
dpkt

Release 1.9.1

Apr 24, 2017

Contents

1	Getting Started	3
1.1	Installation	3
1.2	Examples	3
2	API Reference	9
2.1	API Reference	9
3	About dpkt	83
3.1	Authors	83
3.2	Changelog	84
3.3	Development plans	84
3.4	Contributing	84
3.5	License	85
4	Administration	87
4.1	Notes	87
	Python Module Index	89

dpkt is a python module for fast, simple packet creation / parsing, with definitions for the basic TCP/IP protocols

CHAPTER 1

Getting Started

Installation

DKPT is now available directly from pypi :)

Install the Code

```
pip install dpkt
```

Checkout the Code

```
git clone https://github.com/kbandla/dpkt.git
```

Examples

Examples in dpkt/examples

Print Packets Example

This example uses DPKT to read in a pcap file and print out the contents of the packets This example is focused on the fields in the Ethernet Frame and IP packet

Code Excerpt

```
# For each packet in the pcap process the contents
for timestamp, buf in pcap:

    # Print out the timestamp in UTC
```


Parameters `pcap` – dpkt pcap reader object (dpkt.pcap.Reader)

`examples.print_packets.test()`
Open up a test pcap file and print out the packets

Print ICMP Example

This example expands on the `print_packets` example. It checks for ICMP packets and displays the ICMP contents.

Code Excerpt

```
# For each packet in the pcap process the contents
for timestamp, buf in pcap:

    # Unpack the Ethernet frame (mac src/dst, ethertype)
    eth = dpkt.ethernet.Ethernet(buf)

    # Make sure the Ethernet data contains an IP packet
    if not isinstance(eth.data, dpkt.ip.IP):
        print 'Non IP Packet type not supported %s\n' % eth.data.__class__.__name__
        continue

    # Now grab the data within the Ethernet frame (the IP packet)
    ip = eth.data

    # Now check if this is an ICMP packet
    if isinstance(ip.data, dpkt.icmp.ICMP):
        icmp = ip.data

        # Pull out fragment information (flags and offset all packed into off field,
        ↪so use bitmasks)
        do_not_fragment = bool(ip.off & dpkt.ip.IP_DF)
        more_fragments = bool(ip.off & dpkt.ip.IP_MF)
        fragment_offset = ip.off & dpkt.ip.IP_OFFMASK

        # Print out the info
        print 'Timestamp: ', str(datetime.datetime.utcnow().timestamp(timestamp))
        print 'Ethernet Frame: ', mac_addr(eth.src), mac_addr(eth.dst), eth.type
        print 'IP: %s -> %s (len=%d ttl=%d DF=%d MF=%d offset=%d)' % \
              (inet_to_str(ip.src), inet_to_str(ip.dst), ip.len, ip.ttl, do_not_
        ↪fragment, more_fragments, fragment_offset)
        print 'ICMP: type:%d code:%d checksum:%d data: %s\n' % (icmp.type, icmp.code,
        ↪icmp.sum, repr(icmp.data))
```

Example Output

```
Timestamp: 2013-05-30 22:45:17.283187
Ethernet Frame: 60:33:4b:13:c5:58 02:1a:11:f0:c8:3b 2048
IP: 192.168.43.9 -> 8.8.8.8 (len=84 ttl=64 DF=0 MF=0 offset=0)
ICMP: type:8 code:0 checksum:48051 data: Echo(id=55099, data='Q\xa7\xd6')
↪\x00\x04Q\xe4\x08\t\n\x0b\x0c\r\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e
↪!#$%&\'()*+,-./01234567')

Timestamp: 2013-05-30 22:45:17.775391
Ethernet Frame: 02:1a:11:f0:c8:3b 60:33:4b:13:c5:58 2048
IP: 8.8.8.8 -> 192.168.43.9 (len=84 ttl=40 DF=0 MF=0 offset=0)
ICMP: type:0 code:0 checksum:50099 data: Echo(id=55099, data='Q\xa7\xd6')
↪\x00\x04Q\xe4\x08\t\n\x0b\x0c\r\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e
↪!#$%&\'()*+,-./01234567')
```

```
...
```

dpkt/examples/print_icmp.py This example expands on the `print_packets` example. It checks for ICMP packets and displays the ICMP contents.

```
examples.print_icmp.mac_addr(address)
```

Convert a MAC address to a readable/printable string

Parameters `address` (*str*) – a MAC address in hex form (e.g. “”””””””””)

Returns Printable/readable MAC address

Return type *str*

```
examples.print_icmp.inet_to_str(inet)
```

Convert inet object to a string

Parameters `inet` (*inet struct*) – inet network address

Returns Printable/readable IP address

Return type *str*

```
examples.print_icmp.print_icmp(pcap)
```

Print out information about each packet in a pcap

Parameters `pcap` – dpkt pcap reader object (`dpkt.pcap.Reader`)

```
examples.print_icmp.test()
```

Open up a test pcap file and print out the packets

Print HTTP Requests Example

This example expands on the `print_packets` example. It checks for HTTP request headers and displays their contents.

NOTE: We are not reconstructing ‘flows’ so the request (and response if you tried to parse it) will only parse correctly if they fit within a single packet. Requests can often fit in a single packet but Responses almost never will. For proper reconstruction of flows you may want to look at other projects that use DPKT (<http://chains.readthedocs.io> and others)

Code Excerpt

```
# For each packet in the pcap process the contents
for timestamp, buf in pcap:

    # Unpack the Ethernet frame (mac src/dst, ethertype)
    eth = dpkt.ethernet.Ethernet(buf)

    # Make sure the Ethernet data contains an IP packet
    if not isinstance(eth.data, dpkt.ip.IP):
        print 'Non IP Packet type not supported %s\n' % eth.data.__class__.__name__
        continue

    # Now grab the data within the Ethernet frame (the IP packet)
    ip = eth.data

    # Check for TCP in the transport layer
    if isinstance(ip.data, dpkt.tcp.TCP):

        # Set the TCP data
        tcp = ip.data
```

```

# Now see if we can parse the contents as a HTTP request
try:
    request = dpkt.http.Request(tcp.data)
except (dpkt.dpkt.NeedData, dpkt.dpkt.UnpackError):
    continue

# Pull out fragment information (flags and offset all packed into off field,
↳so use bitmasks)
do_not_fragment = bool(ip.off & dpkt.ip.IP_DF)
more_fragments = bool(ip.off & dpkt.ip.IP_MF)
fragment_offset = ip.off & dpkt.ip.IP_OFFMASK

# Print out the info
print 'Timestamp: ', str(datetime.datetime.utcnow().timestamp())
print 'Ethernet Frame: ', mac_addr(eth.src), mac_addr(eth.dst), eth.type
print 'IP: %s -> %s (len=%d ttl=%d DF=%d MF=%d offset=%d)' % \
    (inet_to_str(ip.src), inet_to_str(ip.dst), ip.len, ip.ttl, do_not_
↳fragment, more_fragments, fragment_offset)
print 'HTTP request: %s\n' % repr(request)

```

Example Output

```

Timestamp: 2004-05-13 10:17:08.222534
Ethernet Frame: 00:00:01:00:00:00 fe:ff:20:00:01:00 2048
IP: 145.254.160.237 -> 65.208.228.223 (len=519 ttl=128 DF=1 MF=0 offset=0)
HTTP request: Request(body='', uri='/download.html', headers={'accept-language': 'en-
↳us,en;q=0.5', 'accept-encoding': 'gzip,deflate', 'connection': 'keep-alive', 'keep-
↳alive': '300', 'accept': 'text/xml,application/xml,application/xhtml+xml,text/html;
↳q=0.9,text/plain;q=0.8,image/png,image/jpeg,image/gif;q=0.2,*/*;q=0.1', 'user-agent
↳': 'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6) Gecko/20040113',
↳'accept-charset': 'ISO-8859-1,utf-8;q=0.7,*;q=0.1', 'host': 'www.ethereal.com',
↳'referer': 'http://www.ethereal.com/development.html'}, version='1.1', data='',
↳method='GET')

Timestamp: 2004-05-13 10:17:10.295515
Ethernet Frame: 00:00:01:00:00:00 fe:ff:20:00:01:00 2048
IP: 145.254.160.237 -> 216.239.59.99 (len=761 ttl=128 DF=1 MF=0 offset=0)
HTTP request: Request(body='', uri='/pagead/ads?client=ca-pub-2309191948673629&
↳random=1084443430285&mt=1082467020&format=468x60_as&output=html&url=http%3A%2F
↳%2Fwww.ethereal.com%2Fdownload.html&color_bg=FFFFFF&color_text=333333&color_
↳link=000000&color_url=666633&color_border=666633', headers={'accept-language': 'en-
↳us,en;q=0.5', 'accept-encoding': 'gzip,deflate', 'connection': 'keep-alive', 'keep-
↳alive': '300', 'accept': 'text/xml,application/xml,application/xhtml+xml,text/html;
↳q=0.9,text/plain;q=0.8,image/png,image/jpeg,image/gif;q=0.2,*/*;q=0.1', 'user-agent
↳': 'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6) Gecko/20040113',
↳'accept-charset': 'ISO-8859-1,utf-8;q=0.7,*;q=0.1', 'host': 'pagead2.
↳googlesyndication.com', 'referer': 'http://www.ethereal.com/download.html'},
↳version='1.1', data='', method='GET')

...

```

dpkt/examples/print_http_requests.py This example expands on the `print_packets` example. It checks for HTTP request headers and displays their contents. NOTE: We are not reconstructing ‘flows’ so the request (and response if you tried to parse it) will only

parse correctly if they fit within a single packet. Requests can often fit in a single packet but Responses almost never will. For proper reconstruction of flows you may want to look at other projects that use

DPKT (<http://chains.readthedocs.io> and others)

`examples.print_http_requests.mac_addr(address)`

Convert a MAC address to a readable/printable string

Parameters `address` (*str*) – a MAC address in hex form (e.g. “”””””””””)

Returns Printable/readable MAC address

Return type `str`

`examples.print_http_requests.inet_to_str(inet)`

Convert inet object to a string

Parameters `inet` (*inet struct*) – inet network address

Returns Printable/readable IP address

Return type `str`

`examples.print_http_requests.print_http_requests(pcap)`

Print out information about each packet in a pcap

Parameters `pcap` – dpkt pcap reader object (`dpkt.pcap.Reader`)

`examples.print_http_requests.test()`

Open up a test pcap file and print out the packets

Jon Oberheide’s Examples

[@jonoberheide’s](<https://twitter.com/jonoberheide>) old examples still apply:

- dpkt Tutorial #1: ICMP Echo
- dpkt Tutorial #2: Parsing a PCAP File
- dpkt Tutorial #3: dns spoofing
- dpkt Tutorial #4: AS Paths from MRT/BGP

Jeff Silverman Docs/Code

Jeff Silverman has some [code](#) and [documentation](#).

API Reference

The dpkt API reference section is currently a work in progress, please have patience as we fill in and improve the documentation.

dpkt Modules

dpkt.ah module

Authentication Header.

```
class dpkt.ah.AH(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Authentication Header.
    TODO: Longer class information....
    __hdr__
        Header fields of AH.
    auth
        Authentication body.
    data
        Message data.
    auth = ''
    unpack(buf)
    data
    len
    nxt
```

rsvd
seq
spi

dpkt.aim module

AOL Instant Messenger.

class dpkt.aim.**FLAP** (*args, **kwargs)
Bases: *dpkt.dpkt.Packet*

Frame Layer Protocol.

See more about the FLAP on https://en.wikipedia.org/wiki/OSCAR_protocol#FLAP_header

__hdr__
Header fields of FLAP.

data
Message data.

unpack (buf)

ast

data

len

seq

type

class dpkt.aim.**SNAC** (*args, **kwargs)
Bases: *dpkt.dpkt.Packet*

Simple Network Atomic Communication.

See more about the SNAC on https://en.wikipedia.org/wiki/OSCAR_protocol#SNAC_data

__hdr__
Header fields of SNAC.

data

family

flags

reqid

subtype

dpkt.aim.**tlv** (buf)

dpkt.aim.**testAIM** ()

dpkt.aim.**testExceptions** ()

dpkt.aoe module

ATA over Ethernet Protocol.

class `dpkt.aoe.AOE` (**args, **kwargs*)

Bases: *dpkt.dpkt.Packet*

ATA over Ethernet Protocol.

See more about the AOE on https://en.wikipedia.org/wiki/ATA_over_Ethernet

__hdr__

Header fields of AOE.

data

Message data.

ver

fl

classmethod `set_cmd` (*cmd, pktclass*)

classmethod `get_cmd` (*cmd*)

unpack (*buf*)

pack_hdr ()

cmd

data

err

maj

min

tag

ver_fl

dpkt.aoeata module

ATA over Ethernet ATA command

class `dpkt.aoeata.AOEATA` (**args, **kwargs*)

Bases: *dpkt.dpkt.Packet*

ATA over Ethernet ATA command.

See more about the AOEATA on https://en.wikipedia.org/wiki/ATA_over_Ethernet

__hdr__

Header fields of AOEATA.

data

Message data.

aflags

cmdstat

data

errfeat

lba0

lba1

lba2

lba3

lba4

lba5

res

scnt

`dpkt.aoeata.test_aoeata()`

dpkt.aoecfg module

ATA over Ethernet ATA command

class `dpkt.aoecfg.AOECFG(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

ATA over Ethernet ATA command.

See more about the AOE on https://en.wikipedia.org/wiki/ATA_over_Ethernet

__hdr__

Header fields of AOECFG.

data

Message data.

aoeccmd

bufcnt

cslen

data

fwver

scnt

`dpkt.aoecfg.test_aoecfg()`

dpkt.arp module

Address Resolution Protocol.

class `dpkt.arp.ARP(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

Address Resolution Protocol.

See more about the ARP on https://en.wikipedia.org/wiki/Address_Resolution_Protocol

__hdr__

Header fields of ARP.

data
hln
hrd
op
pln
pro
sha
spa
tha
tpa

dpkt.asn1 module

Abstract Syntax Notation #1.

`dpkt.asn1.utctime(buf)`

Convert ASN.1 UTCTime string to UTC float.

TODO: Long description here.

Parameters **buf** – A buffer with format “yymnddhmm”

Returns A floating point number, indicates seconds since the Epoch.

`dpkt.asn1.decode(buf)`

Sleazy ASN.1 decoder.

TODO: Long description here.

Parameters **buf** – A buffer with Sleazy ASN.1 data.

Returns A list of (id, value) tuples from ASN.1 BER/DER encoded buffer.

Raises `UnpackError` – An error occurred the ASN.1 length exceed.

`dpkt.asn1.test_asn1()`

dpkt.bgp module

Border Gateway Protocol.

class `dpkt.bgp.BGP(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

Border Gateway Protocol.

BGP is an inter-AS routing protocol. See more about the BGP on https://en.wikipedia.org/wiki/Border_Gateway_Protocol

__hdr__

Header fields of BGP.

#TODO

unpack(*buf*)

```
class Open (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

class Parameter (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

class Authentication (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    code
    data

class BGP.Open.Parameter.Capability (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    code
    data
    len

    BGP.Open.Parameter.data
    BGP.Open.Parameter.len
    BGP.Open.Parameter.type

    BGP.Open.asn
    BGP.Open.data
    BGP.Open.holdtime
    BGP.Open.identifier
    BGP.Open.param_len
    BGP.Open.v

class BGP.Update (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

class Attribute (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    optional
    transitive
    partial
    extended_length
    unpack (buf)

class Origin (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
```

```

    type
class BGP.Update.Attribute.ASPath(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack(buf)
class ASPathSegment(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack(buf)
    data
    len
    type
class BGP.Update.Attribute.NextHop(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    ip
class BGP.Update.Attribute.MultiExitDisc(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    value
class BGP.Update.Attribute.LocalPref(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    value
class BGP.Update.Attribute.AtomicAggregate(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack(buf)
class BGP.Update.Attribute.Aggregator(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    asn
    data
    ip
class BGP.Update.Attribute.Communities(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack(buf)
class Community(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    asn
    data
    value

```

```
class BGP.Update.Attribute.Communities.ReservedCommunity (*args,
                                                         **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    value

class BGP.Update.Attribute.OriginatorID (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    value

class BGP.Update.Attribute.ClusterList (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack (buf)

class BGP.Update.Attribute.MPReachNLRI (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack (buf)
    class SNPA
        Bases: object
        unpack (buf)
    BGP.Update.Attribute.MPReachNLRI.afi
    BGP.Update.Attribute.MPReachNLRI.data
    BGP.Update.Attribute.MPReachNLRI.safi

class BGP.Update.Attribute.MPUnreachNLRI (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack (buf)
    afi
    data
    safi

BGP.Update.Attribute.data
BGP.Update.Attribute.flags
BGP.Update.Attribute.type

class BGP.Notification (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack (buf)
    code
    data
    subcode

class BGP.Keepalive (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack (buf)
```

```

class BGP.RouteRefresh(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    afi
    data
    rsvd
    safi

BGP.data
BGP.len
BGP.marker
BGP.type

class dpkt.bgp.RouteGeneric(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack(buf)

    data
    len

class dpkt.bgp.RouteIPv4(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack(buf)

    data
    len

class dpkt.bgp.RouteIPv6(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack(buf)

    data
    len

dpkt.bgp.test_pack()
dpkt.bgp.test_unpack()

```

dpkt.cdp module

Cisco Discovery Protocol.

```

class dpkt.cdp.CDP(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Cisco Discovery Protocol.

    See more about the BGP on https://en.wikipedia.org/wiki/Cisco\_Discovery\_Protocol

    __hdr__
        Header fields of CDP.

    #TODO

```

```
class Address (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    alen
    data
    p
    plen
    ptype

class CDP.TLV (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    data
    len
    type

CDP.unpack (buf)

CDP.data

CDP.sum

CDP.ttl

CDP.version
```

dpkt.crc32c module

```
dpkt.crc32c.add (crc, buf)

dpkt.crc32c.done (crc)

dpkt.crc32c.cksum (buf)
    Return computed CRC-32c checksum.

dpkt.crc32c.test_crc32c ()
```

dpkt.decorators module

```
dpkt.decorators.decorator_with_args (decorator_to_enhance)
    This is decorator for decorator. It allows any decorator to get additional arguments

dpkt.decorators.deprecated (*args, **kwargs)

class dpkt.decorators.TestDeprecatedDecorator
    Bases: object

    new_method ()

    old_method (*args, **kwargs)

    deprecated_decorator (*args, **kwargs)

    test_deprecated_decorator ()
```

dpkt.dhcp module

Dynamic Host Configuration Protocol.

```
class dpkt.dhcp.DHCP(*args, **kwargs)
    Bases: dpkt.Packet
    Dynamic Host Configuration Protocol.
    TODO: Longer class information....
    __hdr__
        Header fields of DHCP.
    TODO.
    opts = ((53, '\x01'), (55, '2\x03\x01\x06'))
    pack_opts()
        Return packed options string.
    unpack(buf)
    chaddr
    ciaddr
    data
    file
    flags
    giaddr
    hln
    hops
    hrd
    magic
    op
    secs
    siaddr
    sname
    xid
    yiaddr
```

```
dpkt.dhcp.test_dhcp()
```

dpkt.diameter module

Diameter.

```
class dpkt.diameter.Diameter(*args, **kwargs)
    Bases: dpkt.Packet
    Diameter.
    TODO: Longer class information....
```

```
__hdr__
    Header fields of Diameter.

TODO.

request_flag
proxiable_flag
error_flag
retransmit_flag
unpack (buf)
pack_hdr ()
app_id
cmd
data
end_id
flags
hop_id
len
v

class dpkt.diameter.AVP (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    vendor_flag
    mandatory_flag
    protected_flag
    code
    data
    flags
    len
    unpack (buf)
    pack_hdr ()

dpkt.diameter.test_pack ()
dpkt.diameter.test_unpack ()
```

dpkt.dns module

Domain Name System.

`dpkt.dns.pack_name (name, off, label_ptrs)`

`dpkt.dns.unpack_name (buf, off)`

```

class dpkt.dns.DNS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Domain Name System.

    TODO: Longer class information....

    __hdr__
        Header fields of DNS.

    TODO.

    qr
    opcode
    aa
    tc
    rd
    ra
    zero
    rcode

class Q(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    DNS question.

    unpack(buf)

    cls
    data
    name
    type

class DNS.RR(*args, **kwargs)
    Bases: dpkt.dns.Q

    DNS resource record.

    pack_rdata(off, label_ptrs)
    unpack_rdata(buf, off)

    cls
    data
    name
    rdata
    rlen
    ttl
    type

DNS.pack_q(buf, q)
    Append packed DNS question and return buf.

