |              |
| (pad1)                |      |      |              |              |
| (padN)                |      |      |              |              |

### 6.4.8.4 送信時自動設定

Next Header.

Header Ext Len.

- はじめの 8octets を除いた Destinatino Optino Header の長さ
- 単位は 8octets

## 6.4.9 Tunnel Encapsulation Header

### 6.4.9.1 参照 RFC

RFC2473, Generic Packet Tunneling in IPv6 Specification, December 1998

### 6.4.9.2 フォーマット

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

### 6.4.9.3 デフォルト

表 34. Tunnel Encapsulation Header

| 名前           | 種類   | size | default (送信) | default (受信) |
|--------------|------|------|--------------|--------------|
| Option Type  | uint | 8bit | 0x04         | 0x04         |
| Opt Data Len | uint | 8bit | 1            | 1            |
| Limit        | unit | 8bit | 0            | 0            |

## 6.5 IPsec

### 6.5.1 Authentiatino Header

#### 6.5.1.1 参照 RFC

RFC2402, IP Authentication Header, November 1998

#### 6.5.1.2 フォーマット

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

#### 6.5.1.3 デフォルト

表 35. Authentication

| 名前              | 種類   | size  | default (送信) | default (受信) |
|-----------------|------|-------|--------------|--------------|
| 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 送信時自動設定

Payload Length.

- Authentication Header の長さ - 2
- 単位は 4byte

#### 6.5.1.5 その他

- ハッシュ値の大きさは、ハッシュ関数によって違うので、パディングをして8バイト境界になるようにしなければならない padメンバが必要



## 6.5.3 AHAlgorithm

### 6.5.3.1 参照 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 デフォルト

表 37. AH Algorithm Block

| 名前   | 種類   | size | default (送信) | default (受信) |
|------|------|------|--------------|--------------|
| auth | auth |      | 省略不可         | 省略不可         |

## 6.5.4 ESPAlgorithm

### 6.5.4.1 参照 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 Explicit IV, November 1998

### 6.5.4.2 デフォルト

表 38. ESP Algorithm Block

| 名前    | 種類     | size | default (送信) | default (受信) |
|-------|--------|------|--------------|--------------|
| pad   | esppad |      | auto         | any          |
| crypt | crypt  |      | 省略不可         | 省略不可         |
| auth  | auth   |      | null_auth()  | null_auth()  |

## 6.6 ICMPv6 (ND) (RFC2461)

### 6.6.1 Router Solicitation

#### 6.6.1.1 参照 RFC

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

#### 6.6.1.2 フォーマット

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

#### 6.6.1.3 デフォルト

表 39. Router Solicitation

| 名前                     | 種類      | size  | default (送信) | default (受信) |
|------------------------|---------|-------|--------------|--------------|
| 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 送信時自動設定

checksum.

## 6.6.2 Router Advertisement

### 6.6.2.1 参照 RFC

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

### 6.6.2.2 フォーマット

```

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

```

### 6.6.2.3 デフォルト

表 40. Router Advertisement

| 名前                     | 種類      | size  | default (送信) | default (受信) |
|------------------------|---------|-------|--------------|--------------|
| 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 |       |              |              |