```

```
DNS.unpack_q (buf, off)
    Return DNS question and new offset.

DNS.pack_rr (buf, rr)
    Append packed DNS RR and return buf.

DNS.unpack_rr (buf, off)
    Return DNS RR and new offset.

DNS.unpack (buf)

DNS.an

DNS.ar

DNS.data

DNS.id

DNS.ns

DNS.op

DNS.qd

dpkt.dns.test_basic()

dpkt.dns.test_PTR()

dpkt.dns.test_OPT()

dpkt.dns.test_pack_name()

dpkt.dns.test_random_data()

dpkt.dns.test_circular_pointers()

dpkt.dns.test_very_long_name()

dpkt.dns.test_null_response()

dpkt.dns.test_txt_response()
```

dpkt.dpkt module

Simple packet creation and parsing.

exception dpkt.dpkt.**Error**

Bases: `exceptions.Exception`

exception dpkt.dpkt.**UnpackError**

Bases: `dpkt.dpkt.Error`

exception dpkt.dpkt.**NeedData**

Bases: `dpkt.dpkt.UnpackError`

exception dpkt.dpkt.**PackError**

Bases: `dpkt.dpkt.Error`

class dpkt.dpkt.**Packet** (**args, **kwargs*)

Bases: `dpkt.dpkt.Temp`

Base packet class, with metaclass magic to generate members from `self.__hdr__`.

`__hdr__`

Packet header should be defined as a list of (name, structfmt, default) tuples.

__byte_order__

Byte order, can be set to override the default ('>')

Example: >>> class Foo(Packet): ... __hdr__ = (('foo', 'I', 1), ('bar', 'H', 2), ('baz', '4s', 'quux')) ... >>> foo = Foo(bar=3) >>> foo Foo(bar=3) >>> str(foo) ""'quux' >>> foo.bar 3 >>> foo.baz 'quux' >>> foo.foo = 7 >>> foo.baz = 'whee' >>> foo Foo(baz='whee', foo=7, bar=3) >>> Foo('hello, world!') Foo(baz=' wor', foo=1751477356L, bar=28460, data='ld!')

pack_hdr()

Return packed header string.

pack()

Return packed header + self.data string.

unpack(buf)

Unpack packet header fields from buf, and set self.data.

dpkt.dpkt.**hexdump**(buf, length=16)

Return a hexdump output string of the given buffer.

dpkt.dpkt.**in_cksum_add**(s, buf)

dpkt.dpkt.**in_cksum_done**(s)

dpkt.dpkt.**in_cksum**(buf)

Return computed Internet checksum.

dpkt.dpkt.**test_utils**()

dpkt.dtp module

Dynamic Trunking Protocol.

class dpkt.dtp.**DTP**(*args, **kwargs)

Bases: *dpkt.dpkt.Packet*

Dynamic Trunking Protocol.

TODO: Longer class information....

__hdr__

Header fields of DTP.

TODO.

unpack(buf)

data

v

dpkt.esp module

Encapsulated Security Protocol.

class dpkt.esp.**ESP**(*args, **kwargs)

Bases: *dpkt.dpkt.Packet*

Encapsulated Security Protocol.

TODO: Longer class information....

```
__hdr__
    Header fields of ESP.

TODO.

data

seq

spi
```

dpkt.ethernet module

Ethernet II, LLC (802.3+802.2), LLC/SNAP, and Novell raw 802.3, with automatic 802.1q, MPLS, PPPoE, and Cisco ISL decapsulation.

```
dpkt.ethernet.isstr(s)
```

```
class dpkt.ethernet.Ethernet(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
```

Ethernet.

Ethernet II, LLC (802.3+802.2), LLC/SNAP, and Novell raw 802.3, with automatic 802.1q, MPLS, PPPoE, and Cisco ISL decapsulation.

```
__hdr__
    Header fields of Ethernet.
```

```
TODO.
```

```
unpack(buf)
```

```
pack_hdr()
```

```
classmethod set_type(t, pktclass)
```

```
classmethod get_type(t)
```

```
classmethod get_type_rev(k)
```

```
data
```

```
dst
```

```
src
```

```
type
```

```
class dpkt.ethernet.MPLSlabel(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
```

A single entry in MPLS label stack

```
unpack(buf)
```

```
pack_hdr()
```

```
as_tuple()
```

```
data
```

```
class dpkt.ethernet.VLANtag8021Q(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
```

IEEE 802.1q VLAN tag

```

    unpack(buf)
    pack_hdr()
    as_tuple()
    data
    type
class dpkt.ethernet.VLANtagISL(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Cisco Inter-Switch Link VLAN tag
    unpack(buf)
    pack_hdr()
    da
    data
    hsa
    indx
    len
    res
    sa
    snap
dpkt.ethernet.test_eth()
dpkt.ethernet.test_eth_init_with_data()
dpkt.ethernet.test_mpls_label()
dpkt.ethernet.test_802dot1q_tag()
dpkt.ethernet.test_isl_tag()
dpkt.ethernet.test_eth_802dot1q()
dpkt.ethernet.test_eth_802dot1q_stacked()
dpkt.ethernet.test_eth_mpls_stacked()
dpkt.ethernet.test_isl_eth_llc_stp()
dpkt.ethernet.test_eth_llc_snap_cdp()
dpkt.ethernet.test_eth_llc_ipx()
dpkt.ethernet.test_eth_pppoe()

```

dpkt.gre module

Generic Routing Encapsulation.

```

class dpkt.gre.GRE(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Generic Routing Encapsulation.
    TODO: Longer class information....

```

```
    __hdr__
        Header fields of GRE.

TODO.

sre = ()

v

recur

class SRE (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    data

    family

    len

    off

    GRE.opt_fields_fmts ()

    GRE.unpack (buf)

    GRE.data

    GRE.flags

    GRE.p

dpkt.gre.test_gre_v1 ()
```

dpkt.gzip module

GNU zip.

```
class dpkt.gzip.GzipExtra (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data

    id

    len

class dpkt.gzip.Gzip (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    pack_hdr ()

    compress ()
        Compress self.data.

    decompress ()
        Return decompressed payload.

    comment

    data

    extra
```

```

    filename
    flags
    magic
    method
    mtime
    os
    xflags
class dpkt.gzip.TestGzip
    Bases: object
    This data is created with the gzip command line tool
    classmethod setup_class ()
    test_method ()
    test_flags ()
    test_mtime ()
    test_xflags ()
    test_os ()
    test_filename ()
    test_decompress ()

```

dpkt.h225 module

ITU-T H.225.0 Call Signaling.

```

class dpkt.h225.H225 (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    ITU-T H.225.0 Call Signaling.
    TODO: Longer class information...
    __hdr__
        Header fields of H225.
    TODO.
    unpack (buf)
class IE (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack (buf)
    data
    type
H225.data
H225.proto
H225.ref_len

```

```
dpkt.h225.test_pack()  
dpkt.h225.test_unpack()
```

dpkt.hsrp module

Cisco Hot Standby Router Protocol.

```
class dpkt.hsrp.HSRP (*args, **kwargs)  
    Bases: dpkt.dpkt.Packet  
  
    Cisco Hot Standby Router Protocol.  
    TODO: Longer class information....  
  
    __hdr__  
        Header fields of HSRP.  
  
    TODO.  
  
    auth  
  
    data  
  
    group  
  
    hello  
  
    hold  
  
    opcode  
  
    priority  
  
    rsvd  
  
    state  
  
    version  
  
    vip
```

dpkt.http module

Hypertext Transfer Protocol.

```
dpkt.http.parse_headers (f)  
    Return dict of HTTP headers parsed from a file object.  
  
dpkt.http.parse_body (f, headers)  
    Return HTTP body parsed from a file object, given HTTP header dict.  
  
class dpkt.http.Message (*args, **kwargs)  
    Bases: dpkt.dpkt.Packet  
  
    Hypertext Transfer Protocol headers + body.  
    TODO: Longer class information....  
  
    __hdr__  
        Header fields of HTTP.  
  
    TODO.  
  
    headers = None
```

```

    body = None

    unpack (buf, is_body_allowed=True)

    pack_hdr ()

class dpkt.http.Request (*args, **kwargs)
    Bases: dpkt.http.Message

    Hypertext Transfer Protocol Request.

    TODO: Longer class information....

    __hdr__
        Header fields of HTTP request.

    TODO.

    unpack (buf)

class dpkt.http.Response (*args, **kwargs)
    Bases: dpkt.http.Message

    Hypertext Transfer Protocol Response.

    TODO: Longer class information....

    __hdr__
        Header fields of HTTP Response.

    TODO.

    unpack (buf)

dpkt.http.test_parse_request ()
dpkt.http.test_format_request ()
dpkt.http.test_chunked_response ()
dpkt.http.test_multicookie_response ()
dpkt.http.test_noreason_response ()
dpkt.http.test_body_forbidden_response ()
dpkt.http.test_request_version ()
dpkt.http.test_invalid_header ()
dpkt.http.test_gzip_response ()

```

dpkt.icmp module

Internet Control Message Protocol.

```

class dpkt.icmp.ICMP (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Internet Control Message Protocol.

    TODO: Longer class information....

    __hdr__
        Header fields of ICMP.

    TODO.

```

```
class Echo (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    id
    seq

class ICMP.Quote (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack(buf)

    data
    pad

class ICMP.Unreach (*args, **kwargs)
    Bases: dpkt.icmp.Quote

    data
    mtu
    pad

class ICMP.Quench (*args, **kwargs)
    Bases: dpkt.icmp.Quote

    data
    pad

class ICMP.Redirect (*args, **kwargs)
    Bases: dpkt.icmp.Quote

    data
    gw

class ICMP.ParamProbe (*args, **kwargs)
    Bases: dpkt.icmp.Quote

    data
    pad1
    pad2
    ptr

class ICMP.TimeExceed (*args, **kwargs)
    Bases: dpkt.icmp.Quote

    data
    pad

ICMP.unpack(buf)

ICMP.code

ICMP.data

ICMP.sum

ICMP.type
```

```
dpkt.icmp.test_icmp()
```

dpkt.icmp6 module

Internet Control Message Protocol for IPv6.

```
class dpkt.icmp6.ICMP6(*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

Internet Control Message Protocol for IPv6.

TODO: Longer class information....

```
__hdr__
```

Header fields of ICMPv6.

TODO.

```
class Error(*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

```
unpack(buf)
```

data

pad

```
class ICMP6.Unreach(*args, **kwargs)
```

Bases: *dpkt.icmp6.Error*

data

pad

```
class ICMP6.TooBig(*args, **kwargs)
```

Bases: *dpkt.icmp6.Error*

data

mtu

```
class ICMP6.TimeExceed(*args, **kwargs)
```

Bases: *dpkt.icmp6.Error*

data

pad

```
class ICMP6.ParamProb(*args, **kwargs)
```

Bases: *dpkt.icmp6.Error*

data

ptr

```
class ICMP6.Echo(*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

data

id

seq

```
ICMP6.unpack(buf)
```

ICMP6.code

ICMP6.**data**

ICMP6.**sum**

ICMP6.**type**

dpkt.ieee80211 module

IEEE 802.11.

class dpkt.ieee80211.**IEEE80211** (*args, **kwargs)

Bases: *dpkt.dpkt.Packet*

IEEE 802.11.

TODO: Longer class information....

__**hdr**__

Header fields of IEEE802.11.

TODO.

version

type

subtype

to_ds

from_ds

more_frag

retry

pwr_mgt

more_data

wep

order

unpack_ies (*buf*)

class **Capability** (*field*)

Bases: object

IEEE80211.**unpack** (*buf*)

class IEEE80211.**BlockAckReq** (*args, **kwargs)

Bases: *dpkt.dpkt.Packet*

ctl

data

dst

seq

src

class IEEE80211.**BlockAck** (*args, **kwargs)

Bases: *dpkt.dpkt.Packet*

compressed

```

    ack_policy
    multi_tid
    tid
    unpack(buf)
    ctl
    data
    dst
    seq
    src

class IEEE80211.RTS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    dst
    src

class IEEE80211.CTS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    dst

class IEEE80211.ACK(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    dst

class IEEE80211.CFEnd(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    dst
    src

class IEEE80211.MGMT_Frame(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    bssid
    data
    dst
    frag_seq
    src

class IEEE80211.Beacon(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    capability
    data

```

```
    interval
    timestamp
class IEEE80211.Disassoc(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    reason
class IEEE80211.Assoc_Req(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    capability
    data
    interval
class IEEE80211.Assoc_Resp(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    aid
    capability
    data
    status
class IEEE80211.Reassoc_Req(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    capability
    current_ap
    data
    interval
class IEEE80211.Auth(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    algorithm
    auth_seq
    data
class IEEE80211.Deauth(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    reason
class IEEE80211.Action(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack(buf)
    category
    code
    data
```

```

class IEEE80211.BlockAckActionRequest(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    dialog
    parameters
    starting_seq
    timeout

class IEEE80211.BlockAckActionResponse(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    dialog
    parameters
    status_code
    timeout

class IEEE80211.Data(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    bssid
    data
    dst
    frag_seq
    src

class IEEE80211.DataFromDS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    bssid
    data
    dst
    frag_seq
    src

class IEEE80211.DataToDS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    bssid
    data
    dst
    frag_seq
    src

class IEEE80211.DataInterDS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    da

```

```
    data
    dst
    frag_seq
    sa
    src

class IEEE80211.QoS_Data(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    control
    data

class IEEE80211.IE(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    unpack(buf)
    data
    id
    len

class IEEE80211.FH(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    hopindex
    hoppattern
    hopset
    id
    len
    tu

class IEEE80211.DS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    ch
    data
    id
    len

class IEEE80211.CF(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    count
    data
    dur
    id
    len
    max
```

```

    period
IEEE80211.data
IEEE80211.duration
IEEE80211.framectl
class IEEE80211.TIM(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    count
    ctrl
    data
    id
    len
    period
    unpack(buf)
class IEEE80211.IBSS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    atim
    data
    id
    len
dpkt.ieee80211.test_802211_ack()
dpkt.ieee80211.test_80211_beacon()
dpkt.ieee80211.test_80211_data()
dpkt.ieee80211.test_80211_data_qos()
dpkt.ieee80211.test_bug()
dpkt.ieee80211.test_data_ds()
dpkt.ieee80211.test_compressed_block_ack()
dpkt.ieee80211.test_action_block_ack_request()
dpkt.ieee80211.test_action_block_ack_response()

```

dpkt.igmp module

Internet Group Management Protocol.

```

class dpkt.igmp.IGMP(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Internet Group Management Protocol.
    TODO: Longer class information....
    __hdr__
        Header fields of IGMP.

```

`TODO.`
`data`
`group`
`maxresp`
`sum`
`type`

dpkt.ip module

Internet Protocol.

```
class dpkt.ip.IP (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Internet Protocol.
    TODO: Longer class information....

    __hdr__
        Header fields of IP.

    TODO.

    opts = ''
    len
    v
    hl
    rf
    df
    mf
    offset
    unpack (buf)
    classmethod set_proto (p, pktclass)
    classmethod get_proto (p)
    data
    dst
    id
    off
    p
    src
    sum
    tos
    ttl

dpkt.ip.test_ip()
```

```

dpkt.ip.test_hl()
dpkt.ip.test_opt()
dpkt.ip.test_zerolen()
dpkt.ip.test_constructor()
dpkt.ip.test_frag()

```

dpkt.ip6 module

Internet Protocol, version 6.

```

class dpkt.ip6.IP6(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Internet Protocol, version 6.
    TODO: Longer class information....
    __hdr__
        Header fields of IPv6.
    TODO.
    v
    fc
    flow
    unpack(buf)
    headers_str()
        Output extension headers in order defined in RFC1883 (except dest opts)
    classmethod set_proto(p, pktclass)
    classmethod get_proto(p)
    data
    dst
    hlim
    nxt
    plen
    src

class dpkt.ip6.IP6ExtensionHeader(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    An extension header is very similar to a 'sub-packet'. We just want to re-use all the hdr unpacking etc.

class dpkt.ip6.IP6OptsHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6ExtensionHeader
    unpack(buf)
    data
    len
    nxt

```

```
class dpkt.ip6.IP6HopOptsHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6OptsHeader

    data
    len
    nxt

class dpkt.ip6.IP6DstOptsHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6OptsHeader

    data
    len
    nxt

class dpkt.ip6.IP6RoutingHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6ExtensionHeader

    sl_bits
    unpack(buf)
    data
    len
    nxt
    rsvd_sl_bits
    segs_left
    type

class dpkt.ip6.IP6FragmentHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6ExtensionHeader

    unpack(buf)
    frag_off
    m_flag
    data
    frag_off_resv_m
    id
    nxt
    resv

class dpkt.ip6.IP6AHHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6ExtensionHeader

    unpack(buf)
    data
    len
    nxt
    resv
    seq
```

```

    spi

class dpkt.ip6.IP6ESPHeader(*args, **kwargs)
    Bases: dpkt.ip6.IP6ExtensionHeader

    unpack(buf)

    data

    seq

    spi

dpkt.ip6.test_ipg()
dpkt.ip6.test_ip6_routing_header()
dpkt.ip6.test_ip6_fragment_header()
dpkt.ip6.test_ip6_options_header()
dpkt.ip6.test_ip6_ah_header()
dpkt.ip6.test_ip6_esp_header()
dpkt.ip6.test_ip6_extension_headers()

```

dpkt.ipx module

Internetwork Packet Exchange.

```

class dpkt.ipx.IPX(*args, **kwargs)
    Bases: dpkt.Packet

    Internetwork Packet Exchange.

    TODO: Longer class information...

    __hdr__
        Header fields of IPX.

    TODO.

    data

    dst

    len

    pt

    src

    sum

    tc

```

dpkt.llc module

```

class dpkt.llc.LLC(*args, **kwargs)
    Bases: dpkt.Packet

    802.2 Logical Link Control (LLC) data communication protocol.

```

```
__hdr__ = (
    ('dsap', 'B', 0xaa), # Destination Service Access Point ('ssap', 'B', 0xaa), # Source Service Access Point
    ('ctl', 'B', 3) # Control Byte
)

is_snap
unpack(buf)
pack_hdr()
ctl
data
dsap
ssap

dpkt.llc.test_llc()
```

dpkt.loopback module

Platform-dependent loopback header.

```
class dpkt.loopback.Loopback(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Platform-dependent loopback header.
    TODO: Longer class information...

    __hdr__
        Header fields of Loopback.

    TODO.

    unpack(buf)

    data

    family
```

dpkt.mrt module

Multi-threaded Routing Toolkit.

```
class dpkt.mrt.MRTHeader(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data

    len

    subtype

    ts

    type

class dpkt.mrt.TableDump(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack(buf)
```

```

    attr_len
    data
    originated_ts
    peer_as
    peer_ip
    prefix
    prefix_len
    seq
    status
    view
class dpkt.mrt.BGP4MPMessage(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    dst_as
    dst_ip
    family
    intf
    src_as
    src_ip
class dpkt.mrt.BGP4MPMessage_32(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    dst_as
    dst_ip
    family
    intf
    src_as
    src_ip

```

dpkt.netbios module

Network Basic Input/Output System.

`dpkt.netbios.encode_name(name)`
Return the NetBIOS first-level encoded name.

`dpkt.netbios.decode_name(nbname)`
Return the NetBIOS first-level decoded nbname.

`dpkt.netbios.node_to_service_name(name_service_flags)`

```
class dpkt.netbios.NS(*args, **kwargs)
```

Bases: `dpkt.dns.DNS`

NetBIOS Name Service.

```
class Q(*args, **kwargs)
```

Bases: `dpkt.dns.Q`

cls

data

name

type

```
class NS.RR(*args, **kwargs)
```

Bases: `dpkt.dns.RR`

NetBIOS resource record.

```
unpack_rdata(buf, off)
```

cls

data

name

rdata

rlen

ttl

type

```
NS.pack_name(buf, name)
```

```
NS.unpack_name(buf, off)
```

NS.an

NS.ar

NS.data

NS.id

NS.ns

NS.op

NS.qd

```
class dpkt.netbios.Session(*args, **kwargs)
```

Bases: `dpkt.dpkt.Packet`

NetBIOS Session Service.

data

flags

len

type

```
class dpkt.netbios.Datagram(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    NetBIOS Datagram Service.

    data
    flags
    id
    len
    off
    sport
    src
    type
```

dpkt.netflow module

Cisco Netflow.

```
class dpkt.netflow.NetflowBase(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Base class for Cisco Netflow packets.
    TODO: Longer class information....

    __hdr__
        Header fields of NetflowBase.

    TODO.

    unpack(buf)

class NetflowRecordBase(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Base class for netflow v1-v7 netflow records.
    TODO: Longer class information....

    __hdr__
        Header fields of NetflowRecordBase.

    TODO.

    unpack(buf)

NetflowBase.count
NetflowBase.data
NetflowBase.sys_uptime
NetflowBase.unix_nsec
NetflowBase.unix_sec
NetflowBase.version
```

```
class dpkt.netflow.Netflow1(*args, **kwargs)
    Bases: dpkt.netflow.NetflowBase
    Netflow Version 1.
    TODO: Longer class information....

    __hdr__
        Header fields of Netflow Version 1.
    TODO.

class NetflowRecord(*args, **kwargs)
    Bases: dpkt.netflow.NetflowRecordBase
    Netflow v1 flow record.
    TODO: Longer class information....

    __hdr__
        Header fields of Netflow Version 1 flow record.
    TODO.

    bytes_sent
    data
    dst_addr
    dst_port
    end_time
    input_iface
    ip_proto
    next_hop
    output_iface
    pad1
    pad2
    pad3
    pkts_sent
    reserved
    src_addr
    src_port
    start_time
    tcp_flags
    tos

Netflow1.count
Netflow1.data
Netflow1.sys_uptime
Netflow1.unix_nsec
```

```

Netflow1.unix_sec
Netflow1.version
class dpkt.netflow.Netflow5(*args, **kwargs)
    Bases: dpkt.netflow.NetflowBase
    Netflow Version 5.
    TODO: Longer class information....
    __hdr__
        Header fields of Netflow Version 5.
    TODO.
class NetflowRecord(*args, **kwargs)
    Bases: dpkt.netflow.NetflowRecordBase
    Netflow v5 flow record.
    TODO: Longer class information....
    __hdr__
        Header fields of Netflow Version 5 flow record.
    TODO.
    bytes_sent
    data
    dst_addr
    dst_as
    dst_mask
    dst_port
    end_time
    input_iface
    ip_proto
    next_hop
    output_iface
    pad1
    pad2
    pkts_sent
    src_addr
    src_as
    src_mask
    src_port
    start_time
    tcp_flags
    tos

```

Netflow5.**count**
Netflow5.**data**
Netflow5.**engine_id**
Netflow5.**engine_type**
Netflow5.**flow_sequence**
Netflow5.**reserved**
Netflow5.**sys_uptime**
Netflow5.**unix_nsec**
Netflow5.**unix_sec**
Netflow5.**version**

class dpkt.netflow.**Netflow6**(*args, **kwargs)

Bases: *dpkt.netflow.NetflowBase*

Netflow Version 6.

XXX - unsupported by Cisco, but may be found in the field. TODO: Longer class information....

__hdr__

Header fields of Netflow Version 6.

TODO.

class NetflowRecord(*args, **kwargs)

Bases: dpkt.netflow.NetflowRecordBase

Netflow v6 flow record.

TODO: Longer class information....

__hdr__

Header fields of Netflow Version 6 flow record.

TODO.

bytes_sent

data

dst_addr

dst_as

dst_mask

dst_port

end_time

in_encaps

input_iface

ip_proto

next_hop

out_encaps

output_iface

```

    pad1
    peer_nextthop
    pkts_sent
    src_addr
    src_as
    src_mask
    src_port
    start_time
    tcp_flags
    tos
Netflow6.count
Netflow6.data
Netflow6.engine_id
Netflow6.engine_type
Netflow6.flow_sequence
Netflow6.reserved
Netflow6.sys_uptime
Netflow6.unix_nsec
Netflow6.unix_sec
Netflow6.version
class dpkt.netflow.Netflow7(*args, **kwargs)
    Bases: dpkt.netflow.NetflowBase
    Netflow Version 7.
    TODO: Longer class information....
    __hdr__
        Header fields of Netflow Version 7.
    TODO.
class NetflowRecord(*args, **kwargs)
    Bases: dpkt.netflow.NetflowRecordBase
    Netflow v6 flow record.
    TODO: Longer class information....
    __hdr__
        Header fields of Netflow Version 6 flow record.
    TODO.
    bytes_sent
    data
    dst_addr

```

```
dst_as
dst_mask
dst_port
end_time
flags
input_iface
ip_proto
next_hop
output_iface
pad2
pkts_sent
router_sc
src_addr
src_as
src_mask
src_port
start_time
tcp_flags
tos

Netflow7.count
Netflow7.data
Netflow7.flow_sequence
Netflow7.reserved
Netflow7.sys_uptime
Netflow7.unix_nsec
Netflow7.unix_sec
Netflow7.version

dpkt.netflow.test_net_flow_v1_pack()
dpkt.netflow.test_net_flow_v1_unpack()
dpkt.netflow.test_net_flow_v5_pack()
dpkt.netflow.test_net_flow_v5_unpack()
```

dpkt.ntp module

Network Time Protocol.

```

class dpkt.ntp.NTP(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Network Time Protocol.
    TODO: Longer class information....

    __hdr__
        Header fields of NTP.

    TODO.

    v
    li
    mode
    data
    delay
    dispersion
    flags
    id
    interval
    originate_time
    precision
    receive_time
    stratum
    transmit_time
    update_time

dpkt.ntp.test_ntp_pack()
dpkt.ntp.test_ntp_unpack()

```

dpkt.ospf module

Open Shortest Path First.

```

class dpkt.ospf.OSPF(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Open Shortest Path First.
    TODO: Longer class information....

    __hdr__
        Header fields of OSPF.

    TODO.

    area
    atype
    auth

```

data
len
router
sum
type
v

dpkt.pcap module

Libpcap file format.

```
class dpkt.pcap.PktHdr(*args, **kwargs)
    Bases: dpkt.pkt.Packet
```

pcap packet header.

TODO: Longer class information....

```
__hdr__
    Header fields of pcap header.
```

TODO.

caplen

data

len

tv_sec

tv_usec

```
class dpkt.pcap.LEPktHdr(*args, **kwargs)
    Bases: dpkt.pcap.PktHdr
```

caplen

data

len

tv_sec

tv_usec

```
class dpkt.pcap.FileHdr(*args, **kwargs)
    Bases: dpkt.pkt.Packet
```

pcap file header.

TODO: Longer class information....

```
__hdr__
    Header fields of pcap file header.
```

TODO.

data

linktype

magic

```

    sigfigs
    snaplen
    thiszone
    v_major
    v_minor
class dpkt.pcap.LEFileHdr(*args, **kwargs)
    Bases: dpkt.pcap.FileHdr
    data
    linktype
    magic
    sigfigs
    snaplen
    thiszone
    v_major
    v_minor
class dpkt.pcap.Writer(fileobj, snaplen=1500, linktype=1, nano=False)
    Bases: object
    Simple pcap dumpfile writer.
    TODO: Longer class information....
    __hdr__
        Header fields of simple pcap dumpfile writer.
    TODO.
    writepkt(pkt, ts=None)
    close()
class dpkt.pcap.Reader(fileobj)
    Bases: object
    Simple pypcap-compatible pcap file reader.
    TODO: Longer class information....
    __hdr__
        Header fields of simple pypcap-compatible pcap file reader.
    TODO.
    fd
    fileno()
    datalink()
    setfilter(value, optimize=1)
    readpkts()

```

dispatch (*cnt*, *callback*, **args*)

Collect and process packets with a user callback.

Return the number of packets processed, or 0 for a savefile.

Arguments:

cnt – number of packets to process; or 0 to process all packets until EOF

callback – function with (timestamp, pkt, **args*) prototype **args* – optional arguments passed to callback on execution

loop (*callback*, **args*)

`dpkt.pcap.test_pcap_endian()`

`dpkt.pcap.test_reader()`

`dpkt.pcap.test_writer_precision()`

dpkt.pim module

Protocol Independent Multicast.

class `dpkt.pim.PIM` (**args*, ***kwargs*)

Bases: `dpkt.dpkt.Packet`

Protocol Independent Multicast.

TODO: Longer class information....

__hdr__

Header fields of PIM.

TODO.

v

type

data

rsvd

sum

`dpkt.pim.test_pim()`

dpkt.pmap module

Portmap / rpcbind.

class `dpkt.pmap.Pmap` (**args*, ***kwargs*)

Bases: `dpkt.dpkt.Packet`

Portmap / rpcbind.

TODO: Longer class information....

__hdr__

Header fields of Pmap.

TODO.

data

port
 prog
 prot
 vers

dpkt.ppp module

Point-to-Point Protocol.

class dpkt.ppp.**PPP** (*args, **kwargs)
 Bases: *dpkt.dpkt.Packet*

Point-to-Point Protocol.

TODO: Longer class information....

__hdr__
 Header fields of PPP.

TODO.

classmethod **set_p** (p, pktclass)

classmethod **get_p** (p)

unpack (buf)

pack_hdr ()

addr

cntrl

data

p

dpkt.ppp.**test_ppp** ()

dpkt.ppp.**test_ppp_short** ()

dpkt.ppp.**test_packing** ()

dpkt.pppoe module

PPP-over-Ethernet.

class dpkt.pppoe.**PPPoE** (*args, **kwargs)
 Bases: *dpkt.dpkt.Packet*

PPP-over-Ethernet.

TODO: Longer class information....

__hdr__
 Header fields of PPPoE.

TODO.

v

type

```
    unpack (buf)

    code

    data

    len

    session

class dpkt.pppoe.PPP (*args, **kwargs)
    Bases: dpkt.ppp.PPP

    unpack (buf)

    pack_hdr ()

    data

    p

dpkt.pppoe.test_pppoe_discovery ()
dpkt.pppoe.test_pppoe_session ()
dpkt.pppoe.test_ppp_packing ()
dpkt.pppoe.test_ppp_short ()
```

dpkt.qq module

```
class dpkt.qq.QQBasicPacket (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    command

    data

    header_type

    qqNum

    sequence

    source

class dpkt.qq.QQ3Packet (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    command

    data

    header_type

    sequence

    source

    unknown1

    unknown10

    unknown11

    unknown12

    unknown13
```

`unknown2``unknown3``unknown4``unknown5``unknown6``unknown7``unknown8``unknown9`

```
class dpkt.qq.QQ5Packet (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

`command``data``header_type``qqNum``sequence``source``unknown`

dpkt.radiotap module

Radiotap

```
class dpkt.radiotap.Radiotap (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

Radiotap.

TODO: Longer class information...

`__hdr__`

Header fields of Radiotap.

TODO.

`tsft_present``flags_present``rate_present``channel_present``fhss_present``ant_sig_present``ant_noise_present``lock_qual_present``tx_attn_present``db_tx_attn_present`

```
dbm_tx_power_present
ant_present
db_ant_sig_present
db_ant_noise_present
rx_flags_present
chanplus_present
ext_present
unpack (buf)

class Antenna (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    index

class Radiotap.AntennaNoise (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    db

class Radiotap.AntennaSignal (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    db

class Radiotap.Channel (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    flags
    freq

class Radiotap.FHSS (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    pattern
    set

class Radiotap.Flags (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    fcs
    data
    val

class Radiotap.LockQuality (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
```

```

    val
Radiotap.data
Radiotap.length
Radiotap.pad
Radiotap.present_flags
Radiotap.version
class Radiotap.RxFlags (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    val
class Radiotap.Rate (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    val
class Radiotap.TSFT (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    usecs
class Radiotap.TxAttenuation (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    val
class Radiotap.DbTxAttenuation (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    db
class Radiotap.DbAntennaNoise (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    db
class Radiotap.DbAntennaSignal (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    db
class Radiotap.DbmTxPower (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    dbm
dpkt.radiotap.test_Radiotap()

```

```
dpkt.radiotap.test_fcs()
```

dpkt.radius module

Remote Authentication Dial-In User Service.

```
class dpkt.radius.RADIUS(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Remote Authentication Dial-In User Service.
    TODO: Longer class information....

    __hdr__
        Header fields of RADIUS.

    TODO.

    attrs = ''
    unpack(buf)

    auth
    code
    data
    id
    len

dpkt.radius.parse_attrs(buf)
    Parse attributes buffer into a list of (type, data) tuples.
```

dpkt.rfb module

Remote Framebuffer Protocol.

```
class dpkt.rfb.RFB(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Remote Framebuffer Protocol.
    TODO: Longer class information....

    __hdr__
        Header fields of RADIUS.

    TODO.

    data
    type

class dpkt.rfb.SetPixelFormat(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    pad
    pixel_fmt
```

```
class dpkt.rfb.SetEncodings (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    num_encodings
    pad

class dpkt.rfb.FramebufferUpdateRequest (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    height
    incremental
    width
    x_position
    y_position

class dpkt.rfb.KeyEvent (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    down_flag
    key
    pad

class dpkt.rfb.PointerEvent (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    button_mask
    data
    x_position
    y_position

class dpkt.rfb.FramebufferUpdate (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    num_rects
    pad

class dpkt.rfb.SetColourMapEntries (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    first_colour
    num_colours
    pad

class dpkt.rfb.CutText (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
```

data
length
pad

dpkt.rip module

Routing Information Protocol.

```
class dpkt.rip.RIP (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Routing Information Protocol.
    TODO: Longer class information....
    __hdr__
        Header fields of RIP.
    TODO.
    unpack (buf)
    cmd
    data
    rsvd
    v

class dpkt.rip.RTE (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    addr
    data
    family
    metric
    next_hop
    route_tag
    subnet

class dpkt.rip.Auth (*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    auth
    data
    rsvd
    type

dpkt.rip.test_rtp_pack()
dpkt.rip.test_rtp_unpack()
```

dpkt.rpc module

Remote Procedure Call.

```
class dpkt.rpc.RPC (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

Remote Procedure Call.

TODO: Longer class information....

```
__hdr__
```

Header fields of RPC.

TODO.

```
class Auth (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

```
unpack (buf)
```

```
data
```

```
flavor
```

```
class RPC.Call (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

```
unpack (buf)
```

```
data
```

```
proc
```

```
prog
```

```
rpcvers
```

```
vers
```

```
class RPC.Reply (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

```
class Accept (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

```
unpack (buf)
```

```
data
```

```
stat
```

```
class RPC.Reply.Reject (*args, **kwargs)
```

Bases: *dpkt.dpkt.Packet*

```
unpack (buf)
```

```
data
```

```
stat
```

```
RPC.Reply.unpack (buf)
```

```
RPC.Reply.data
```

```
RPC.Reply.stat
```

```
RPC.unpack (buf)
```

RPC.data

RPC.dir

RPC.xid

dpkt.rpc.unpack_xdrlist(*cls, buf*)

dpkt.rpc.pack_xdrlist(**args*)

dpkt.rtp module

Real-Time Transport Protocol.

class dpkt.rtp.RTP(**args, **kwargs*)

Bases: *dpkt.dpkt.Packet*

Real-Time Transport Protocol.

TODO: Longer class information....

__hdr__

Header fields of RTP.

TODO.

csrc = ''

version

p

x

cc

m

data

seq

ssrc

ts

pt

unpack(*buf*)

dpkt.rtp.test_rtp()

dpkt.rx module

Rx Protocol.

class dpkt.rx.Rx(**args, **kwargs*)

Bases: *dpkt.dpkt.Packet*

Rx Protocol.

TODO: Longer class information....

__hdr__

Header fields of Rx.

`TODO.`
`call`
`cid`
`data`
`epoch`
`flags`
`security`
`seq`
`serial`
`service`
`status`
`sum`
`type`

dpkt.sccp module

Cisco Skinny Client Control Protocol.

```
class dpkt.sccp.ActivateCallPlane(*args, **kwargs)
    Bases: dpkt.Packet
```

`data`
`line_instance`

```
class dpkt.sccp.CallInfo(*args, **kwargs)
    Bases: dpkt.Packet
```

`call_id`
`call_type`
`called_party`
`called_party_name`
`calling_party`
`calling_party_name`
`data`
`line_instance`
`orig_called_party`
`orig_called_party_name`

```
class dpkt.sccp.CallState(*args, **kwargs)
    Bases: dpkt.Packet
```

`call_id`
`call_state`
`data`

```
    line_instance

class dpkt.sccp.ClearPromptStatus(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    call_id

    data

    line_instance

class dpkt.sccp.CloseReceiveChannel(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    conference_id

    data

    passthrupty_id

class dpkt.sccp.DisplayPromptStatus(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    call_id

    data

    display_msg

    line_instance

    msg_timeout

class dpkt.sccp.DisplayText(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data

    display_msg

class dpkt.sccp.KeypadButton(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    button

    data

class dpkt.sccp.OpenReceiveChannel(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    conference_id

    data

    echo_cancel_type

    g723_bitrate

    ms_packet

    passthrupty_id

    payload_capability

class dpkt.sccp.OpenReceiveChannelAck(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    channel_status
```

```

    data
    ip
    passthrupty_id
    port
class dpkt.sccp.SelectStartKeys(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    call_id
    data
    line_id
    softkey_map
    softkey_set
class dpkt.sccp.SetLamp(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    lamp_mode
    stimulus
    stimulus_instance
class dpkt.sccp.SetSpeakerMode(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    speaker
class dpkt.sccp.StartMediaTransmission(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    conference_id
    data
    g723_bitrate
    max_frames_per_pkt
    ms_packet
    passthrupty_id
    payload_capability
    precedence
    remote_ip
    remote_port
    silence_suppression
class dpkt.sccp.StartTone(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    data
    tone

```

```
class dpkt.sccp.StopMediaTransmission(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    conference_id
    data
    passthru_party_id

class dpkt.sccp.SCCP(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Cisco Skinny Client Control Protocol.
    TODO: Longer class information....

    __hdr__
        Header fields of SCCP.

    TODO.

    unpack(buf)

    data
    len
    msg
    msgid
    rsvd
```

dpkt.sctp module

Stream Control Transmission Protocol.

```
class dpkt.sctp.SCTP(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    Stream Control Transmission Protocol.
    TODO: Longer class information....

    __hdr__
        Header fields of SCTP.

    TODO.

    unpack(buf)

    data
    dport
    sport
    sum
    vtag

class dpkt.sctp.Chunk(*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack(buf)

    data
```

flags**len****type**`dpkt.sctp.test_sctp_pack()``dpkt.sctp.test_sctp_unpack()`

dpkt.sip module

Session Initiation Protocol.

class `dpkt.sip.Request(*args, **kwargs)`Bases: `dpkt.http.Request`

SIP request.

TODO: Longer class information....

__hdr__

Header fields of SIP request.

TODO.**class** `dpkt.sip.Response(*args, **kwargs)`Bases: `dpkt.http.Response`

SIP response.

TODO: Longer class information....

__hdr__

Header fields of SIP response.

TODO.

dpkt.sll module

Linux libpcap “cooked” capture encapsulation.

class `dpkt.sll.SLL(*args, **kwargs)`Bases: `dpkt.dpkt.Packet`

Linux libpcap “cooked” capture encapsulation.

TODO: Longer class information....

__hdr__

Header fields of SLL.

TODO.**unpack** (*buf*)**data****ethtype****hdr****hlen****hrd**

type

`dpkt.sll.test_sll()`

dpkt.smb module

Server Message Block.

class `dpkt.smb.SMB(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

Server Message Block.

TODO: Longer class information....

```
__hdr__ = [
    ('proto', '4s', b'\x00SMB'), ('cmd', 'B', 0), ('status', 'I', SMB_STATUS_SUCCESS), ('flags', 'B', 0),
    ('flags2', 'H', 0), ('_pidhi', 'H', 0), ('security', '8s', b''), ('rsvd', 'H', 0), ('tid', 'H', 0), ('_pidlo', 'H', 0),
    ('uid', 'H', 0), ('mid', 'H', 0)
```

```
]
```

pid

cmd

data

flags

flags2

mid

proto

rsvd

security

status

tid

uid

`dpkt.smb.test_smb()`

dpkt.snoop module

Snoop file format.

class `dpkt.snoop.PktHdr(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

snoop packet header.

TODO: Longer class information....

```
__hdr__
```

Header fields of snoop packet header.

TODO.

cum_drops

```

    data
    incl_len
    orig_len
    rec_len
    ts_sec
    ts_usec

class dpkt.snoop.FileHdr(*args, **kwargs)
    Bases: dpkt.Packet
    snoop file header.
    TODO: Longer class information....

    __hdr__
        Header fields of snoop file header.

    TODO.

    data
    linktype
    magic
    v

class dpkt.snoop.Writer(fileobj, linktype=4)
    Bases: object
    Simple snoop dumpfile writer.
    TODO: Longer class information....

    TODO.

    writepkt(pkt, ts=None)
    close()

class dpkt.snoop.Reader(fileobj)
    Bases: object
    Simple pypcap-compatible snoop file reader.
    TODO: Longer class information....

    TODO.

    fileno()
    datalink()
    setfilter(value, optimize=1)
    readpkts()
    dispatch(cnt, callback, *args)
    loop(callback, *args)

```

dpkt.ssl module

Secure Sockets Layer / Transport Layer Security.

class `dpkt.ssl.SSL2(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

unpack (*buf*)

data

len

msg

pad

class `dpkt.ssl.TLS(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

unpack (*buf*)

data

len

type

version

`dpkt.ssl.parse_variable_array` (*buf*, *lenbytes*)

Parse an array described using the ‘Type name<x..y>’ syntax from the spec Read a length at the start of buf, and returns that many bytes after, in a tuple with the TOTAL bytes consumed (including the size). This does not check that the array is the right length for any given datatype.

`dpkt.ssl.parse_extensions` (*buf*)

Parse TLS extensions in passed buf. Returns an ordered list of extension tuples with ordinal extension type as first value and extension data as second value. Passed buf must start with the 2-byte extensions length TLV. <http://www.iana.org/assignments/tls-extensiontype-values/tls-extensiontype-values.xhtml>

exception `dpkt.ssl.SSL3Exception`

Bases: `exceptions.Exception`

class `dpkt.ssl.TLSRecord(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

SSLv3 or TLSv1+ packet.