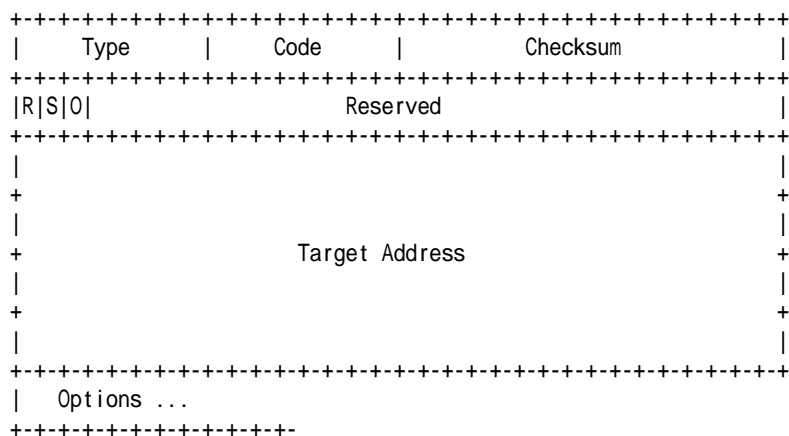


## 6.6.4 Neighbor Advertisement

### 6.6.4.1 参照 RFC

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

### 6.6.4.2 フォーマット



### 6.6.4.3 デフォルト

表 42. Neighbor Advertisement

| 名前                     | 種類           | size   | default (送信)   | default (受信) |
|------------------------|--------------|--------|----------------|--------------|
| 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      |        |                |              |



## 6.6.6 Source Link Layer Address Option

### 6.6.6.1 参照 RFC

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

### 6.6.6.2 フォーマット

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

### 6.6.6.3 デフォルト

表 44. Source Link Layer Address Option (option)

| 名前                 | 種類            | size  | default (送信)      | default (受信)      |
|--------------------|---------------|-------|-------------------|-------------------|
| 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 その他

本来は、ether 以外の link Layer アドレスにも対応しているので、長さは可変。今回は、Ether のみを対象とするので固定。

## 6.6.7 Target Link Layer Address option

### 6.6.7.1 参照 RFC

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

### 6.6.7.2 フォーマット

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

### 6.6.7.3 デフォルト

表 45. Target Link Layer Address Option (option)

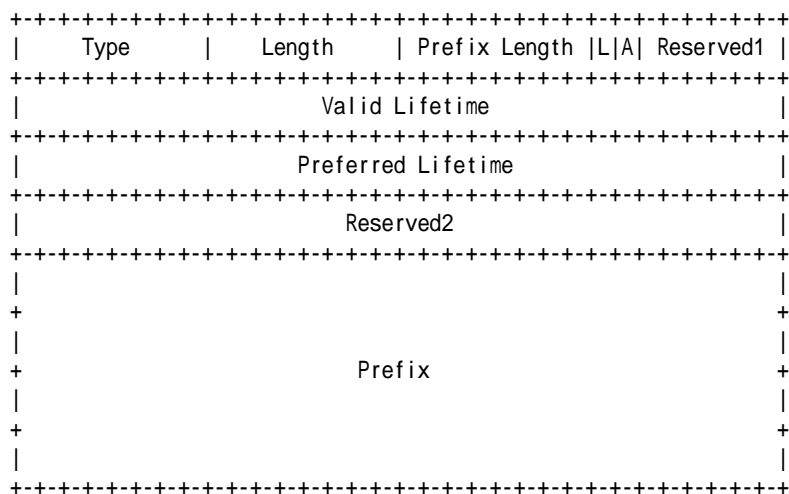
| 名前                 | 種類         | size  | default (送信)      | default (受信)      |
|--------------------|------------|-------|-------------------|-------------------|
| Type               | uint       | 8bit  | 2                 | 2                 |
| Length             | uint       | 8bit  | auto              | use               |
| Link Layer Address | Ether Addr | 48bit | Target Ether Addr | Tester Ether Addr |

## 6.6.8 Prefix Information Option

### 6.6.8.1 参照 RFC

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

### 6.6.8.2 フォーマット



### 6.6.8.3 デフォルト

表 46. Prefix Information (option)

| 名前                 | 種類           | size   | default (送信) | default (受信) |
|--------------------|--------------|--------|--------------|--------------|
| 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 | 省略不可         | 省略不可         |

## 6.6.9 MTU Option

### 6.6.9.1 参照 RFC

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

### 6.6.9.2 フォーマット

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

### 6.6.9.3 デフォルト

表 47. MTU (Option)

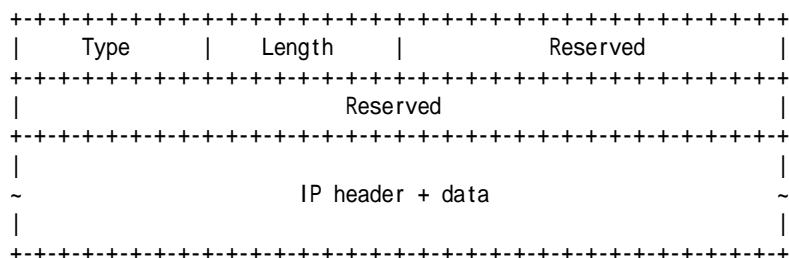
| 名前        | 種類   | size  | default (送信) | default (受信) |
|-----------|------|-------|--------------|--------------|
| Type      | uint | 8bit  | 5            | 5            |
| Length    | uint | 8bit  | 1            | 1            |
| Resereved | uint | 16bit | 0            | 0            |
| MTU       | uint | 32bit | 1500         | 1500         |

## 6.6.10 Redirected Header (Option)

### 6.6.10.1 参照 RFC

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

### 6.6.10.2 フォーマット



### 6.6.10.3 デフォルト

表 48. Redirected Header (option)

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 4            | 4            |
| Length   | uint    | 8bit  | auto         | any          |
| Reserved | uint    | 48bit | 0            | 0            |
| payload  | payload |       | 省略不可         | 省略不可         |

### 6.6.10.4 送信時自動設定

Length.

- オプション全体の長さ
- 単位は 8 octets

### 6.6.10.5 その他

単位が 8octets ということは、payload 部の長さが 8octets の倍数でないため

## 6.7 Information Messages

### 6.7.1 Multicast Listener Query

#### 6.7.1.1 参照 RFC

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

#### 6.7.1.2 フォーマット

```
+++++
| Type | Code | Checksum |
+++++
| Maximum Response Delay | Reserved |
+++++
|
+
|
+
|
+
|
+++++
```

#### 6.7.1.3 デフォルト

表 49. Multicast Listener Query

| 名前                 | 種類           | size   | default (送信) | default (受信) |
|--------------------|--------------|--------|--------------|--------------|
| 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 | 省略不可         | 省略不可         |

## 6.7.2 Multicast Listener Report

### 6.7.2.1 参照 RFC

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

### 6.7.2.2 フォーマット

```
+++++
| Type | Code | Checksum |
+++++
| Maximum Response Delay | Reserved |
+++++
|
+
|
+
|
+
|
+++++
```

### 6.7.2.3 デフォルト

表 50. Multicast Listener Report

| 名前                 | 種類           | size   | default (送信) | default (受信) |
|--------------------|--------------|--------|--------------|--------------|
| 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 | 省略不可         | 省略不可         |

## 6.7.3 Multicast Listener Done

### 6.7.3.1 参照 RFC

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

### 6.7.3.2 フォーマット

```
+++++
| Type | Code | Checksum |
+++++
| Maximum Response Delay | Reserved |
+++++
|
+
|
+
|
+
|
+++++
```

### 6.7.3.3 デフォルト

表 51. Multicast Listener Done

| 名前                 | 種類           | size   | default (送信) | default (受信) |
|--------------------|--------------|--------|--------------|--------------|
| 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 | 省略不可         | 省略不可         |

## 6.7.4 ICMP Echo Request

### 6.7.4.1 参照 RFC

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

### 6.7.4.2 フォーマット

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

### 6.7.4.3 デフォルト

表 52. ICMP Echo Request

| 名前              | 種類      | size  | default (送信) | default (受信) |
|-----------------|---------|-------|--------------|--------------|
| 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 |       | 省略不可         | 省略不可         |

## 6.7.5 Echo Reply

### 6.7.5.1 参照 RFC

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

### 6.7.5.2 フォーマット

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

### 6.7.5.3 デフォルト

表 53. ICMP Echo Reply

| 名前              | 種類      | size  | default (送信) | default (受信) |
|-----------------|---------|-------|--------------|--------------|
| 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 |       | 省略不可         | 省略不可         |