In addition to the fields specified in the header, there are compressed and decrypted fields, indicating whether, in the language of the spec, this is a TLSPlaintext, TLSCompressed, or TLSCiphertext. The application will have to figure out when it’s appropriate to change these values.

length

unpack (*buf*)

data

type

version

class `dpkt.ssl.TLSChangeCipherSpec(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

ChangeCipherSpec message is just a single byte with value 1

```

    data
    type
class dpkt.ssl.TLSAppData
    Bases: str

    As far as TLSRecord is concerned, AppData is just an opaque blob.

class dpkt.ssl.TLSAlert (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data
    description
    level

class dpkt.ssl.TLSHelloRequest (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data

class dpkt.ssl.TLSClientHello (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    data
    random
    version

class dpkt.ssl.TLSServerHello (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    data
    random
    version

class dpkt.ssl.TLSCertificate (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    unpack (buf)

    data

class dpkt.ssl.TLSUnknownHandshake (*args, **kwargs)
    Bases: dpkt.dpkt.Packet

    data

dpkt.ssl.TLSServerKeyExchange
    alias of TLSUnknownHandshake

dpkt.ssl.TLSCertificateRequest
    alias of TLSUnknownHandshake

dpkt.ssl.TLSServerHelloDone
    alias of TLSUnknownHandshake

dpkt.ssl.TLSCertificateVerify
    alias of TLSUnknownHandshake

```

`dpkt.ssl.TLSClientKeyExchange`
alias of *TLSUnknownHandshake*

`dpkt.ssl.TLSFinished`
alias of *TLSUnknownHandshake*

class `dpkt.ssl.TLSHandshake` (*args, **kwargs)
Bases: *dpkt.dpkt.Packet*

A TLS Handshake message

This goes for all messages encapsulated in the Record layer, but especially important for handshakes and app data: A message may be spread across a number of TLSRecords, in addition to the possibility of there being more than one in a given Record. You have to put together the contents of TLSRecord's yourself.

unpack (buf)

length

data

length_bytes

type

class `dpkt.ssl.SSLFactory`
Bases: *object*

`dpkt.ssl.tls_multi_factory` (buf)
Attempt to parse one or more TLSRecord's out of buf

Parameters **buf** – string containing SSL/TLS messages. May have an incomplete record on the end

Returns

[TLSRecord] int, total bytes consumed, != len(buf) if an incomplete record was left at the end.

Raises SSL3Exception.

class `dpkt.ssl.TestTLS`
Bases: *object*

Test basic TLS functionality. Test that each TLSRecord is correctly discovered and added to TLS.records

classmethod `setup_class` ()

test_records_length ()

test_record_type ()

test_record_version ()

class `dpkt.ssl.TestTLSRecord`
Bases: *object*

Test basic TLSRecord functionality For this test, the contents of the record doesn't matter, since we're not parsing the next layer.

classmethod `setup_class` ()

test_content_type ()

test_version ()

test_length ()

```

    test_data()
    test_initial_flags()
    test_repack()
    test_total_length()
    test_raises_need_data_when_buf_is_short()

class dpkt.ssl.TestTLSChangeCipherSpec
    Bases: object
    It's just a byte. This will be quick, I promise
    classmethod setup_class()
    test_pares()
    test_total_length()

class dpkt.ssl.TestTLSAppData
    Bases: object
    AppData is basically just a string
    test_value()

class dpkt.ssl.TestTLSHandshake
    Bases: object
    classmethod setup_class()
    test_created_inside_message()
    test_length()
    test_raises_need_data()

class dpkt.ssl.TestClientHello
    Bases: object
    This data is extracted from and verified by Wireshark
    classmethod setup_class()
    test_client_hello_constructed()
        Make sure the correct class was constructed
    test_client_random_correct()
    test_cipher_suite_length()
    test_session_id()
    test_compression_methods()
    test_total_length()

class dpkt.ssl.TestServerHello
    Bases: object
    Again, from Wireshark
    classmethod setup_class()
    test_constructed()
    test_random_correct()

```

```
test_cipher_suite()
test_total_length()
class dpkt.ssl.TestTLSCertificate
    Bases: object
    We use a 2016 certificate record from iana.org as test data.
    classmethod setup_class()
    test_num_certs()
class dpkt.ssl.TestTLSMultiFactory
    Bases: object
    Made up test data
    classmethod setup_class()
    test_num_messages()
    test_bytes_parsed()
    test_first_msg_data()
    test_second_msg_data()
```

dpkt.ssl_ciphersuites module

Nicely formatted cipher suite definitions for TLS

A list of cipher suites in the form of CipherSuite objects. These are supposed to be immutable; don't mess with them.

```
class dpkt.ssl_ciphersuites.CipherSuite(code, kx, auth, cipher, mode, mac, name=None, encoding=None)
```

Bases: object

Encapsulates a cipher suite.

Members/args: * code: two-byte ID code, as int * kx: key exchange algorithm, e.g. 'RSA' or 'DHE' * auth: authentication algorithm, e.g. 'RSA' or 'DSS' * cipher: stream or block cipher algorithm, e.g. 'AES_128' * mode: mode of operation for block ciphers, e.g. 'CBC' or 'GCM' * mac: message authentication code algorithm, e.g. 'MD5' or 'SHA256' * name: cipher suite name as defined in the RFCs,

e.g. 'TLS_RSA_WITH_RC4_40_MD5', can be generated by default from the other parameters

•encoding: encoding algorithm, defaults to cipher+mode

Additional members: * kx_auth: kx+auth algorithm, as 'KeyExchangeAlgorithm' in RFCs

kx

auth

kx_auth

encoding

name

MAC_SIZES = {'SHA': 20, 'SHA256': 32, 'SHA384': 48, 'MD5': 16}

BLOCK_SIZES = {'AES_128': 16, 'AES_256': 16}

mac_size

In bytes. Default to 0.

block_size

In bytes. Default to 1.

`dpkt.ssl_ciphersuites.BY_NAME(name)`

class `dpkt.ssl_ciphersuites.TestCipherSuites`

Bases: `object`

test_kx()

test_auth()

test_name()

dpkt.stp module

Spanning Tree Protocol.

class `dpkt.stp.STP(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

Spanning Tree Protocol.

TODO: Longer class information....

__hdr__

Header fields of STP.

TODO.

age

max_age

bridge_id

data

flags

hello

port_id

proto_id

root_id

root_path

type

v

fd

`dpkt.stp.test_stp()`

dpkt.stun module

Simple Traversal of UDP through NAT.

class `dpkt.stun.STUN` (**args*, ***kwargs*)

Bases: *dpkt.dpkt.Packet*

Simple Traversal of UDP through NAT.

STUN - RFC 3489 <http://tools.ietf.org/html/rfc3489> Each packet has a 20 byte header followed by 0 or more attribute TLVs.

__hdr__

Header fields of STUN.

TODO.

data

len

type

xid

`dpkt.stun.tlv` (*buf*)

`dpkt.stun.parse_attrs` (*buf*)

Parse STUN.data buffer into a list of (attribute, data) tuples.

`dpkt.stun.test_stun_response` ()

`dpkt.stun.test_stun_padded` ()

dpkt.tcp module

Transmission Control Protocol.

class `dpkt.tcp.TCP` (**args*, ***kwargs*)

Bases: *dpkt.dpkt.Packet*

Transmission Control Protocol.

TODO: Longer class information....

__hdr__

Header fields of TCP.

TODO.

opts = ''

off

unpack (*buf*)

ack

data

dport

flags

seq

sport**sum****urp****win**`dpkt.tcp.parse_opts(buf)`

Parse TCP option buffer into a list of (option, data) tuples.

`dpkt.tcp.test_parse_opts()``dpkt.tcp.test_offset()`

dpkt.telnet module

Telnet.

`dpkt.telnet.strip_options(buf)`

Return a list of lines and dict of options from telnet data.

`dpkt.telnet.test_telnet()`

dpkt.tftp module

Trivial File Transfer Protocol.

class `dpkt.tftp.TFTP(*args, **kwargs)`Bases: `dpkt.dpkt.Packet`

Trivial File Transfer Protocol.

TODO: Longer class information....

__hdr__

Header fields of TFTP.

TODO.**unpack** (*buf*)**data****opcode**`dpkt.tftp.test_op_rrq()``dpkt.tftp.test_op_data()``dpkt.tftp.test_op_err()`

dpkt.tns module

Transparent Network Substrate.

class `dpkt.tns.TNS(*args, **kwargs)`Bases: `dpkt.dpkt.Packet`

Transparent Network Substrate.

TODO: Longer class information....

```
__hdr__  
    Header fields of TNS.
```

```
TODO.
```

```
unpack (buf)
```

```
data
```

```
hdrsum
```

```
length
```

```
msg
```

```
pktsum
```

```
rsvd
```

```
type
```

```
dpkt.tns.test_tns()
```

dpkt.tpkt module

ISO Transport Service on top of the TCP (TPKT).

```
class dpkt.tpkt.TPKT (*args, **kwargs)
```

```
    Bases: dpkt.dpkt.Packet
```

ISO Transport Service on top of the TCP (TPKT).

TODO: Longer class information....

```
__hdr__  
    Header fields of TPKT.
```

```
TODO.
```

```
data
```

```
len
```

```
rsvd
```

```
v
```

dpkt.udp module

User Datagram Protocol.

```
class dpkt.udp.UDP (*args, **kwargs)
```

```
    Bases: dpkt.dpkt.Packet
```

User Datagram Protocol.

TODO: Longer class information....

```
__hdr__  
    Header fields of UDP.
```

```
TODO.
```

```
data
```

```

dport
sport
sum
ulen

```

dpkt.vrrp module

Virtual Router Redundancy Protocol.

```

class dpkt.vrrp.VRRP(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Virtual Router Redundancy Protocol.
    TODO: Longer class information....
    __hdr__
        Header fields of VRRP.
    TODO.
    addrs = ()
    auth = ''
    v
    type
    unpack(buf)
    advtime
    atype
    count
    data
    priority
    sum
    vrid

```

```
dpkt.vrrp.test_vrrp()
```

dpkt.yahoo module

Yahoo Messenger.

```

class dpkt.yahoo.YHOO(*args, **kwargs)
    Bases: dpkt.dpkt.Packet
    Yahoo Messenger.
    TODO: Longer class information....
    __hdr__
        Header fields of Yahoo Messenger.
    TODO.

```

connid

data

length

magic

nick1

nick2

service

type

unknown

version

class `dpkt.yahoo.YMSG(*args, **kwargs)`

Bases: `dpkt.dpkt.Packet`

data

length

type

unknown1

unknown2

version

Authors

Original author

Dug Song <dugsong@monkey.org>

Contributors

Timur Alperovich <timuralp@umich.edu> radiotap module

Nic Bellamy <nic.bellamy@vadacom.co.nz> HTTP header parsing fix

the grugq <thegrugq@gmail.com> better RTP module

David Helder <dhelder@gizmolabs.org> bug fixes

Przemysław Karwasiecki <karwas@gmail.com> TABLE_DUMP in MRT module

Reza Lotun <rilotun@cs.ubc.ca> MetaPacket cleanup

Jeff Nathan <jeff@snort.org> bug fixes

Tim Newsham <newsham@lava.net> IPv6 bugfixing and improvements

keisuke.nishimoto@gmail.com Snoop file parser

Jon Oberheide <jon@oberheide.org> STUN, H.225, TPKT, NTP, RIP, Diameter, SCTP, BGP, MRT, RX modules

plotnikoff@gmail.com handle dynamic imports from py2exe/freeze.py/zipped egg packages

simdream@gmail.com handle multiple cookie values in HTTP

Owen Stephens <owen@owenstephens.co.uk> IP6 extension header support

Robert Stone <otaku@monkey.org> Netflow and QQ modules

Thomas Taranowski <thomastaranowski@yahoo.com> dnet IP checksum bug on i386

Jirka Vejraska bug fixes

Tim Yardley <yardley@gmail.com> DHCP definitions

obormot <oscar.ibatullin@gmail.com> pcapng module, Packet repr improvements

Kyle Keppler <kyle.keppler@gmail.com> Python 3 port

Hao Sun <sunhao2013@gmail.com> Python 3 port

If you want to contribute to dpkt, see *Contributing*.

Changelog

Development plans

Current plans

- Be Awesome

Future plans

- Maintain the Awesome

Contributing

Report a Bug or Make a Feature Request

Please go to the GitHub Issues page: <https://github.com/kbandla/dpkt/issues>.

Checkout the Code

```
git clone https://github.com/kbandla/dpkt.git
```

Become a Developer

dpkt uses the ‘GitHub Flow’ model: [GitHub Flow](#)

- To work on something new, create a descriptively named branch off of master (ie: my-awesome)
- Commit to that branch locally and regularly push your work to the same named branch on the server
- When you need feedback or help, or you think the branch is ready for merging, open a pull request
- After someone else has reviewed and signed off on the feature, they or you can merge it into master

New Feature or Bug

```
$ git checkout -b my-awesome  
$ git push -u origin my-awesome  
$ <code for a bit>; git push  
$ <code for a bit>; git push  
$ tox (this will run all the tests)
```

- Go to github and hit ‘New pull request’
- Someone reviews it and says ‘AOK’
- Merge the pull request (green button)

License

BSD 3-Clause License, as the upstream project

Notes

PyPI Release How-To

Notes and information on how to do the PyPI release for the dpkt project.

Package Requirements

- pip install tox
- pip install wheel

Tox Background

Tox will install the dpkt package into a blank virtualenv and then execute all the tests against the newly installed package. So if everything goes okay, you know the pypi package installed fine and the tests (which pull from the installed dpkt package) also ran okay.

Create the PyPI Release

```
$ cd dpkt
$ tox
$ vi dpkt/__init__.py and bump the version
$ python setup.py release
<enter your pypi password>
```

If everything above went okay...

```
$ git add dpkt/__init__.py
$ git commit -m "dpkt version 1.8.7 (or whatever)"
$ git tag v1.8.7 (or whatever)
$ git push --tags
$ git push
```

Git Releases (discussion)

You can also do a ‘release’ on GitHub (the tags above are perfect for that). In general this is discouraged, people should always do a `$pip install dpkt`. If people want older releases they can do a `$pip install dpkt==<old version>`. Providing tarballs/zip file on GitHub will just confuse new users and they’ll have a ‘bad experience’ when trying to deal with a tarball.

a

`dpkt.ah`, 9
`dpkt.aim`, 10
`dpkt.aoe`, 11
`dpkt.aoeata`, 11
`dpkt.aoecfg`, 12
`dpkt.arp`, 12
`dpkt.asn1`, 13

b

`dpkt.bgp`, 13

c

`dpkt.cdp`, 17
`dpkt.crc32c`, 18

d

`dpkt.decorators`, 18
`dpkt.dhcp`, 19
`dpkt.diameter`, 19
`dpkt.dns`, 20
`dpkt.dpkt`, 22
`dpkt.dtp`, 23

e

`dpkt.esp`, 23
`dpkt.ethernet`, 24
`examples.print_http_requests`, 7
`examples.print_icmp`, 6
`examples.print_packets`, 4

g

`dpkt.gre`, 25
`dpkt.gzip`, 26

h

`dpkt.h225`, 27
`dpkt.hsrp`, 28
`dpkt.http`, 28

i

`dpkt.icmp`, 29
`dpkt.icmp6`, 31
`dpkt.ieee80211`, 32
`dpkt.igmp`, 37
`dpkt.ip`, 38
`dpkt.ip6`, 39
`dpkt.ipx`, 41

l

`dpkt.llc`, 41
`dpkt.loopback`, 42

m

`dpkt.mrt`, 42

n

`dpkt.netbios`, 43
`dpkt.netflow`, 45
`dpkt.ntp`, 50

o

`dpkt.ospf`, 51

p

`dpkt.pcap`, 52
`dpkt.pim`, 54
`dpkt.pmap`, 54
`dpkt.ppp`, 55
`dpkt.pppoe`, 55

q

`dpkt.qq`, 56

r

`dpkt.radiotap`, 57
`dpkt.radius`, 60
`dpkt.rfb`, 60
`dpkt.rip`, 62

`dpkt.rpc`, 63
`dpkt.rtp`, 64
`dpkt.rx`, 64

S

`dpkt.sccp`, 65
`dpkt.sctp`, 68
`dpkt.sip`, 69
`dpkt.sll`, 69
`dpkt.smb`, 70
`dpkt.snoop`, 70
`dpkt.ssl`, 72
`dpkt.ssl_ciphersuites`, 76
`dpkt.stp`, 77
`dpkt.stun`, 78

t

`dpkt.tcp`, 78
`dpkt.telnet`, 79
`dpkt.tftp`, 79
`dpkt.tns`, 79
`dpkt.tpkt`, 80

u

`dpkt.udp`, 80

v

`dpkt.vrrp`, 81

y

`dpkt.yahoo`, 81

Symbols

- `__byte_order__` (dpkt.dpkt.Packet attribute), 22
- `__hdr__` (dpkt.ah.AH attribute), 9
- `__hdr__` (dpkt.aim.FLAP attribute), 10
- `__hdr__` (dpkt.aim.SNAC attribute), 10
- `__hdr__` (dpkt.aoe.AOE attribute), 11
- `__hdr__` (dpkt.aoeata.AOEATA attribute), 11
- `__hdr__` (dpkt.aoecfg.AOECFG attribute), 12
- `__hdr__` (dpkt.arp.ARP attribute), 12
- `__hdr__` (dpkt.bgp.BGP attribute), 13
- `__hdr__` (dpkt.cdp.CDP attribute), 17
- `__hdr__` (dpkt.dhcp.DHCP attribute), 19
- `__hdr__` (dpkt.diameter.Diameter attribute), 19
- `__hdr__` (dpkt.dns.DNS attribute), 21
- `__hdr__` (dpkt.dpkt.Packet attribute), 22
- `__hdr__` (dpkt.dtp.DTP attribute), 23
- `__hdr__` (dpkt.esp.ESP attribute), 23
- `__hdr__` (dpkt.ethernet.Ethernet attribute), 24
- `__hdr__` (dpkt.gre.GRE attribute), 25
- `__hdr__` (dpkt.h225.H225 attribute), 27
- `__hdr__` (dpkt.hsrp.HSRP attribute), 28
- `__hdr__` (dpkt.http.Message attribute), 28
- `__hdr__` (dpkt.http.Request attribute), 29
- `__hdr__` (dpkt.http.Response attribute), 29
- `__hdr__` (dpkt.icmp.ICMP attribute), 29
- `__hdr__` (dpkt.icmp6.ICMP6 attribute), 31
- `__hdr__` (dpkt.ieee80211.IEEE80211 attribute), 32
- `__hdr__` (dpkt.igmp.IGMP attribute), 37
- `__hdr__` (dpkt.ip.IP attribute), 38
- `__hdr__` (dpkt.ip6.IP6 attribute), 39
- `__hdr__` (dpkt.ipx.IPX attribute), 41
- `__hdr__` (dpkt.loopback.Loopback attribute), 42
- `__hdr__` (dpkt.netflow.Netflow1 attribute), 46
- `__hdr__` (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- `__hdr__` (dpkt.netflow.Netflow5 attribute), 47
- `__hdr__` (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
- `__hdr__` (dpkt.netflow.Netflow6 attribute), 48
- `__hdr__` (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
- `__hdr__` (dpkt.netflow.Netflow7 attribute), 49
- `__hdr__` (dpkt.netflow.Netflow7.NetflowRecord attribute), 49
- `__hdr__` (dpkt.netflow.NetflowBase attribute), 45
- `__hdr__` (dpkt.netflow.NetflowBase.NetflowRecordBase attribute), 45
- `__hdr__` (dpkt.ntp.NTP attribute), 51
- `__hdr__` (dpkt.ospf.OSPF attribute), 51
- `__hdr__` (dpkt.pcap.FileHdr attribute), 52
- `__hdr__` (dpkt.pcap.PktHdr attribute), 52
- `__hdr__` (dpkt.pcap.Reader attribute), 53
- `__hdr__` (dpkt.pcap.Writer attribute), 53
- `__hdr__` (dpkt.pim.PIM attribute), 54
- `__hdr__` (dpkt.pmap.Pmap attribute), 54
- `__hdr__` (dpkt.ppp.PPP attribute), 55
- `__hdr__` (dpkt.pppoe.PPPoE attribute), 55
- `__hdr__` (dpkt.radiotap.Radiotap attribute), 57
- `__hdr__` (dpkt.radius.RADIUS attribute), 60
- `__hdr__` (dpkt.rfb.RFB attribute), 60
- `__hdr__` (dpkt.rip.RIP attribute), 62
- `__hdr__` (dpkt.rpc.RPC attribute), 63
- `__hdr__` (dpkt.rtp.RTP attribute), 64
- `__hdr__` (dpkt.rx.Rx attribute), 64
- `__hdr__` (dpkt.sccp.SCCP attribute), 68
- `__hdr__` (dpkt.sctp.SCTP attribute), 68
- `__hdr__` (dpkt.sip.Request attribute), 69
- `__hdr__` (dpkt.sip.Response attribute), 69
- `__hdr__` (dpkt.sll.SLL attribute), 69
- `__hdr__` (dpkt.snoop.FileHdr attribute), 71
- `__hdr__` (dpkt.snoop.PktHdr attribute), 70
- `__hdr__` (dpkt.stp.STP attribute), 77
- `__hdr__` (dpkt.stun.STUN attribute), 78
- `__hdr__` (dpkt.tcp.TCP attribute), 78
- `__hdr__` (dpkt.tftp.TFTP attribute), 79
- `__hdr__` (dpkt.tns.TNS attribute), 79
- `__hdr__` (dpkt.tpkt.TPKT attribute), 80
- `__hdr__` (dpkt.udp.UDP attribute), 80
- `__hdr__` (dpkt.vrrp.VRRP attribute), 81