## 6.8 Error Message

### 6.8.1 Packet Too Big

#### 6.8.1.1 参照 RFC

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

#### 6.8.1.2 フォーマット

```
+++++
| 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 デフォルト

表 54. Packet Too Big

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 2            | 2            |
| Code     | uint    | 8bit  | 0            | 0            |
| Checksum | uint    | 16bit | auto         | check        |
| MTU      | uint    | 32bit | 省略不可         | 省略不可         |
| payload  | payload |       | 省略不可         | 省略不可         |

## 6.8.2 Destination Unreachable

### 6.8.2.1 参照 RFC

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

### 6.8.2.2 フォーマット

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

### 6.8.2.3 デフォルト

表 55. Destination Unreachable Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 1            | 1            |
| Code     | uint    | 8bit  | 0            | 0            |
| Checksum | uint    | 16bit | auto         | check        |
| Unused   | uint    | 32bit | 0            | 0            |
| payload  | payload |       | 省略不可         | 省略不可         |

## 6.8.3 Time Exceeded Messages

### 6.8.3.1 参照 RFC

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

### 6.8.3.2 フォーマット

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 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 デフォルト

表 56. Time Exceeded Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 3            | 3            |
| Code     | uint    | 8bit  | 0            | 0            |
| Checksum | uint    | 16bit | auto         | check        |
| Unused   | uint    | 32bit | 0            | 0            |
| payload  | payload |       | 省略不可         | 省略不可         |

## 6.8.4 Parameter Problem Messages

### 6.8.4.1 参照 RFC

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

### 6.8.4.2 フォーマット

```

+++++
| 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 デフォルト

表 57. Parameter Problem Messages

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 4            | 4            |
| Code     | uint    | 8bit  | [12]         | [12]         |
| Checksum | uint    | 16bit | auto         | check        |
| Pointer  | uint    | 32bit | 省略不可         | 省略不可         |
| payload  | payload |       | 省略不可         | 省略不可         |

## 6.9 IPv4

### 6.9.1 IPv4 ヘッダ

#### 6.9.1.1 フォーマット

```

+-----+-----+-----+-----+-----+-----+-----+-----+
|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 デフォルト

表 58. IPv4 ヘッダ

| 名前                  | 種類           | size  | default (送信) | default (受信) |
|---------------------|--------------|-------|--------------|--------------|
| 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 | 省略不可         | 省略不可         |
| Destination Address | IPv4 Address | 32bit | 省略不可         | 省略不可         |
| (option)            | ref          |       |              |              |
| (Padding)           | date         |       |              |              |

## 6.9.2 End Of Option List オプション

### 6.9.2.1 デフォルト

表 59. IPv4 End of Option List

| 名前   | 種類   | size | default (送信) | default (受信) |
|------|------|------|--------------|--------------|
| Type | uint | 8bit | 0            | 0            |

## 6.9.3 No Operation オプション

### 6.9.3.1 デフォルト

表 60. IPv4 No Operation Option

| 名前   | 種類   | size | default (送信) | default (受信) |
|------|------|------|--------------|--------------|
| Type | uint | 8bit | 1            | 1            |

## 6.9.4 Loose Source Route オプション

### 6.9.4.1 デフォルト

表 61. IPv4 Loose Source Route Option

| 名前        | 種類   | size | default (送信) | default (受信) |
|-----------|------|------|--------------|--------------|
| Type      | uint | 8bit | 131          | 131          |
| Length    | uint |      | auto         | auto         |
| Pointer   |      |      |              |              |
| RouteData | IPv4 |      |              |              |
| つづく       |      |      |              |              |

## 6.9.5 Strict Source Route オプション

### 6.9.5.1 デフォルト

表 62. Strict Source Route Option

| 名前        | 種類   | size | default (送信) | default (受信) |
|-----------|------|------|--------------|--------------|
| Type      | uint | 8bit | 137          | 137          |
| Length    | uint |      | auto         | auto         |
| Pointer   |      |      |              |              |
| RouteData | IPv4 |      |              |              |
| つづく       |      |      |              |              |

## 6.9.6 Record Route オプション

### 6.9.6.1 デフォルト

表 63. IPv4 Record Route Option

| 名前        | 種類   | size | default (送信) | default (受信) |
|-----------|------|------|--------------|--------------|
| Type      | uint | 8bit | 7            | 7            |
| Length    | uint |      | auto         | auto         |
| Pointer   |      |      |              |              |
| RouteData | IPv4 |      |              |              |
| つづく       |      |      |              |              |

## 6.9.7 Timestamp オプション

### 6.9.7.1 デフォルト

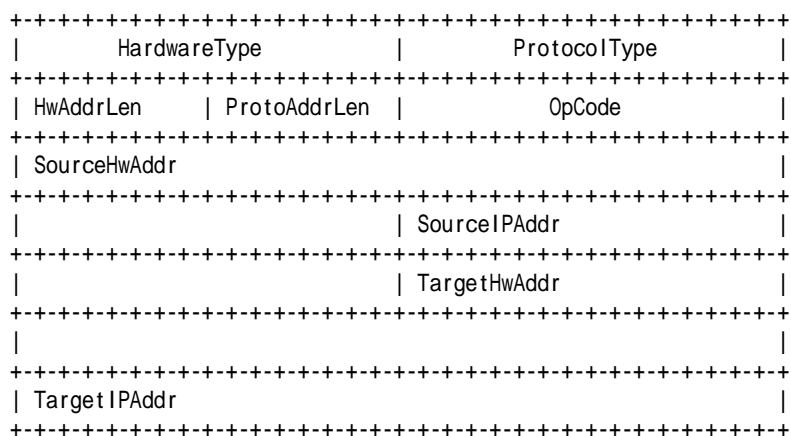
表 64. IPv4 Timestamp Option

| 名前        | 種類   | size | default (送信) | default (受信) |
|-----------|------|------|--------------|--------------|
| Type      | uint | 8bit | 68           | 68           |
| Length    | uint |      | auto         | auto         |
| Pointer   |      |      |              |              |
| Overflow  |      |      |              |              |
| Flag      |      |      |              |              |
| Timestamp |      |      |              |              |
| つづく       |      |      |              |              |

## 6.10 ARP

### 6.10.1 ARP パケット

#### 6.10.1.1 フォーマット



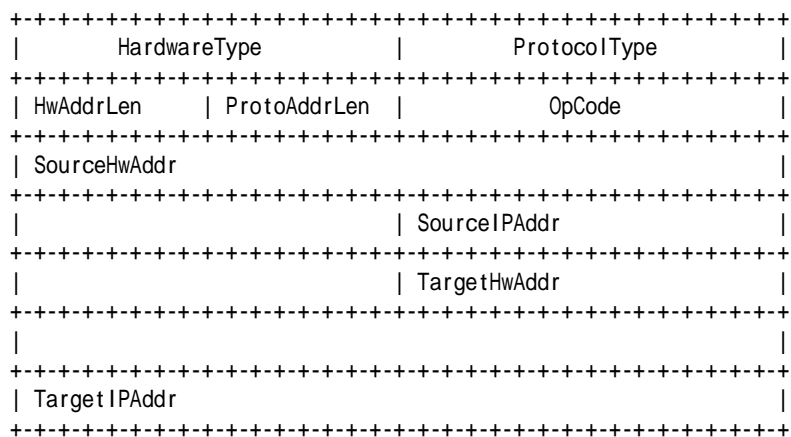
#### 6.10.1.2 デフォルト

表 65. ARP

| 名前          | 種類    | size  | default (送信)   | default (受信)   |
|-------------|-------|-------|----------------|----------------|
| 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 | 省略不可           | 省略不可           |
| TargetHAddr | ether | 48bit | Target Address | Tester Address |
| TargetPAddr | v4    | 32bit | 省略不可           | 省略不可           |

## 6.10.2 RARP パケット

### 6.10.2.1 フォーマット



### 6.10.2.2 デフォルト

表 66. RARP

| 名前          | 種類    | size  | default (送信)   | default (受信)   |
|-------------|-------|-------|----------------|----------------|
| 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 | 省略不可           | 省略不可           |
| TargetHAddr | ether | 48bit | Target Address | Tester Address |
| TargetPAddr | v4    | 32bit | 省略不可           | 省略不可           |

## 6.11 ICMPv4

### 6.11.1 Destination Unreachable Message

#### 6.11.1.1 フォーマット