__hdr__ (dpkt.yahoo.YHOO attribute), 81

A

aa (dpkt.dns.DNS attribute), 21
ack (dpkt.tcp.TCP attribute), 78
ack_policy (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33
ActivateCallPlane (class in dpkt.sccp), 65
add() (in module dpkt.crc32c), 18
addr (dpkt.ppp.PPP attribute), 55
addr (dpkt.rip.RTE attribute), 62
addrs (dpkt.vrrp.VRRP attribute), 81
advtime (dpkt.vrrp.VRRP attribute), 81
afi (dpkt.bgp.BGP.RouteRefresh attribute), 17
afi (dpkt.bgp.BGP.Update.Attribute.MPReachNLRI attribute), 16
afi (dpkt.bgp.BGP.Update.Attribute.MPUnreachNLRI attribute), 16
aflags (dpkt.aoeata.AOEATA attribute), 11
age (dpkt.stp.STP attribute), 77
AH (class in dpkt.ah), 9
aid (dpkt.ieee80211.IEEE80211.Assoc_Resp attribute), 34
alen (dpkt.cdp.CDP.Address attribute), 18
algorithm (dpkt.ieee80211.IEEE80211.Auth attribute), 34
an (dpkt.dns.DNS attribute), 22
an (dpkt.netbios.NS attribute), 44
ant_noise_present (dpkt.radiotap.Radiotap attribute), 57
ant_present (dpkt.radiotap.Radiotap attribute), 58
ant_sig_present (dpkt.radiotap.Radiotap attribute), 57
AOE (class in dpkt.aoe), 11
AOEATA (class in dpkt.aoeata), 11
aoeccmd (dpkt.aoeCFG.AOECFG attribute), 12
AOECFG (class in dpkt.aoeCFG), 12
app_id (dpkt.diameter.Diameter attribute), 20
ar (dpkt.dns.DNS attribute), 22
ar (dpkt.netbios.NS attribute), 44
area (dpkt.ospf.OSPF attribute), 51
ARP (class in dpkt.arp), 12
as_tuple() (dpkt.ethernet.MPLSLabel method), 24
as_tuple() (dpkt.ethernet.VLANtag8021Q method), 25
asn (dpkt.bgp.BGP.Open attribute), 14
asn (dpkt.bgp.BGP.Update.Attribute.Aggregator attribute), 15
asn (dpkt.bgp.BGP.Update.Attribute.Communities.Community attribute), 15
ast (dpkt.aim.FLAP attribute), 10
atim (dpkt.ieee80211.IEEE80211.IBSS attribute), 37
attr_len (dpkt.mrt.TableDump attribute), 43
attrs (dpkt.radius.RADIUS attribute), 60
atype (dpkt.ospf.OSPF attribute), 51
atype (dpkt.vrrp.VRRP attribute), 81
Auth (class in dpkt.rip), 62

auth (dpkt.ah.AH attribute), 9
auth (dpkt.hsrp.HSRP attribute), 28
auth (dpkt.ospf.OSPF attribute), 51
auth (dpkt.radius.RADIUS attribute), 60
auth (dpkt.rip.Auth attribute), 62
auth (dpkt.ssl_ciphersuites.CipherSuite attribute), 76
auth (dpkt.vrrp.VRRP attribute), 81
auth_seq (dpkt.ieee80211.IEEE80211.Auth attribute), 34
AVP (class in dpkt.diameter), 20

B

BGP (class in dpkt.bgp), 13
BGP.Keepalive (class in dpkt.bgp), 16
BGP.Notification (class in dpkt.bgp), 16
BGP.Open (class in dpkt.bgp), 13
BGP.Open.Parameter (class in dpkt.bgp), 14
BGP.Open.Parameter.Authentication (class in dpkt.bgp), 14
BGP.Open.Parameter.Capability (class in dpkt.bgp), 14
BGP.RouteRefresh (class in dpkt.bgp), 16
BGP.Update (class in dpkt.bgp), 14
BGP.Update.Attribute (class in dpkt.bgp), 14
BGP.Update.Attribute.Aggregator (class in dpkt.bgp), 15
BGP.Update.Attribute.ASPath (class in dpkt.bgp), 15
BGP.Update.Attribute.ASPath.ASPathSegment (class in dpkt.bgp), 15
BGP.Update.Attribute.AtomicAggregate (class in dpkt.bgp), 15
BGP.Update.Attribute.ClusterList (class in dpkt.bgp), 16
BGP.Update.Attribute.Communities (class in dpkt.bgp), 15
BGP.Update.Attribute.Communities.Community (class in dpkt.bgp), 15
BGP.Update.Attribute.Communities.ReservedCommunity (class in dpkt.bgp), 15
BGP.Update.Attribute.LocalPref (class in dpkt.bgp), 15
BGP.Update.Attribute.MPReachNLRI (class in dpkt.bgp), 16
BGP.Update.Attribute.MPReachNLRI.SNPA (class in dpkt.bgp), 16
BGP.Update.Attribute.MPUnreachNLRI (class in dpkt.bgp), 16
BGP.Update.Attribute.MultiExitDisc (class in dpkt.bgp), 15
BGP.Update.Attribute.NextHop (class in dpkt.bgp), 15
BGP.Update.Attribute.Origin (class in dpkt.bgp), 14
BGP.Update.Attribute.OriginatorID (class in dpkt.bgp), 16
BGP4MPMessage (class in dpkt.mrt), 43
BGP4MPMessage_32 (class in dpkt.mrt), 43
block_size (dpkt.ssl_ciphersuites.CipherSuite attribute), 77
BLOCK_SIZES (dpkt.ssl_ciphersuites.CipherSuite attribute), 76

- body (dpkt.http.Message attribute), 28
 - bridge_id (dpkt.stp.STP attribute), 77
 - bssid (dpkt.ieee80211.IEEE80211.Data attribute), 35
 - bssid (dpkt.ieee80211.IEEE80211.DataFromDS attribute), 35
 - bssid (dpkt.ieee80211.IEEE80211.DataToDS attribute), 35
 - bssid (dpkt.ieee80211.IEEE80211.MGMT_Frame attribute), 33
 - bufcnt (dpkt.aocfg.AOECFG attribute), 12
 - button (dpkt.sccp.KeypadButton attribute), 66
 - button_mask (dpkt.rfb.PointerEvent attribute), 61
 - BY_NAME() (in module dpkt.ssl_ciphersuites), 77
 - bytes_sent (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 - bytes_sent (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 - bytes_sent (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 - bytes_sent (dpkt.netflow.Netflow7.NetflowRecord attribute), 49
- ## C
- call (dpkt.rx.Rx attribute), 65
 - call_id (dpkt.sccp.CallInfo attribute), 65
 - call_id (dpkt.sccp.CallState attribute), 65
 - call_id (dpkt.sccp.ClearPromptStatus attribute), 66
 - call_id (dpkt.sccp.DisplayPromptStatus attribute), 66
 - call_id (dpkt.sccp.SelectStartKeys attribute), 67
 - call_state (dpkt.sccp.CallState attribute), 65
 - call_type (dpkt.sccp.CallInfo attribute), 65
 - called_party (dpkt.sccp.CallInfo attribute), 65
 - called_party_name (dpkt.sccp.CallInfo attribute), 65
 - CallInfo (class in dpkt.sccp), 65
 - calling_party (dpkt.sccp.CallInfo attribute), 65
 - calling_party_name (dpkt.sccp.CallInfo attribute), 65
 - CallState (class in dpkt.sccp), 65
 - capability (dpkt.ieee80211.IEEE80211.Assoc_Req attribute), 34
 - capability (dpkt.ieee80211.IEEE80211.Assoc_Resp attribute), 34
 - capability (dpkt.ieee80211.IEEE80211.Beacon attribute), 33
 - capability (dpkt.ieee80211.IEEE80211.Reassoc_Req attribute), 34
 - caplen (dpkt.pcap.LEPktHdr attribute), 52
 - caplen (dpkt.pcap.PktHdr attribute), 52
 - category (dpkt.ieee80211.IEEE80211.Action attribute), 34
 - cc (dpkt.rtp.RTP attribute), 64
 - CDP (class in dpkt.cdp), 17
 - CDP.Address (class in dpkt.cdp), 17
 - CDP.TLV (class in dpkt.cdp), 18
 - ch (dpkt.ieee80211.IEEE80211.DS attribute), 36
 - chaddr (dpkt.dhcp.DHCP attribute), 19
 - channel_present (dpkt.radiotap.Radiotap attribute), 57
 - channel_status (dpkt.sccp.OpenReceiveChannelAck attribute), 66
 - chanplus_present (dpkt.radiotap.Radiotap attribute), 58
 - Chunk (class in dpkt.sctp), 68
 - ciaddr (dpkt.dhcp.DHCP attribute), 19
 - cid (dpkt.rx.Rx attribute), 65
 - CipherSuite (class in dpkt.ssl_ciphersuites), 76
 - cksum() (in module dpkt.crc32c), 18
 - ClearPromptStatus (class in dpkt.sccp), 66
 - close() (dpkt.pcap.Writer method), 53
 - close() (dpkt.snoop.Writer method), 71
 - CloseReceiveChannel (class in dpkt.sccp), 66
 - cls (dpkt.dns.DNS.Q attribute), 21
 - cls (dpkt.dns.DNS.RR attribute), 21
 - cls (dpkt.netbios.NS.Q attribute), 44
 - cls (dpkt.netbios.NS.RR attribute), 44
 - cmd (dpkt.aoe.AOE attribute), 11
 - cmd (dpkt.diameter.Diameter attribute), 20
 - cmd (dpkt.rip.RIP attribute), 62
 - cmd (dpkt.smb.SMB attribute), 70
 - cmdstat (dpkt.aoeata.AOEATA attribute), 11
 - cntrl (dpkt.ppp.PPP attribute), 55
 - code (dpkt.bgp.BGP.Notification attribute), 16
 - code (dpkt.bgp.BGP.Open.Parameter.Authentication attribute), 14
 - code (dpkt.bgp.BGP.Open.Parameter.Capability attribute), 14
 - code (dpkt.diameter.AVP attribute), 20
 - code (dpkt.icmp.ICMP attribute), 30
 - code (dpkt.icmp6.ICMP6 attribute), 31
 - code (dpkt.ieee80211.IEEE80211.Action attribute), 34
 - code (dpkt.pppoe.PPPoE attribute), 56
 - code (dpkt.radius.RADIUS attribute), 60
 - command (dpkt.qq.QQ3Packet attribute), 56
 - command (dpkt.qq.QQ5Packet attribute), 57
 - command (dpkt.qq.QQBasicPacket attribute), 56
 - comment (dpkt.gzip.Gzip attribute), 26
 - compress() (dpkt.gzip.Gzip method), 26
 - compressed (dpkt.ieee80211.IEEE80211.BlockAck attribute), 32
 - conference_id (dpkt.sccp.CloseReceiveChannel attribute), 66
 - conference_id (dpkt.sccp.OpenReceiveChannel attribute), 66
 - conference_id (dpkt.sccp.StartMediaTransmission attribute), 67
 - conference_id (dpkt.sccp.StopMediaTransmission attribute), 68
 - connid (dpkt.yahoo.YHOO attribute), 81
 - control (dpkt.ieee80211.IEEE80211.QoS_Data attribute), 36
 - count (dpkt.ieee80211.IEEE80211.CF attribute), 36

count (dpkt.ieee80211.IEEE80211.TIM attribute), 37
 count (dpkt.netflow.Netflow1 attribute), 46
 count (dpkt.netflow.Netflow5 attribute), 47
 count (dpkt.netflow.Netflow6 attribute), 49
 count (dpkt.netflow.Netflow7 attribute), 50
 count (dpkt.netflow.NetflowBase attribute), 45
 count (dpkt.vrrp.VRRP attribute), 81
 cslen (dpkt.aoeCfg.AOECFG attribute), 12
 csrc (dpkt.rtp.RTP attribute), 64
 ctl (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33
 ctl (dpkt.ieee80211.IEEE80211.BlockAckReq attribute), 32
 ctl (dpkt.llc.LLC attribute), 42
 ctrl (dpkt.ieee80211.IEEE80211.TIM attribute), 37
 cum_drops (dpkt.snoop.PktHdr attribute), 70
 current_ap (dpkt.ieee80211.IEEE80211.Reassoc_Req attribute), 34
 CutText (class in dpkt.rfb), 61

D

da (dpkt.ethernet.VLANtagISL attribute), 25
 da (dpkt.ieee80211.IEEE80211.DataInterDS attribute), 35
 data (dpkt.ah.AH attribute), 9
 data (dpkt.aim.FLAP attribute), 10
 data (dpkt.aim.SNAC attribute), 10
 data (dpkt.aoe.AOE attribute), 11
 data (dpkt.aoeata.AOEATA attribute), 11
 data (dpkt.aoeCfg.AOECFG attribute), 12
 data (dpkt.arp.ARP attribute), 12
 data (dpkt.bgp.BGP attribute), 17
 data (dpkt.bgp.BGP.Notification attribute), 16
 data (dpkt.bgp.BGP.Open attribute), 14
 data (dpkt.bgp.BGP.Open.Parameter attribute), 14
 data (dpkt.bgp.BGP.Open.Parameter.Authentication attribute), 14
 data (dpkt.bgp.BGP.Open.Parameter.Capability attribute), 14
 data (dpkt.bgp.BGP.RouteRefresh attribute), 17
 data (dpkt.bgp.BGP.Update.Attribute attribute), 16
 data (dpkt.bgp.BGP.Update.Attribute.Aggregator attribute), 15
 data (dpkt.bgp.BGP.Update.Attribute.ASPath.ASPathSegment attribute), 15
 data (dpkt.bgp.BGP.Update.Attribute.Communities.Community attribute), 15
 data (dpkt.bgp.BGP.Update.Attribute.Communities.ReservedCommunity attribute), 16
 data (dpkt.bgp.BGP.Update.Attribute.LocalPref attribute), 15
 data (dpkt.bgp.BGP.Update.Attribute.MPReachNLRI attribute), 16
 data (dpkt.bgp.BGP.Update.Attribute.MPUnreachNLRI attribute), 16
 data (dpkt.bgp.BGP.Update.Attribute.MultiExitDisc attribute), 15
 data (dpkt.bgp.BGP.Update.Attribute.NextHop attribute), 15
 data (dpkt.bgp.BGP.Update.Attribute.Origin attribute), 14
 data (dpkt.bgp.BGP.Update.Attribute.OriginatorID attribute), 16
 data (dpkt.bgp.RouteGeneric attribute), 17
 data (dpkt.bgp.RouteIPv4 attribute), 17
 data (dpkt.bgp.RouteIPv6 attribute), 17
 data (dpkt.cdp.CDP attribute), 18
 data (dpkt.cdp.CDP.Address attribute), 18
 data (dpkt.cdp.CDP.TLV attribute), 18
 data (dpkt.dhcp.DHCP attribute), 19
 data (dpkt.diameter.AVP attribute), 20
 data (dpkt.diameter.Diameter attribute), 20
 data (dpkt.dns.DNS attribute), 22
 data (dpkt.dns.DNS.Q attribute), 21
 data (dpkt.dns.DNS.RR attribute), 21
 data (dpkt.dtp.DTP attribute), 23
 data (dpkt.esp.ESP attribute), 24
 data (dpkt.ethernet.Ethernet attribute), 24
 data (dpkt.ethernet.MPLSlabel attribute), 24
 data (dpkt.ethernet.VLANtag8021Q attribute), 25
 data (dpkt.ethernet.VLANtagISL attribute), 25
 data (dpkt.gre.GRE attribute), 26
 data (dpkt.gre.GRE.SRE attribute), 26
 data (dpkt.gzip.Gzip attribute), 26
 data (dpkt.gzip.GzipExtra attribute), 26
 data (dpkt.h225.H225 attribute), 27
 data (dpkt.h225.H225.IE attribute), 27
 data (dpkt.hsrp.HSRP attribute), 28
 data (dpkt.icmp.ICMP attribute), 30
 data (dpkt.icmp.ICMP.Echo attribute), 30
 data (dpkt.icmp.ICMP.ParamProbe attribute), 30
 data (dpkt.icmp.ICMP.Quench attribute), 30
 data (dpkt.icmp.ICMP.Quote attribute), 30
 data (dpkt.icmp.ICMP.Redirect attribute), 30
 data (dpkt.icmp.ICMP.TimeExceed attribute), 30
 data (dpkt.icmp.ICMP.Unreach attribute), 30
 data (dpkt.icmp6.ICMP6 attribute), 32
 data (dpkt.icmp6.ICMP6.Echo attribute), 31
 data (dpkt.icmp6.ICMP6.Error attribute), 31
 data (dpkt.icmp6.ICMP6.ParamProb attribute), 31
 data (dpkt.icmp6.ICMP6.TimeExceed attribute), 31
 data (dpkt.icmp6.ICMP6.TooBig attribute), 31
 data (dpkt.icmp6.ICMP6.Unreach attribute), 31
 data (dpkt.ieee80211.IEEE80211 attribute), 37
 data (dpkt.ieee80211.IEEE80211.ACK attribute), 33
 data (dpkt.ieee80211.IEEE80211.Action attribute), 34
 data (dpkt.ieee80211.IEEE80211.Assoc_Req attribute), 34
 data (dpkt.ieee80211.IEEE80211.Assoc_Resp attribute), 34

data (dpkt.ieee80211.IEEE80211.Auth attribute), 34
 data (dpkt.ieee80211.IEEE80211.Beacon attribute), 33
 data (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33
 data (dpkt.ieee80211.IEEE80211.BlockAckActionRequest attribute), 35
 data (dpkt.ieee80211.IEEE80211.BlockAckActionResponse attribute), 35
 data (dpkt.ieee80211.IEEE80211.BlockAckReq attribute), 32
 data (dpkt.ieee80211.IEEE80211.CF attribute), 36
 data (dpkt.ieee80211.IEEE80211.CFEnd attribute), 33
 data (dpkt.ieee80211.IEEE80211.CTS attribute), 33
 data (dpkt.ieee80211.IEEE80211.Data attribute), 35
 data (dpkt.ieee80211.IEEE80211.DataFromDS attribute), 35
 data (dpkt.ieee80211.IEEE80211.DataInterDS attribute), 35
 data (dpkt.ieee80211.IEEE80211.DataToDS attribute), 35
 data (dpkt.ieee80211.IEEE80211.Deauth attribute), 34
 data (dpkt.ieee80211.IEEE80211.Disassoc attribute), 34
 data (dpkt.ieee80211.IEEE80211.DS attribute), 36
 data (dpkt.ieee80211.IEEE80211.FH attribute), 36
 data (dpkt.ieee80211.IEEE80211.IBSS attribute), 37
 data (dpkt.ieee80211.IEEE80211.IE attribute), 36
 data (dpkt.ieee80211.IEEE80211.MGMT_Frame attribute), 33
 data (dpkt.ieee80211.IEEE80211.QoS_Data attribute), 36
 data (dpkt.ieee80211.IEEE80211.Reassoc_Req attribute), 34
 data (dpkt.ieee80211.IEEE80211.RTS attribute), 33
 data (dpkt.ieee80211.IEEE80211.TIM attribute), 37
 data (dpkt.igmp.IGMP attribute), 38
 data (dpkt.ip.IP attribute), 38
 data (dpkt.ip6.IP6 attribute), 39
 data (dpkt.ip6.IP6AHHeader attribute), 40
 data (dpkt.ip6.IP6DstOptsHeader attribute), 40
 data (dpkt.ip6.IP6ESPHeader attribute), 41
 data (dpkt.ip6.IP6FragmentHeader attribute), 40
 data (dpkt.ip6.IP6HopOptsHeader attribute), 40
 data (dpkt.ip6.IP6OptsHeader attribute), 39
 data (dpkt.ip6.IP6RoutingHeader attribute), 40
 data (dpkt.ipx.IPX attribute), 41
 data (dpkt.llc.LLC attribute), 42
 data (dpkt.loopback.Loopback attribute), 42
 data (dpkt.mrt.BGP4MPMessage attribute), 43
 data (dpkt.mrt.BGP4MPMessage_32 attribute), 43
 data (dpkt.mrt.MRTHeader attribute), 42
 data (dpkt.mrt.TableDump attribute), 43
 data (dpkt.netbios.Datagram attribute), 45
 data (dpkt.netbios.NS attribute), 44
 data (dpkt.netbios.NS.Q attribute), 44
 data (dpkt.netbios.NS.RR attribute), 44
 data (dpkt.netbios.Session attribute), 44
 data (dpkt.netflow.Netflow1 attribute), 46
 data (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 data (dpkt.netflow.Netflow5 attribute), 48
 data (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 data (dpkt.netflow.Netflow6 attribute), 49
 data (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 data (dpkt.netflow.Netflow7 attribute), 50
 data (dpkt.netflow.Netflow7.NetflowRecord attribute), 49
 data (dpkt.netflow.NetflowBase attribute), 45
 data (dpkt.ntp.NTP attribute), 51
 data (dpkt.ospf.OSPF attribute), 51
 data (dpkt.pcap.FileHdr attribute), 52
 data (dpkt.pcap.LEFileHdr attribute), 53
 data (dpkt.pcap.LEPktHdr attribute), 52
 data (dpkt.pcap.PktHdr attribute), 52
 data (dpkt.pim.PIM attribute), 54
 data (dpkt.pmap.Pmap attribute), 54
 data (dpkt.ppp.PPP attribute), 55
 data (dpkt.pppoe.PPP attribute), 56
 data (dpkt.pppoe.PPPoE attribute), 56
 data (dpkt.qq.QQ3Packet attribute), 56
 data (dpkt.qq.QQ5Packet attribute), 57
 data (dpkt.qq.QQBasicPacket attribute), 56
 data (dpkt.radiotap.Radiotap attribute), 59
 data (dpkt.radiotap.Radiotap.Antenna attribute), 58
 data (dpkt.radiotap.Radiotap.AntennaNoise attribute), 58
 data (dpkt.radiotap.Radiotap.AntennaSignal attribute), 58
 data (dpkt.radiotap.Radiotap.Channel attribute), 58
 data (dpkt.radiotap.Radiotap.DbAntennaNoise attribute), 59
 data (dpkt.radiotap.Radiotap.DbAntennaSignal attribute), 59
 data (dpkt.radiotap.Radiotap.DbmTxPower attribute), 59
 data (dpkt.radiotap.Radiotap.DbTxAttenuation attribute), 59
 data (dpkt.radiotap.Radiotap.FHSS attribute), 58
 data (dpkt.radiotap.Radiotap.Flags attribute), 58
 data (dpkt.radiotap.Radiotap.LockQuality attribute), 58
 data (dpkt.radiotap.Radiotap.Rate attribute), 59
 data (dpkt.radiotap.Radiotap.RxFlags attribute), 59
 data (dpkt.radiotap.Radiotap.TSFT attribute), 59
 data (dpkt.radiotap.Radiotap.TxAttenuation attribute), 59
 data (dpkt.radius.RADIUS attribute), 60
 data (dpkt.rfb.CutText attribute), 61
 data (dpkt.rfb.FramebufferUpdate attribute), 61
 data (dpkt.rfb.FramebufferUpdateRequest attribute), 61
 data (dpkt.rfb.KeyEvent attribute), 61
 data (dpkt.rfb.PointerEvent attribute), 61
 data (dpkt.rfb.RFB attribute), 60
 data (dpkt.rfb.SetColourMapEntries attribute), 61
 data (dpkt.rfb.SetEncodings attribute), 61
 data (dpkt.rfb.SetPixelFormat attribute), 60
 data (dpkt.rip.Auth attribute), 62
 data (dpkt.rip.RIP attribute), 62
 data (dpkt.rip.RTE attribute), 62