```

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

```

#### 6.11.1.2 デフォルト

表 67. ICMPv4 Destination Unreachable Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 3            | 3            |
| Code     | uint    | 8bit  | 省略不可         | 省略不可         |
| Checksum | uint    | 16bit | auto         | check        |
| Unused   | uint    | 32bit | 0            | 0            |
| payload  | payload |       | 省略不可         | 省略不可         |

### 6.11.2 Time Exceeded Message

#### 6.11.2.1 フォーマット

```

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

```

### 6.11.2.2 デフォルト

表 68. ICMPv4 Time Exceeded Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 11           | 11           |
| Code     | uint    | 8bit  | 省略不可         | 省略不可         |
| Checksum | uint    | 16bit | auto         | check        |
| Unused   | uint    | 32bit | 0            | 0            |
| payload  | payload |       | 省略不可         | 省略不可         |

### 6.11.3 Parameter Problem Message

#### 6.11.3.1 フォーマット

```

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

```

#### 6.11.3.2 デフォルト

表 69. ICMPv4 Parameter Problem Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| 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 |       | 省略不可         | 省略不可         |

## 6.11.4 Source Quench Message

### 6.11.4.1 フォーマット

```

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

```

### 6.11.4.2 デフォルト

表 70. ICMPv4 Source Quench Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 4            | 4            |
| Code     | uint    | 8bit  | 0            | 0            |
| Checksum | uint    | 16bit | auto         | check        |
| Unused   | uint    | 32bit | 0            | 0            |
| payload  | payload |       | 省略不可         | 省略不可         |

## 6.11.5 Redirect Message

### 6.11.5.1 フォーマット

```

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

```

### 6.11.5.2 デフォルト

表 71. ICMPv4 Redirect Message

| 名前       | 種類      | size  | default (送信) | default (受信) |
|----------|---------|-------|--------------|--------------|
| Type     | uint    | 8bit  | 5            | 5            |
| Code     | uint    | 8bit  | 省略不可         | 省略不可         |
| Checksum | uint    | 16bit | auto         | check        |
| Address  | IPv4    | 32bit | 省略不可         | 省略不可         |
| payload  | payload |       | 省略不可         | 省略不可         |

## 6.11.6 Echo Request Message

### 6.11.6.1 フォーマット

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

### 6.11.6.2 デフォルト

表 72. ICMPv4 Echo Request

| 名前             | 種類      | size  | default (送信) | default (受信) |
|----------------|---------|-------|--------------|--------------|
| 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 |       | 省略不可         | 省略不可         |

## 6.11.7 Echo Reply Message

### 6.11.7.1 フォーマット

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

### 6.11.7.2 デフォルト

表 73. ICMPv4 Echo Reply

| 名前             | 種類      | size  | default (送信) | default (受信) |
|----------------|---------|-------|--------------|--------------|
| 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 |       | 省略不可         | 省略不可         |

## 6.11.8 Packet Too Big

### 6.11.8.1 参照 RFC

RFC1191

### 6.11.8.2 フォーマット

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

### 6.11.8.3 デフォルト

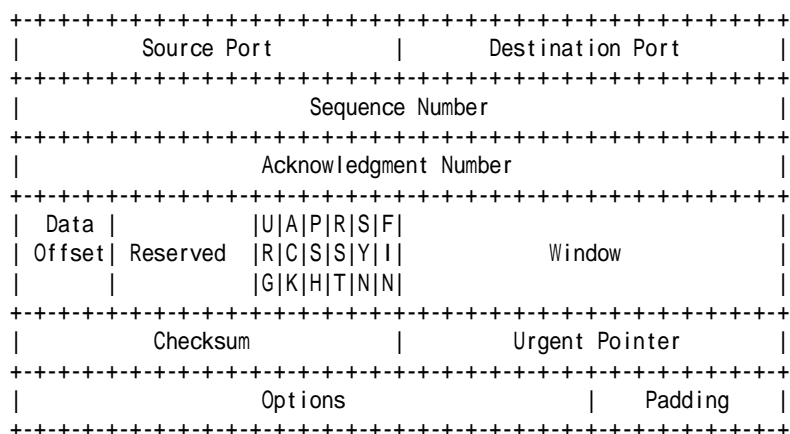
表 74. ICMPv4 Packet Too Big

| 名前         | 種類      | size  | default (送信) | default (受信) |
|------------|---------|-------|--------------|--------------|
| 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 |       | 省略不可         | 省略不可         |

## 6.12 TCP

### 6.12.1 TCP ヘッダ

#### 6.12.1.1 フォーマット



#### 6.12.1.2 デフォルト

表 75. TCP header

| 名前               | 種類      | size  | default (送信) | default (受信) |
|------------------|---------|-------|--------------|--------------|
| Source Port      | uint    | 16bit | 省略不可         | 省略不可         |
| Destinatino Port | uint    | 16bit | 省略不可         | 省略不可         |
| 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    | 16bit | 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 オプション

### 6.12.2.1 デフォルト

表 76. IPv4 End Of Option list Option

| 名前   | 種類   | size | default (送信) | default (受信) |
|------|------|------|--------------|--------------|
| Kind | uint | 8bit | 0            | 0            |

## 6.12.3 No Operation オプション

### 6.12.3.1 デフォルト

表 77. IPv4 No Operation Option

| 名前   | 種類   | size | default (送信) | default (受信) |
|------|------|------|--------------|--------------|
| Kind | uint | 8bit | 1            | 1            |

## 6.12.4 Maximum Segment Size オプション

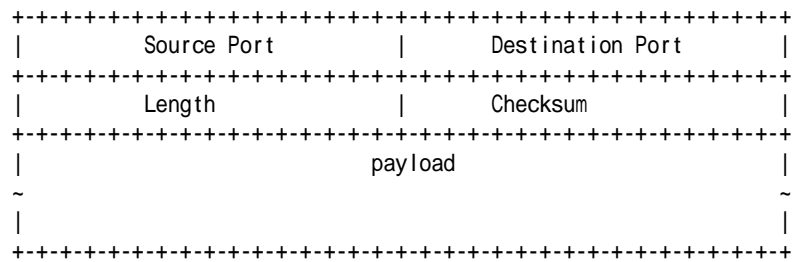
### 6.12.4.1 デフォルト

表 78. IPv4 Maximum Segment Size Option

| 名前         | 種類   | size | default (送信) | default (受信) |
|------------|------|------|--------------|--------------|
| Kind       | uint | 8bit | 2            | 2            |
| Length     |      |      |              |              |
| MaxSegSize |      |      |              |              |

## 6.13 UDP

### 6.13.1 Any UDP



#### 6.13.1.1 デフォルト

表 79. UDP

| 名前               | 種類      | size  | default (送信) | default (受信) |
|------------------|---------|-------|--------------|--------------|
| Source Port      | uint    | 16bit | 省略不可         | 省略不可         |
| Destinatino Port | uint    | 16bit | 省略不可         | 省略不可         |
| Length           | uint    | 16bit | auto         | any          |
| Checksum         | uint    | 16bit | auto         | check        |
| payload          | payload |       | 省略不可         | 省略不可         |