- data (dpkt.rpc.RPC attribute), 63
- data (dpkt.rpc.RPC.Auth attribute), 63
- data (dpkt.rpc.RPC.Call attribute), 63
- data (dpkt.rpc.RPC.Reply attribute), 63
- data (dpkt.rpc.RPC.Reply.Accept attribute), 63
- data (dpkt.rpc.RPC.Reply.Reject attribute), 63
- data (dpkt.rtp.RTP attribute), 64
- data (dpkt.rx.Rx attribute), 65
- data (dpkt.sccp.ActivateCallPlane attribute), 65
- data (dpkt.sccp.CallInfo attribute), 65
- data (dpkt.sccp.CallState attribute), 65
- data (dpkt.sccp.ClearPromptStatus attribute), 66
- data (dpkt.sccp.CloseReceiveChannel attribute), 66
- data (dpkt.sccp.DisplayPromptStatus attribute), 66
- data (dpkt.sccp.DisplayText attribute), 66
- data (dpkt.sccp.KeypadButton attribute), 66
- data (dpkt.sccp.OpenReceiveChannel attribute), 66
- data (dpkt.sccp.OpenReceiveChannelAck attribute), 66
- data (dpkt.sccp.SCCP attribute), 68
- data (dpkt.sccp.SelectStartKeys attribute), 67
- data (dpkt.sccp.SetLamp attribute), 67
- data (dpkt.sccp.SetSpeakerMode attribute), 67
- data (dpkt.sccp.StartMediaTransmission attribute), 67
- data (dpkt.sccp.StartTone attribute), 67
- data (dpkt.sccp.StopMediaTransmission attribute), 68
- data (dpkt.sctp.Chunk attribute), 68
- data (dpkt.sctp.SCTP attribute), 68
- data (dpkt.sll.SLL attribute), 69
- data (dpkt.smb.SMB attribute), 70
- data (dpkt.snoop.FileHdr attribute), 71
- data (dpkt.snoop.PktHdr attribute), 70
- data (dpkt.ssl.SSL2 attribute), 72
- data (dpkt.ssl.TLS attribute), 72
- data (dpkt.ssl.TLSAlert attribute), 73
- data (dpkt.ssl.TLSCertificate attribute), 73
- data (dpkt.ssl.TLSChangeCipherSpec attribute), 72
- data (dpkt.ssl.TLSClientHello attribute), 73
- data (dpkt.ssl.TLSHandshake attribute), 74
- data (dpkt.ssl.TLSHelloRequest attribute), 73
- data (dpkt.ssl.TLSRecord attribute), 72
- data (dpkt.ssl.TLSServerHello attribute), 73
- data (dpkt.ssl.TLSUnknownHandshake attribute), 73
- data (dpkt.stp.STP attribute), 77
- data (dpkt.stun.STUN attribute), 78
- data (dpkt.tcp.TCP attribute), 78
- data (dpkt.tftp.TFTP attribute), 79
- data (dpkt.tns.TNS attribute), 80
- data (dpkt.tpkt.TPKT attribute), 80
- data (dpkt.udp.UDP attribute), 80
- data (dpkt.vrrp.VRRP attribute), 81
- data (dpkt.yahoo.YHOO attribute), 82
- data (dpkt.yahoo.YMSG attribute), 82
- Datagram (class in dpkt.netbios), 44
- datalink() (dpkt.pcap.Reader method), 53
- datalink() (dpkt.snoop.Reader method), 71
- db (dpkt.radiotap.Radiotap.AntennaNoise attribute), 58
- db (dpkt.radiotap.Radiotap.AntennaSignal attribute), 58
- db (dpkt.radiotap.Radiotap.DbAntennaNoise attribute), 59
- db (dpkt.radiotap.Radiotap.DbAntennaSignal attribute), 59
- db (dpkt.radiotap.Radiotap.DbTxAttenuation attribute), 59
- db_ant_noise_present (dpkt.radiotap.Radiotap attribute), 58
- db_ant_sig_present (dpkt.radiotap.Radiotap attribute), 58
- db_tx_attn_present (dpkt.radiotap.Radiotap attribute), 57
- dbm (dpkt.radiotap.Radiotap.DbTxPower attribute), 59
- dbm_tx_power_present (dpkt.radiotap.Radiotap attribute), 57
- decode() (in module dpkt.asn1), 13
- decode_name() (in module dpkt.netbios), 43
- decompress() (dpkt.gzip.Gzip method), 26
- decorator_with_args() (in module dpkt.decorators), 18
- delay (dpkt.ntp.NTP attribute), 51
- deprecated() (in module dpkt.decorators), 18
- deprecated_decorator() (dpkt.decorators.TestDeprecatedDecorator method), 18
- description (dpkt.ssl.TLSAlert attribute), 73
- df (dpkt.ip.IP attribute), 38
- DHCP (class in dpkt.dhcp), 19
- dialog (dpkt.ieee80211.IEEE80211.BlockAckActionRequest attribute), 35
- dialog (dpkt.ieee80211.IEEE80211.BlockAckActionResponse attribute), 35
- Diameter (class in dpkt.diameter), 19
- dir (dpkt.rpc.RPC attribute), 64
- dispatch() (dpkt.pcap.Reader method), 53
- dispatch() (dpkt.snoop.Reader method), 71
- dispersion (dpkt.ntp.NTP attribute), 51
- display_msg (dpkt.sccp.DisplayPromptStatus attribute), 66
- display_msg (dpkt.sccp.DisplayText attribute), 66
- DisplayPromptStatus (class in dpkt.sccp), 66
- DisplayText (class in dpkt.sccp), 66
- DNS (class in dpkt.dns), 20
- DNS.Q (class in dpkt.dns), 21
- DNS.RR (class in dpkt.dns), 21
- done() (in module dpkt.crc32c), 18
- down_flag (dpkt.rfb.KeyEvent attribute), 61
- dpkt.ah (module), 9
- dpkt.aim (module), 10
- dpkt.aoe (module), 11
- dpkt.aoeata (module), 11
- dpkt.aocfg (module), 12
- dpkt.arp (module), 12
- dpkt.asn1 (module), 13
- dpkt.bgp (module), 13

dpkt.cdp (module), 17
 dpkt.crc32c (module), 18
 dpkt.decorators (module), 18
 dpkt.dhcp (module), 19
 dpkt.diameter (module), 19
 dpkt.dns (module), 20
 dpkt.dpkt (module), 22
 dpkt.dtp (module), 23
 dpkt.esp (module), 23
 dpkt.ethernet (module), 24
 dpkt.gre (module), 25
 dpkt.gzip (module), 26
 dpkt.h225 (module), 27
 dpkt.hsrp (module), 28
 dpkt.http (module), 28
 dpkt.icmp (module), 29
 dpkt.icmp6 (module), 31
 dpkt.ieee80211 (module), 32
 dpkt.igmp (module), 37
 dpkt.ip (module), 38
 dpkt.ip6 (module), 39
 dpkt.ipx (module), 41
 dpkt.llc (module), 41
 dpkt.loopback (module), 42
 dpkt.mrt (module), 42
 dpkt.netbios (module), 43
 dpkt.netflow (module), 45
 dpkt.ntp (module), 50
 dpkt.ospf (module), 51
 dpkt.pcap (module), 52
 dpkt.pim (module), 54
 dpkt.pmap (module), 54
 dpkt.ppp (module), 55
 dpkt.pppoe (module), 55
 dpkt.qq (module), 56
 dpkt.radiotap (module), 57
 dpkt.radius (module), 60
 dpkt.rfb (module), 60
 dpkt.rip (module), 62
 dpkt.rpc (module), 63
 dpkt.rtp (module), 64
 dpkt.rx (module), 64
 dpkt.sccp (module), 65
 dpkt.sctp (module), 68
 dpkt.sip (module), 69
 dpkt.sll (module), 69
 dpkt.smb (module), 70
 dpkt.snoop (module), 70
 dpkt.ssl (module), 72
 dpkt.ssl_ciphersuites (module), 76
 dpkt.stp (module), 77
 dpkt.stun (module), 78
 dpkt.tcp (module), 78
 dpkt.telnet (module), 79
 dpkt.tftp (module), 79
 dpkt.tns (module), 79
 dpkt.tpkt (module), 80
 dpkt.udp (module), 80
 dpkt.vrrp (module), 81
 dpkt.yahoo (module), 81
 dport (dpkt.sctp.SCTP attribute), 68
 dport (dpkt.tcp.TCP attribute), 78
 dport (dpkt.udp.UDP attribute), 80
 dsap (dpkt.llc.LLC attribute), 42
 dst (dpkt.ethernet.Ethernet attribute), 24
 dst (dpkt.ieee80211.IEEE80211.ACK attribute), 33
 dst (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33
 dst (dpkt.ieee80211.IEEE80211.BlockAckReq attribute), 32
 dst (dpkt.ieee80211.IEEE80211.CFEnd attribute), 33
 dst (dpkt.ieee80211.IEEE80211.CTS attribute), 33
 dst (dpkt.ieee80211.IEEE80211.Data attribute), 35
 dst (dpkt.ieee80211.IEEE80211.DataFromDS attribute), 35
 dst (dpkt.ieee80211.IEEE80211.DataInterDS attribute), 36
 dst (dpkt.ieee80211.IEEE80211.DataToDS attribute), 35
 dst (dpkt.ieee80211.IEEE80211.MGMT_Frame attribute), 33
 dst (dpkt.ieee80211.IEEE80211.RTS attribute), 33
 dst (dpkt.ip.IP attribute), 38
 dst (dpkt.ip6.IP6 attribute), 39
 dst (dpkt.ipx.IPX attribute), 41
 dst_addr (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 dst_addr (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 dst_addr (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 dst_addr (dpkt.netflow.Netflow7.NetflowRecord attribute), 49
 dst_as (dpkt.mrt.BGP4MPMessage attribute), 43
 dst_as (dpkt.mrt.BGP4MPMessage_32 attribute), 43
 dst_as (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 dst_as (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 dst_as (dpkt.netflow.Netflow7.NetflowRecord attribute), 49
 dst_ip (dpkt.mrt.BGP4MPMessage attribute), 43
 dst_ip (dpkt.mrt.BGP4MPMessage_32 attribute), 43
 dst_mask (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 dst_mask (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 dst_mask (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

- ul style="list-style-type: none; padding-left: 0;">
- dst_port (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- dst_port (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
- dst_port (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
- dst_port (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
- DTP (class in dpkt.dtp), 23
- dur (dpkt.ieee80211.IEEE80211.CF attribute), 36
- duration (dpkt.ieee80211.IEEE80211 attribute), 37
- E**
- echo_cancel_type (dpkt.sccp.OpenReceiveChannel attribute), 66
- encode_name() (in module dpkt.netbios), 43
- encoding (dpkt.ssl.ciphersuites.CipherSuite attribute), 76
- end_id (dpkt.diameter.Diameter attribute), 20
- end_time (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- end_time (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
- end_time (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
- end_time (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
- engine_id (dpkt.netflow.Netflow5 attribute), 48
- engine_id (dpkt.netflow.Netflow6 attribute), 49
- engine_type (dpkt.netflow.Netflow5 attribute), 48
- engine_type (dpkt.netflow.Netflow6 attribute), 49
- epoch (dpkt.rx.Rx attribute), 65
- err (dpkt.aoe.AOE attribute), 11
- errfeat (dpkt.aoeata.AOEATA attribute), 11
- Error, 22
- error_flag (dpkt.diameter.Diameter attribute), 20
- ESP (class in dpkt.esp), 23
- Ethernet (class in dpkt.ethernet), 24
- ethtype (dpkt.sll.SLL attribute), 69
- examples.print_http_requests (module), 7
- examples.print_icmp (module), 6
- examples.print_packets (module), 4
- ext_present (dpkt.radiotap.Radiotap attribute), 58
- extended_length (dpkt.bgp.BGP.Update.Attribute attribute), 14
- extra (dpkt.gzip.Gzip attribute), 26
- F**
- family (dpkt.aim.SNAC attribute), 10
- family (dpkt.gre.GRE.SRE attribute), 26
- family (dpkt.loopback.Loopback attribute), 42
- family (dpkt.mrt.BGP4MPMessage attribute), 43
- family (dpkt.mrt.BGP4MPMessage_32 attribute), 43
- family (dpkt.rip.RTE attribute), 62
- fc (dpkt.ip6.IP6 attribute), 39
- at-fcs (dpkt.radiotap.Radiotap.Flags attribute), 58
- fd (dpkt.pcap.Reader attribute), 53
- at-fd (dpkt.stp.STP attribute), 77
- fhss_present (dpkt.radiotap.Radiotap attribute), 57
- at-file (dpkt.dhcp.DHCP attribute), 19
- FileHdr (class in dpkt.pcap), 52
- at-FileHdr (class in dpkt.snoop), 71
- filename (dpkt.gzip.Gzip attribute), 26
- fileno() (dpkt.pcap.Reader method), 53
- fileno() (dpkt.snoop.Reader method), 71
- first_colour (dpkt.rfb.SetColourMapEntries attribute), 61
- fl (dpkt.aoe.AOE attribute), 11
- flags (dpkt.aim.SNAC attribute), 10
- at-flags (dpkt.bgp.BGP.Update.Attribute attribute), 16
- flags (dpkt.dhcp.DHCP attribute), 19
- flags (dpkt.diameter.AVP attribute), 20
- flags (dpkt.diameter.Diameter attribute), 20
- flags (dpkt.gre.GRE attribute), 26
- at-flags (dpkt.gzip.Gzip attribute), 27
- flags (dpkt.netbios.Datagram attribute), 45
- at-flags (dpkt.netbios.Session attribute), 44
- flags (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
- at-flags (dpkt.ntp.NTP attribute), 51
- flags (dpkt.radiotap.Radiotap.Channel attribute), 58
- at-flags (dpkt.rx.Rx attribute), 65
- flags (dpkt.sctp.Chunk attribute), 68
- flags (dpkt.smb.SMB attribute), 70
- flags (dpkt.stp.STP attribute), 77
- flags (dpkt.tcp.TCP attribute), 78
- flags2 (dpkt.smb.SMB attribute), 70
- flags_present (dpkt.radiotap.Radiotap attribute), 57
- FLAP (class in dpkt.aim), 10
- flavor (dpkt.rpc.RPC.Auth attribute), 63
- flow (dpkt.ip6.IP6 attribute), 39
- flow_sequence (dpkt.netflow.Netflow5 attribute), 48
- flow_sequence (dpkt.netflow.Netflow6 attribute), 49
- flow_sequence (dpkt.netflow.Netflow7 attribute), 50
- frag_off (dpkt.ip6.IP6FragmentHeader attribute), 40
- frag_off_resv_m (dpkt.ip6.IP6FragmentHeader attribute), 40
- frag_seq (dpkt.ieee80211.IEEE80211.Data attribute), 35
- frag_seq (dpkt.ieee80211.IEEE80211.DataFromDS attribute), 35
- frag_seq (dpkt.ieee80211.IEEE80211.DataInterDS attribute), 36
- frag_seq (dpkt.ieee80211.IEEE80211.DataToDS attribute), 35
- frag_seq (dpkt.ieee80211.IEEE80211.MGMT_Frame attribute), 33
- FramebufferUpdate (class in dpkt.rfb), 61
- FramebufferUpdateRequest (class in dpkt.rfb), 61
- framectl (dpkt.ieee80211.IEEE80211 attribute), 37
- freq (dpkt.radiotap.Radiotap.Channel attribute), 58
- from_ds (dpkt.ieee80211.IEEE80211 attribute), 32

fwver (dpkt.aoeefg.AOECFG attribute), 12

G

g723_bitrate (dpkt.sccp.OpenReceiveChannel attribute), 66
 g723_bitrate (dpkt.sccp.StartMediaTransmission attribute), 67
 get_cmd() (dpkt.aoe.AOE class method), 11
 get_p() (dpkt.ppp.PPP class method), 55
 get_proto() (dpkt.ip.IP class method), 38
 get_proto() (dpkt.ip6.IP6 class method), 39
 get_type() (dpkt.ethernet.Ethernet class method), 24
 get_type_rev() (dpkt.ethernet.Ethernet class method), 24
 giaddr (dpkt.dhcp.DHCP attribute), 19
 GRE (class in dpkt.gre), 25
 GRE.SRE (class in dpkt.gre), 26
 group (dpkt.hsrp.HSRP attribute), 28
 group (dpkt.igmp.IGMP attribute), 38
 gw (dpkt.icmp.ICMP.Redirect attribute), 30
 Gzip (class in dpkt.gzip), 26
 GzipExtra (class in dpkt.gzip), 26

H

H225 (class in dpkt.h225), 27
 H225.IE (class in dpkt.h225), 27
 hdr (dpkt.sll.SLL attribute), 69
 hdrsum (dpkt.tns.TNS attribute), 80
 header_type (dpkt.qq.QQ3Packet attribute), 56
 header_type (dpkt.qq.QQ5Packet attribute), 57
 header_type (dpkt.qq.QQBasicPacket attribute), 56
 headers (dpkt.http.Message attribute), 28
 headers_str() (dpkt.ip6.IP6 method), 39
 height (dpkt.rfb.FramebufferUpdateRequest attribute), 61
 hello (dpkt.hsrp.HSRP attribute), 28
 hello (dpkt.stp.STP attribute), 77
 hexdump() (in module dpkt.dpkt), 23
 hl (dpkt.ip.IP attribute), 38
 hlen (dpkt.sll.SLL attribute), 69
 hlim (dpkt.ip6.IP6 attribute), 39
 hln (dpkt.arp.ARP attribute), 13
 hln (dpkt.dhcp.DHCP attribute), 19
 hold (dpkt.hsrp.HSRP attribute), 28
 holdtime (dpkt.bgp.BGP.Open attribute), 14
 hop_id (dpkt.diameter.Diameter attribute), 20
 hopindex (dpkt.ieee80211.IEEE80211.FH attribute), 36
 hoppattern (dpkt.ieee80211.IEEE80211.FH attribute), 36
 hops (dpkt.dhcp.DHCP attribute), 19
 hopset (dpkt.ieee80211.IEEE80211.FH attribute), 36
 hrd (dpkt.arp.ARP attribute), 13
 hrd (dpkt.dhcp.DHCP attribute), 19
 hrd (dpkt.sll.SLL attribute), 69
 hsa (dpkt.ethernet.VLANtagISL attribute), 25
 HSRP (class in dpkt.hsrp), 28

I

ICMP (class in dpkt.icmp), 29
 ICMP.Echo (class in dpkt.icmp), 29
 ICMP.ParamProbe (class in dpkt.icmp), 30
 ICMP.Quench (class in dpkt.icmp), 30
 ICMP.Quote (class in dpkt.icmp), 30
 ICMP.Redirect (class in dpkt.icmp), 30
 ICMP.TimeExceed (class in dpkt.icmp), 30
 ICMP.Unreach (class in dpkt.icmp), 30
 ICMP6 (class in dpkt.icmp6), 31
 ICMP6.Echo (class in dpkt.icmp6), 31
 ICMP6.Error (class in dpkt.icmp6), 31
 ICMP6.ParamProb (class in dpkt.icmp6), 31
 ICMP6.TimeExceed (class in dpkt.icmp6), 31
 ICMP6.TooBig (class in dpkt.icmp6), 31
 ICMP6.Unreach (class in dpkt.icmp6), 31
 id (dpkt.dns.DNS attribute), 22
 id (dpkt.gzip.GzipExtra attribute), 26
 id (dpkt.icmp.ICMP.Echo attribute), 30
 id (dpkt.icmp6.ICMP6.Echo attribute), 31
 id (dpkt.ieee80211.IEEE80211.CF attribute), 36
 id (dpkt.ieee80211.IEEE80211.DS attribute), 36
 id (dpkt.ieee80211.IEEE80211.FH attribute), 36
 id (dpkt.ieee80211.IEEE80211.IBSS attribute), 37
 id (dpkt.ieee80211.IEEE80211.IE attribute), 36
 id (dpkt.ieee80211.IEEE80211.TIM attribute), 37
 id (dpkt.ip.IP attribute), 38
 id (dpkt.ip6.IP6FragmentHeader attribute), 40
 id (dpkt.netbios.Datagram attribute), 45
 id (dpkt.netbios.NS attribute), 44
 id (dpkt.ntp.NTP attribute), 51
 id (dpkt.radius.RADIUS attribute), 60
 identifier (dpkt.bgp.BGP.Open attribute), 14
 IEEE80211 (class in dpkt.ieee80211), 32
 IEEE80211.ACK (class in dpkt.ieee80211), 33
 IEEE80211.Action (class in dpkt.ieee80211), 34
 IEEE80211.Assoc_Req (class in dpkt.ieee80211), 34
 IEEE80211.Assoc_Resp (class in dpkt.ieee80211), 34
 IEEE80211.Auth (class in dpkt.ieee80211), 34
 IEEE80211.Beacon (class in dpkt.ieee80211), 33
 IEEE80211.BlockAck (class in dpkt.ieee80211), 32
 IEEE80211.BlockAckActionRequest (class in dpkt.ieee80211), 34
 IEEE80211.BlockAckActionResponse (class in dpkt.ieee80211), 35
 IEEE80211.BlockAckReq (class in dpkt.ieee80211), 32
 IEEE80211.Capability (class in dpkt.ieee80211), 32
 IEEE80211.CF (class in dpkt.ieee80211), 36
 IEEE80211.CFEnd (class in dpkt.ieee80211), 33
 IEEE80211.CTS (class in dpkt.ieee80211), 33
 IEEE80211.Data (class in dpkt.ieee80211), 35
 IEEE80211.DataFromDS (class in dpkt.ieee80211), 35
 IEEE80211.DataInterDS (class in dpkt.ieee80211), 35
 IEEE80211.DataToDS (class in dpkt.ieee80211), 35

- IEEE80211.Deauth (class in dpkt.ieee80211), 34
 - IEEE80211.Disassoc (class in dpkt.ieee80211), 34
 - IEEE80211.DS (class in dpkt.ieee80211), 36
 - IEEE80211.FH (class in dpkt.ieee80211), 36
 - IEEE80211.IBSS (class in dpkt.ieee80211), 37
 - IEEE80211.IE (class in dpkt.ieee80211), 36
 - IEEE80211.MGMT_Frame (class in dpkt.ieee80211), 33
 - IEEE80211.QoS_Data (class in dpkt.ieee80211), 36
 - IEEE80211.Reassoc_Req (class in dpkt.ieee80211), 34
 - IEEE80211.RTS (class in dpkt.ieee80211), 33
 - IEEE80211.TIM (class in dpkt.ieee80211), 37
 - IGMP (class in dpkt.igmp), 37
 - in_cksum() (in module dpkt.dpkt), 23
 - in_cksum_add() (in module dpkt.dpkt), 23
 - in_cksum_done() (in module dpkt.dpkt), 23
 - in_encaps (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 - incl_len (dpkt.snoop.PktHdr attribute), 71
 - incremental (dpkt.rfb.FramebufferUpdateRequest attribute), 61
 - index (dpkt.radiotap.Radiotap.Antenna attribute), 58
 - indx (dpkt.ethernet.VLANtagISL attribute), 25
 - inet_to_str() (in module examples.print_http_requests), 8
 - inet_to_str() (in module examples.print_icmp), 6
 - inet_to_str() (in module examples.print_packets), 4
 - input_iface (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 - input_iface (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 - input_iface (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 - input_iface (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
 - interval (dpkt.ieee80211.IEEE80211.Assoc_Req attribute), 34
 - interval (dpkt.ieee80211.IEEE80211.Beacon attribute), 33
 - interval (dpkt.ieee80211.IEEE80211.Reassoc_Req attribute), 34
 - interval (dpkt.ntp.NTP attribute), 51
 - intf (dpkt.mrt.BGP4MPMessage attribute), 43
 - intf (dpkt.mrt.BGP4MPMessage_32 attribute), 43
 - IP (class in dpkt.ip), 38
 - ip (dpkt.bgp.BGP.Update.Attribute.Aggregator attribute), 15
 - ip (dpkt.bgp.BGP.Update.Attribute.NextHop attribute), 15
 - ip (dpkt.sccp.OpenReceiveChannelAck attribute), 67
 - IP6 (class in dpkt.ip6), 39
 - IP6AHHeader (class in dpkt.ip6), 40
 - IP6DstOptsHeader (class in dpkt.ip6), 40
 - IP6ESPHeader (class in dpkt.ip6), 41
 - IP6ExtensionHeader (class in dpkt.ip6), 39
 - IP6FragmentHeader (class in dpkt.ip6), 40
 - IP6HopOptsHeader (class in dpkt.ip6), 40
 - IP6OptsHeader (class in dpkt.ip6), 39
 - IP6RoutingHeader (class in dpkt.ip6), 40
 - ip_proto (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 - ip_proto (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 - ip_proto (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 - ip_proto (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
 - IPX (class in dpkt.ipx), 41
 - is_snap (dpkt.llc.LLC attribute), 42
 - isstr() (in module dpkt.ethernet), 24
- ## K
- key (dpkt.rfb.KeyEvent attribute), 61
 - KeyEvent (class in dpkt.rfb), 61
 - KeypadButton (class in dpkt.sccp), 66
 - kx (dpkt.ssl_ciphersuites.CipherSuite attribute), 76
 - kx_auth (dpkt.ssl_ciphersuites.CipherSuite attribute), 76
- ## L
- lamp_mode (dpkt.sccp.SetLamp attribute), 67
 - lba0 (dpkt.aoeata.AOEATA attribute), 12
 - lba1 (dpkt.aoeata.AOEATA attribute), 12
 - lba2 (dpkt.aoeata.AOEATA attribute), 12
 - lba3 (dpkt.aoeata.AOEATA attribute), 12
 - lba4 (dpkt.aoeata.AOEATA attribute), 12
 - lba5 (dpkt.aoeata.AOEATA attribute), 12
 - LEFileHdr (class in dpkt.pcap), 53
 - len (dpkt.ah.AH attribute), 9
 - len (dpkt.aim.FLAP attribute), 10
 - len (dpkt.bgp.BGP attribute), 17
 - len (dpkt.bgp.BGP.Open.Parameter attribute), 14
 - len (dpkt.bgp.BGP.Open.Parameter.Capability attribute), 14
 - len (dpkt.bgp.BGP.Update.Attribute.ASPath.ASPathSegment attribute), 15
 - len (dpkt.bgp.RouteGeneric attribute), 17
 - len (dpkt.bgp.RouteIPV4 attribute), 17
 - len (dpkt.bgp.RouteIPV6 attribute), 17
 - len (dpkt.cdp.CDP.TLV attribute), 18
 - len (dpkt.diameter.AVP attribute), 20
 - len (dpkt.diameter.Diameter attribute), 20
 - len (dpkt.ethernet.VLANtagISL attribute), 25
 - len (dpkt.gre.GRE.SRE attribute), 26
 - len (dpkt.gzip.GzipExtra attribute), 26
 - len (dpkt.ieee80211.IEEE80211.CF attribute), 36
 - len (dpkt.ieee80211.IEEE80211.DS attribute), 36
 - len (dpkt.ieee80211.IEEE80211.FH attribute), 36
 - len (dpkt.ieee80211.IEEE80211.IBSS attribute), 37
 - len (dpkt.ieee80211.IEEE80211.IE attribute), 36
 - len (dpkt.ieee80211.IEEE80211.TIM attribute), 37

len (dpkt.ip.IP attribute), 38
 len (dpkt.ip6.IP6AHHeader attribute), 40
 len (dpkt.ip6.IP6DstOptsHeader attribute), 40
 len (dpkt.ip6.IP6HopOptsHeader attribute), 40
 len (dpkt.ip6.IP6OptsHeader attribute), 39
 len (dpkt.ip6.IP6RoutingHeader attribute), 40
 len (dpkt.ipx.IPX attribute), 41
 len (dpkt.mrt.MRTHeader attribute), 42
 len (dpkt.netbios.Datagram attribute), 45
 len (dpkt.netbios.Session attribute), 44
 len (dpkt.ospf.OSPF attribute), 52
 len (dpkt.pcap.LEPktHdr attribute), 52
 len (dpkt.pcap.PktHdr attribute), 52
 len (dpkt.pppoe.PPPoE attribute), 56
 len (dpkt.radius.RADIUS attribute), 60
 len (dpkt.sccp.SCCP attribute), 68
 len (dpkt.sctp.Chunk attribute), 69
 len (dpkt.ssl.SSL2 attribute), 72
 len (dpkt.ssl.TLS attribute), 72
 len (dpkt.stun.STUN attribute), 78
 len (dpkt.tpkt.TPKT attribute), 80
 length (dpkt.radiotap.Radiotap attribute), 59
 length (dpkt.rfb.CutText attribute), 62
 length (dpkt.ssl.TLSHandshake attribute), 74
 length (dpkt.ssl.TLSRecord attribute), 72
 length (dpkt.tns.TNS attribute), 80
 length (dpkt.yahoo.YHOO attribute), 82
 length (dpkt.yahoo.YMSG attribute), 82
 length_bytes (dpkt.ssl.TLSHandshake attribute), 74
 LEPktHdr (class in dpkt.pcap), 52
 level (dpkt.ssl.TLSAlert attribute), 73
 li (dpkt.ntp.NTP attribute), 51
 line_id (dpkt.sccp.SelectStartKeys attribute), 67
 line_instance (dpkt.sccp.ActivateCallPlane attribute), 65
 line_instance (dpkt.sccp.CallInfo attribute), 65
 line_instance (dpkt.sccp.CallState attribute), 65
 line_instance (dpkt.sccp.ClearPromptStatus attribute), 66
 line_instance (dpkt.sccp.DisplayPromptStatus attribute), 66
 linktype (dpkt.pcap.FileHdr attribute), 52
 linktype (dpkt.pcap.LEFileHdr attribute), 53
 linktype (dpkt.snoop.FileHdr attribute), 71
 LLC (class in dpkt.llc), 41
 lock_qual_present (dpkt.radiotap.Radiotap attribute), 57
 loop() (dpkt.pcap.Reader method), 54
 loop() (dpkt.snoop.Reader method), 71
 Loopback (class in dpkt.loopback), 42

M

m (dpkt.rtp.RTP attribute), 64
 m_flag (dpkt.ip6.IP6FragmentHeader attribute), 40
 mac_addr() (in module examples.print_http_requests), 8
 mac_addr() (in module examples.print_icmp), 6
 mac_addr() (in module examples.print_packets), 4

mac_size (dpkt.ssl_ciphersuites.CipherSuite attribute), 76
 MAC_SIZES (dpkt.ssl_ciphersuites.CipherSuite attribute), 76
 magic (dpkt.dhcp.DHCP attribute), 19
 magic (dpkt.gzip.Gzip attribute), 27
 magic (dpkt.pcap.FileHdr attribute), 52
 magic (dpkt.pcap.LEFileHdr attribute), 53
 magic (dpkt.snoop.FileHdr attribute), 71
 magic (dpkt.yahoo.YHOO attribute), 82
 maj (dpkt.aoe.AOE attribute), 11
 mandatory_flag (dpkt.diameter.AVP attribute), 20
 marker (dpkt.bgp.BGP attribute), 17
 max (dpkt.ieee80211.IEEE80211.CF attribute), 36
 max_age (dpkt.stp.STP attribute), 77
 max_frames_per_pkt (dpkt.sccp.StartMediaTransmission attribute), 67
 maxresp (dpkt.igmp.IGMP attribute), 38
 Message (class in dpkt.http), 28
 method (dpkt.gzip.Gzip attribute), 27
 metric (dpkt.rip.RTE attribute), 62
 mf (dpkt.ip.IP attribute), 38
 mid (dpkt.smb.SMB attribute), 70
 min (dpkt.aoe.AOE attribute), 11
 mode (dpkt.ntp.NTP attribute), 51
 more_data (dpkt.ieee80211.IEEE80211 attribute), 32
 more_frag (dpkt.ieee80211.IEEE80211 attribute), 32
 MPLSlabel (class in dpkt.ethernet), 24
 MRTHeader (class in dpkt.mrt), 42
 ms_packet (dpkt.sccp.OpenReceiveChannel attribute), 66
 ms_packet (dpkt.sccp.StartMediaTransmission attribute), 67
 msg (dpkt.sccp.SCCP attribute), 68
 msg (dpkt.ssl.SSL2 attribute), 72
 msg (dpkt.tns.TNS attribute), 80
 msg_timeout (dpkt.sccp.DisplayPromptStatus attribute), 66
 msgid (dpkt.sccp.SCCP attribute), 68
 mtime (dpkt.gzip.Gzip attribute), 27
 mtu (dpkt.icmp.ICMP.Unreach attribute), 30
 mtu (dpkt.icmp6.ICMP6.TooBig attribute), 31
 multi_tid (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33

N

name (dpkt.dns.DNS.Q attribute), 21
 name (dpkt.dns.DNS.RR attribute), 21
 name (dpkt.netbios.NS.Q attribute), 44
 name (dpkt.netbios.NS.RR attribute), 44
 name (dpkt.ssl_ciphersuites.CipherSuite attribute), 76
 NeedData, 22
 Netflow1 (class in dpkt.netflow), 45
 Netflow1.NetflowRecord (class in dpkt.netflow), 46
 Netflow5 (class in dpkt.netflow), 47
 Netflow5.NetflowRecord (class in dpkt.netflow), 47

Netflow6 (class in dpkt.netflow), 48
 Netflow6.NetflowRecord (class in dpkt.netflow), 48
 Netflow7 (class in dpkt.netflow), 49
 Netflow7.NetflowRecord (class in dpkt.netflow), 49
 NetflowBase (class in dpkt.netflow), 45
 NetflowBase.NetflowRecordBase (class in dpkt.netflow), 45
 new_method() (dpkt.decorators.TestDeprecatedDecorator method), 18
 next_hop (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 next_hop (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 next_hop (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 next_hop (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
 next_hop (dpkt.rip.RTE attribute), 62
 nick1 (dpkt.yahoo.YHOO attribute), 82
 nick2 (dpkt.yahoo.YHOO attribute), 82
 node_to_service_name() (in module dpkt.netbios), 43
 NS (class in dpkt.netbios), 43
 ns (dpkt.dns.DNS attribute), 22
 ns (dpkt.netbios.NS attribute), 44
 NS.Q (class in dpkt.netbios), 44
 NS.RR (class in dpkt.netbios), 44
 NTP (class in dpkt.ntp), 50
 num_colours (dpkt.rfb.SetColourMapEntries attribute), 61
 num_encodings (dpkt.rfb.SetEncodings attribute), 61
 num_rects (dpkt.rfb.FramebufferUpdate attribute), 61
 nxt (dpkt.ah.AH attribute), 9
 nxt (dpkt.ip6.IP6 attribute), 39
 nxt (dpkt.ip6.IP6AHHeader attribute), 40
 nxt (dpkt.ip6.IP6DstOptsHeader attribute), 40
 nxt (dpkt.ip6.IP6FragmentHeader attribute), 40
 nxt (dpkt.ip6.IP6HopOptsHeader attribute), 40
 nxt (dpkt.ip6.IP6OptsHeader attribute), 39
 nxt (dpkt.ip6.IP6RoutingHeader attribute), 40

O

off (dpkt.gre.GRE.SRE attribute), 26
 off (dpkt.ip.IP attribute), 38
 off (dpkt.netbios.Datagram attribute), 45
 off (dpkt.tcp.TCP attribute), 78
 offset (dpkt.ip.IP attribute), 38
 old_method() (dpkt.decorators.TestDeprecatedDecorator method), 18
 op (dpkt.arp.ARP attribute), 13
 op (dpkt.dhcp.DHCP attribute), 19
 op (dpkt.dns.DNS attribute), 22
 op (dpkt.netbios.NS attribute), 44
 opcode (dpkt.dns.DNS attribute), 21
 opcode (dpkt.hsrp.HSRP attribute), 28

opcode (dpkt.tftp.TFTP attribute), 79
 OpenReceiveChannel (class in dpkt.sccp), 66
 OpenReceiveChannelAck (class in dpkt.sccp), 66
 opt_fields_fmts() (dpkt.gre.GRE method), 26
 optional (dpkt.bgp.BGP.Update.Attribute attribute), 14
 opts (dpkt.dhcp.DHCP attribute), 19
 opts (dpkt.ip.IP attribute), 38
 opts (dpkt.tcp.TCP attribute), 78
 order (dpkt.ieee80211.IEEE80211 attribute), 32
 orig_called_party (dpkt.sccp.CallInfo attribute), 65
 orig_called_party_name (dpkt.sccp.CallInfo attribute), 65
 orig_len (dpkt.snoop.PktHdr attribute), 71
 originate_time (dpkt.ntp.NTP attribute), 51
 originated_ts (dpkt.mrt.TableDump attribute), 43
 os (dpkt.gzip.Gzip attribute), 27
 OSPF (class in dpkt.ospf), 51
 out_encaps (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 output_iface (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 output_iface (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 output_iface (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
 output_iface (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

P

p (dpkt.cdp.CDP.Address attribute), 18
 p (dpkt.gre.GRE attribute), 26
 p (dpkt.ip.IP attribute), 38
 p (dpkt.ppp.PPP attribute), 55
 p (dpkt.pppoe.PPP attribute), 56
 p (dpkt.rtp.RTP attribute), 64
 pack() (dpkt.dpkt.Packet method), 23
 pack_hdr() (dpkt.aoe.AOE method), 11
 pack_hdr() (dpkt.diameter.AVP method), 20
 pack_hdr() (dpkt.diameter.Diameter method), 20
 pack_hdr() (dpkt.dpkt.Packet method), 23
 pack_hdr() (dpkt.ethernet.Ethernet method), 24
 pack_hdr() (dpkt.ethernet.MPLSlabel method), 24
 pack_hdr() (dpkt.ethernet.VLANtag8021Q method), 25
 pack_hdr() (dpkt.ethernet.VLANtagISL method), 25
 pack_hdr() (dpkt.gzip.Gzip method), 26
 pack_hdr() (dpkt.http.Message method), 29
 pack_hdr() (dpkt.llc.LLC method), 42
 pack_hdr() (dpkt.ppp.PPP method), 55
 pack_hdr() (dpkt.pppoe.PPP method), 56
 pack_name() (dpkt.netbios.NS method), 44
 pack_name() (in module dpkt.dns), 20
 pack_opts() (dpkt.dhcp.DHCP method), 19
 pack_q() (dpkt.dns.DNS method), 21
 pack_rdata() (dpkt.dns.DNS.RR method), 21
 pack_rr() (dpkt.dns.DNS method), 22

- pack_xdrlist() (in module dpkt.rpc), 64
- PackError, 22
- Packet (class in dpkt.dpkt), 22
- pad (dpkt.icmp.ICMP.Quench attribute), 30
- pad (dpkt.icmp.ICMP.Quote attribute), 30
- pad (dpkt.icmp.ICMP.TimeExceed attribute), 30
- pad (dpkt.icmp.ICMP.Unreach attribute), 30
- pad (dpkt.icmp6.ICMP6.Error attribute), 31
- pad (dpkt.icmp6.ICMP6.TimeExceed attribute), 31
- pad (dpkt.icmp6.ICMP6.Unreach attribute), 31
- pad (dpkt.radiotap.Radiotap attribute), 59
- pad (dpkt.rfb.CutText attribute), 62
- pad (dpkt.rfb.FramebufferUpdate attribute), 61
- pad (dpkt.rfb.KeyEvent attribute), 61
- pad (dpkt.rfb.SetColourMapEntries attribute), 61
- pad (dpkt.rfb.SetEncodings attribute), 61
- pad (dpkt.rfb.SetPixelFormat attribute), 60
- pad (dpkt.ssl.SSL2 attribute), 72
- pad1 (dpkt.icmp.ICMP.ParamProbe attribute), 30
- pad1 (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- pad1 (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
- pad1 (dpkt.netflow.Netflow6.NetflowRecord attribute), 48
- pad2 (dpkt.icmp.ICMP.ParamProbe attribute), 30
- pad2 (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- pad2 (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
- pad2 (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
- pad3 (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- param_len (dpkt.bgp.BGP.Open attribute), 14
- parameters (dpkt.ieee80211.IEEE80211.BlockAckActionRequest attribute), 35
- parameters (dpkt.ieee80211.IEEE80211.BlockAckActionResponse attribute), 35
- parse_attrs() (in module dpkt.radius), 60
- parse_attrs() (in module dpkt.stun), 78
- parse_body() (in module dpkt.http), 28
- parse_extensions() (in module dpkt.ssl), 72
- parse_headers() (in module dpkt.http), 28
- parse_opts() (in module dpkt.tcp), 79
- parse_variable_array() (in module dpkt.ssl), 72
- partial (dpkt.bgp.BGP.Update.Attribute attribute), 14
- passthrupty_id (dpkt.sccp.CloseReceiveChannel attribute), 66
- passthrupty_id (dpkt.sccp.OpenReceiveChannel attribute), 66
- passthrupty_id (dpkt.sccp.OpenReceiveChannelAck attribute), 67
- passthrupty_id (dpkt.sccp.StartMediaTransmission attribute), 67
- passthrupty_id (dpkt.sccp.StopMediaTransmission attribute), 68
- pattern (dpkt.radiotap.Radiotap.FHSS attribute), 58
- payload_capability (dpkt.sccp.OpenReceiveChannel attribute), 66
- payload_capability (dpkt.sccp.StartMediaTransmission attribute), 67
- peer_as (dpkt.mrt.TableDump attribute), 43
- peer_ip (dpkt.mrt.TableDump attribute), 43
- peer_nexthop (dpkt.netflow.Netflow6.NetflowRecord attribute), 49
- period (dpkt.ieee80211.IEEE80211.CF attribute), 36
- period (dpkt.ieee80211.IEEE80211.TIM attribute), 37
- pid (dpkt.smb.SMB attribute), 70
- PIM (class in dpkt.pim), 54
- pixel_fmt (dpkt.rfb.SetPixelFormat attribute), 60
- PktHdr (class in dpkt.pcap), 52
- PktHdr (class in dpkt.snoop), 70
- pkts_sent (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
- pkts_sent (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
- pkts_sent (dpkt.netflow.Netflow6.NetflowRecord attribute), 49
- pkts_sent (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
- pktsum (dpkt.tns.TNS attribute), 80
- plen (dpkt.cdp.CDP.Address attribute), 18
- plen (dpkt.ip6.IP6 attribute), 39
- pln (dpkt.arp.ARP attribute), 13
- Pmap (class in dpkt.pmap), 54
- PointerEvent (class in dpkt.rfb), 61
- port (dpkt.pmap.Pmap attribute), 54
- ports (dpkt.sccp.OpenReceiveChannelAck attribute), 67
- port_id (dpkt.stp.STP attribute), 77
- PPP (class in dpkt.ppp), 55
- PPP (class in dpkt.pppoe), 56
- PPPoE (class in dpkt.pppoe), 55
- precedence (dpkt.sccp.StartMediaTransmission attribute), 67
- precision (dpkt.ntp.NTP attribute), 51
- prefix (dpkt.mrt.TableDump attribute), 43
- prefix_len (dpkt.mrt.TableDump attribute), 43
- present_flags (dpkt.radiotap.Radiotap attribute), 59
- print_http_requests() (in module examples.print_http_requests), 8
- print_icmp() (in module examples.print_icmp), 6
- print_packets() (in module examples.print_packets), 4
- priority (dpkt.hsrp.HSRP attribute), 28
- priority (dpkt.vrrp.VRRP attribute), 81
- pro (dpkt.arp.ARP attribute), 13
- proc (dpkt.rpc.RPC.Call attribute), 63
- prog (dpkt.pmap.Pmap attribute), 55
- prog (dpkt.rpc.RPC.Call attribute), 63
- prot (dpkt.pmap.Pmap attribute), 55
- protected_flag (dpkt.diameter.AVP attribute), 20
- proto (dpkt.h225.H225 attribute), 27
- proto (dpkt.smb.SMB attribute), 70
- proto_id (dpkt.stp.STP attribute), 77

proxiable_flag (dpkt.diameter.Diameter attribute), 20
pt (dpkt.ipx.IPX attribute), 41
pt (dpkt.rtp.RTP attribute), 64
ptr (dpkt.icmp.ICMP.ParamProbe attribute), 30
ptr (dpkt.icmp6.ICMP6.ParamProb attribute), 31
ptype (dpkt.cdp.CDP.Address attribute), 18
pwr_mgt (dpkt.ieee80211.IEEE80211 attribute), 32

Q

qd (dpkt.dns.DNS attribute), 22
qd (dpkt.netbios.NS attribute), 44
QQ3Packet (class in dpkt.qq), 56
QQ5Packet (class in dpkt.qq), 57
QQBasicPacket (class in dpkt.qq), 56
qqNum (dpkt.qq.QQ5Packet attribute), 57
qqNum (dpkt.qq.QQBasicPacket attribute), 56
qr (dpkt.dns.DNS attribute), 21

R

ra (dpkt.dns.DNS attribute), 21
Radiotap (class in dpkt.radiotap), 57
Radiotap.Antenna (class in dpkt.radiotap), 58
Radiotap.AntennaNoise (class in dpkt.radiotap), 58
Radiotap.AntennaSignal (class in dpkt.radiotap), 58
Radiotap.Channel (class in dpkt.radiotap), 58
Radiotap.DbAntennaNoise (class in dpkt.radiotap), 59
Radiotap.DbAntennaSignal (class in dpkt.radiotap), 59
Radiotap.DbmTxPower (class in dpkt.radiotap), 59
Radiotap.DbTxAttenuation (class in dpkt.radiotap), 59
Radiotap.FHSS (class in dpkt.radiotap), 58
Radiotap.Flags (class in dpkt.radiotap), 58
Radiotap.LockQuality (class in dpkt.radiotap), 58
Radiotap.Rate (class in dpkt.radiotap), 59
Radiotap.RxFlags (class in dpkt.radiotap), 59
Radiotap.TSFT (class in dpkt.radiotap), 59
Radiotap.TxAttenuation (class in dpkt.radiotap), 59
RADIUS (class in dpkt.radius), 60
random (dpkt.ssl.TLSClientHello attribute), 73
random (dpkt.ssl.TLSServerHello attribute), 73
rate_present (dpkt.radiotap.Radiotap attribute), 57
rcode (dpkt.dns.DNS attribute), 21
rd (dpkt.dns.DNS attribute), 21
rdata (dpkt.dns.DNS.RR attribute), 21
rdata (dpkt.netbios.NS.RR attribute), 44
Reader (class in dpkt.pcap), 53
Reader (class in dpkt.snoop), 71
readpkts() (dpkt.pcap.Reader method), 53
readpkts() (dpkt.snoop.Reader method), 71
reason (dpkt.ieee80211.IEEE80211.Deauth attribute), 34
reason (dpkt.ieee80211.IEEE80211.Disassoc attribute), 34
rec_len (dpkt.snoop.PktHdr attribute), 71
receive_time (dpkt.ntp.NTP attribute), 51
recur (dpkt.gre.GRE attribute), 26

ref_len (dpkt.h225.H225 attribute), 27
remote_ip (dpkt.sccp.StartMediaTransmission attribute), 67
remote_port (dpkt.sccp.StartMediaTransmission attribute), 67
reqid (dpkt.aim.SNAC attribute), 10
Request (class in dpkt.http), 29
Request (class in dpkt.sip), 69
request_flag (dpkt.diameter.Diameter attribute), 20
res (dpkt.aoeata.AOEATA attribute), 12
res (dpkt.ethernet.VLANtagISL attribute), 25
reserved (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
reserved (dpkt.netflow.Netflow5 attribute), 48
reserved (dpkt.netflow.Netflow6 attribute), 49
reserved (dpkt.netflow.Netflow7 attribute), 50
Response (class in dpkt.http), 29
Response (class in dpkt.sip), 69
resv (dpkt.ip6.IP6AHHeader attribute), 40
resv (dpkt.ip6.IP6FragmentHeader attribute), 40
retransmit_flag (dpkt.diameter.Diameter attribute), 20
retry (dpkt.ieee80211.IEEE80211 attribute), 32
rf (dpkt.ip.IP attribute), 38
RFB (class in dpkt.rfb), 60
RIP (class in dpkt.rip), 62
rlen (dpkt.dns.DNS.RR attribute), 21
rlen (dpkt.netbios.NS.RR attribute), 44
root_id (dpkt.stp.STP attribute), 77
root_path (dpkt.stp.STP attribute), 77
route_tag (dpkt.rip.RTE attribute), 62
RouteGeneric (class in dpkt.bgp), 17
RouteIPv4 (class in dpkt.bgp), 17
RouteIPv6 (class in dpkt.bgp), 17
router (dpkt.ospf.OSPF attribute), 52
router_sc (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
RPC (class in dpkt.rpc), 63
RPC.Auth (class in dpkt.rpc), 63
RPC.Call (class in dpkt.rpc), 63
RPC.Reply (class in dpkt.rpc), 63
RPC.Reply.Accept (class in dpkt.rpc), 63
RPC.Reply.Reject (class in dpkt.rpc), 63
rpcvers (dpkt.rpc.RPC.Call attribute), 63
rsvd (dpkt.ah.AH attribute), 9
rsvd (dpkt.bgp.BGP.RouteRefresh attribute), 17
rsvd (dpkt.hsrp.HSRP attribute), 28
rsvd (dpkt.pim.PIM attribute), 54
rsvd (dpkt.rip.Auth attribute), 62
rsvd (dpkt.rip.RIP attribute), 62
rsvd (dpkt.sccp.SCCP attribute), 68
rsvd (dpkt.smb.SMB attribute), 70
rsvd (dpkt.tns.TNS attribute), 80
rsvd (dpkt.tpkt.TPKT attribute), 80
rsvd_sl_bits (dpkt.ip6.IP6RoutingHeader attribute), 40

RTE (class in dpkt.rip), 62
 RTP (class in dpkt.rtp), 64
 Rx (class in dpkt.rx), 64
 rx_flags_present (dpkt.radiotap.Radiotap attribute), 58

S

sa (dpkt.ethernet.VLANtagISL attribute), 25
 sa (dpkt.ieee80211.IEEE80211.DataInterDS attribute), 36
 safi (dpkt.bgp.BGP.RouteRefresh attribute), 17
 safi (dpkt.bgp.BGP.Update.Attribute.MPReachNLRI attribute), 16
 safi (dpkt.bgp.BGP.Update.Attribute.MPUreachNLRI attribute), 16
 SCCP (class in dpkt.sccp), 68
 scnt (dpkt.aoeata.AOEATA attribute), 12
 scnt (dpkt.aoeconfig.AOECONFIG attribute), 12
 SCTP (class in dpkt.sctp), 68
 secs (dpkt.dhcp.DHCP attribute), 19
 security (dpkt.rx.Rx attribute), 65
 security (dpkt.smb.SMB attribute), 70
 segs_left (dpkt.ip6.IP6RoutingHeader attribute), 40
 SelectStartKeys (class in dpkt.sccp), 67
 seq (dpkt.ah.AH attribute), 10
 seq (dpkt.aim.FLAP attribute), 10
 seq (dpkt.esp.ESP attribute), 24
 seq (dpkt.icmp.ICMP.Echo attribute), 30
 seq (dpkt.icmp6.ICMP6.Echo attribute), 31
 seq (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33
 seq (dpkt.ieee80211.IEEE80211.BlockAckReq attribute), 32
 seq (dpkt.ip6.IP6AHHeader attribute), 40
 seq (dpkt.ip6.IP6ESPHeader attribute), 41
 seq (dpkt.mrt.TableDump attribute), 43
 seq (dpkt.rtp.RTP attribute), 64
 seq (dpkt.rx.Rx attribute), 65
 seq (dpkt.tcp.TCP attribute), 78
 sequence (dpkt.qq.QQ3Packet attribute), 56
 sequence (dpkt.qq.QQ5Packet attribute), 57
 sequence (dpkt.qq.QQBasicPacket attribute), 56
 serial (dpkt.rx.Rx attribute), 65
 service (dpkt.rx.Rx attribute), 65
 service (dpkt.yahoo.YHOO attribute), 82
 Session (class in dpkt.netbios), 44
 session (dpkt.pppoe.PPPoE attribute), 56
 set (dpkt.radiotap.Radiotap.FHSS attribute), 58
 set_cmd() (dpkt.aoe.AOE class method), 11
 set_p() (dpkt.ppp.PPP class method), 55
 set_proto() (dpkt.ip.IP class method), 38
 set_proto() (dpkt.ip6.IP6 class method), 39
 set_type() (dpkt.ethernet.Ethernet class method), 24
 SetColourMapEntries (class in dpkt.rfb), 61
 SetEncodings (class in dpkt.rfb), 60
 setfilter() (dpkt.pcap.Reader method), 53
 setfilter() (dpkt.snoop.Reader method), 71
 SetLamp (class in dpkt.sccp), 67
 SetPixelFormat (class in dpkt.rfb), 60
 SetSpeakerMode (class in dpkt.sccp), 67
 setup_class() (dpkt.gzip.TestGzip class method), 27
 setup_class() (dpkt.ssl.TestClientHello class method), 75
 setup_class() (dpkt.ssl.TestServerHello class method), 75
 setup_class() (dpkt.ssl.TestTLS class method), 74
 setup_class() (dpkt.ssl.TestTLSCertificate class method), 76
 setup_class() (dpkt.ssl.TestTLSChangeCipherSpec class method), 75
 setup_class() (dpkt.ssl.TestTLSHandshake class method), 75
 setup_class() (dpkt.ssl.TestTLSMultiFactory class method), 76
 setup_class() (dpkt.ssl.TestTLSRecord class method), 74
 sha (dpkt.arp.ARP attribute), 13
 siaddr (dpkt.dhcp.DHCP attribute), 19
 sigfigs (dpkt.pcap.FileHdr attribute), 53
 sigfigs (dpkt.pcap.LEFileHdr attribute), 53
 silence_suppression (dpkt.sccp.StartMediaTransmission attribute), 67
 sl_bits (dpkt.ip6.IP6RoutingHeader attribute), 40
 SLL (class in dpkt.sll), 69
 SMB (class in dpkt.smb), 70
 SNAC (class in dpkt.aim), 10
 sname (dpkt.dhcp.DHCP attribute), 19
 snap (dpkt.ethernet.VLANtagISL attribute), 25
 snaplen (dpkt.pcap.FileHdr attribute), 53
 snaplen (dpkt.pcap.LEFileHdr attribute), 53
 softkey_map (dpkt.sccp.SelectStartKeys attribute), 67
 softkey_set (dpkt.sccp.SelectStartKeys attribute), 67
 source (dpkt.qq.QQ3Packet attribute), 56
 source (dpkt.qq.QQ5Packet attribute), 57
 source (dpkt.qq.QQBasicPacket attribute), 56
 spa (dpkt.arp.ARP attribute), 13
 speaker (dpkt.sccp.SetSpeakerMode attribute), 67
 spi (dpkt.ah.AH attribute), 10
 spi (dpkt.esp.ESP attribute), 24
 spi (dpkt.ip6.IP6AHHeader attribute), 40
 spi (dpkt.ip6.IP6ESPHeader attribute), 41
 sport (dpkt.netbios.Datagram attribute), 45
 sport (dpkt.sctp.SCTP attribute), 68
 sport (dpkt.tcp.TCP attribute), 78
 sport (dpkt.udp.UDP attribute), 81
 src (dpkt.ethernet.Ethernet attribute), 24
 src (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33
 src (dpkt.ieee80211.IEEE80211.BlockAckReq attribute), 32
 src (dpkt.ieee80211.IEEE80211.CFEnd attribute), 33
 src (dpkt.ieee80211.IEEE80211.Data attribute), 35
 src (dpkt.ieee80211.IEEE80211.DataFromDS attribute), 35

src (dpkt.ieee80211.IEEE80211.DataInterDS attribute), 36

src (dpkt.ieee80211.IEEE80211.DataToDS attribute), 35

src (dpkt.ieee80211.IEEE80211.MGMT_Frame attribute), 33

src (dpkt.ieee80211.IEEE80211.RTS attribute), 33

src (dpkt.ip.IP attribute), 38

src (dpkt.ip6.IP6 attribute), 39

src (dpkt.ipx.IPX attribute), 41

src (dpkt.netbios.Datagram attribute), 45

src_addr (dpkt.netflow.Netflow1.NetflowRecord attribute), 46

src_addr (dpkt.netflow.Netflow5.NetflowRecord attribute), 47

src_addr (dpkt.netflow.Netflow6.NetflowRecord attribute), 49

src_addr (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

src_as (dpkt.mrt.BGP4MPMessage attribute), 43

src_as (dpkt.mrt.BGP4MPMessage_32 attribute), 43

src_as (dpkt.netflow.Netflow5.NetflowRecord attribute), 47

src_as (dpkt.netflow.Netflow6.NetflowRecord attribute), 49

src_as (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

src_ip (dpkt.mrt.BGP4MPMessage attribute), 43

src_ip (dpkt.mrt.BGP4MPMessage_32 attribute), 43

src_mask (dpkt.netflow.Netflow5.NetflowRecord attribute), 47

src_mask (dpkt.netflow.Netflow6.NetflowRecord attribute), 49

src_mask (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

src_port (dpkt.netflow.Netflow1.NetflowRecord attribute), 46

src_port (dpkt.netflow.Netflow5.NetflowRecord attribute), 47

src_port (dpkt.netflow.Netflow6.NetflowRecord attribute), 49

src_port (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

sre (dpkt.gre.GRE attribute), 26

ssap (dpkt.llc.LLC attribute), 42

SSL2 (class in dpkt.ssl), 72

SSL3Exception, 72

SSLFactory (class in dpkt.ssl), 74

ssrc (dpkt.rtp.RTP attribute), 64

start_time (dpkt.netflow.Netflow1.NetflowRecord attribute), 46

start_time (dpkt.netflow.Netflow5.NetflowRecord attribute), 47

start_time (dpkt.netflow.Netflow6.NetflowRecord attribute), 49

start_time (dpkt.netflow.Netflow7.NetflowRecord attribute), 50

starting_seq (dpkt.ieee80211.IEEE80211.BlockAckActionRequest attribute), 35

StartMediaTransmission (class in dpkt.sccp), 67

StartTone (class in dpkt.sccp), 67

stat (dpkt.rpc.RPC.Reply attribute), 63

stat (dpkt.rpc.RPC.Reply.Accept attribute), 63

stat (dpkt.rpc.RPC.Reply.Reject attribute), 63

state (dpkt.hsrp.HSRP attribute), 28

status (dpkt.ieee80211.IEEE80211.Assoc_Resp attribute), 34

status (dpkt.mrt.TableDump attribute), 43

status (dpkt.rx.Rx attribute), 65

status (dpkt.smb.SMB attribute), 70

status_code (dpkt.ieee80211.IEEE80211.BlockAckActionResponse attribute), 35

stimulus (dpkt.sccp.SetLamp attribute), 67

stimulus_instance (dpkt.sccp.SetLamp attribute), 67

StopMediaTransmission (class in dpkt.sccp), 67

STP (class in dpkt.stp), 77

stratum (dpkt.ntp.NTP attribute), 51

strip_options() (in module dpkt.telnet), 79

STUN (class in dpkt.stun), 78

subcode (dpkt.bgp.BGP.Notification attribute), 16

subnet (dpkt.rip.RTE attribute), 62

subtype (dpkt.aim.SNAC attribute), 10

subtype (dpkt.ieee80211.IEEE80211 attribute), 32

subtype (dpkt.mrt.MRTHeader attribute), 42

sum (dpkt.cdp.CDP attribute), 18

sum (dpkt.icmp.ICMP attribute), 30

sum (dpkt.icmp6.ICMP6 attribute), 32

sum (dpkt.igmp.IGMP attribute), 38

sum (dpkt.ip.IP attribute), 38

sum (dpkt.ipx.IPX attribute), 41

sum (dpkt.ospf.OSPF attribute), 52

sum (dpkt.pim.PIM attribute), 54

sum (dpkt.rx.Rx attribute), 65

sum (dpkt.sctp.SCTP attribute), 68

sum (dpkt.tcp.TCP attribute), 79

sum (dpkt.udp.UDP attribute), 81

sum (dpkt.vrrp.VRRP attribute), 81

sys_uptime (dpkt.netflow.Netflow1 attribute), 46

sys_uptime (dpkt.netflow.Netflow5 attribute), 48

sys_uptime (dpkt.netflow.Netflow6 attribute), 49

sys_uptime (dpkt.netflow.Netflow7 attribute), 50

sys_uptime (dpkt.netflow.NetflowBase attribute), 45

T

TableDump (class in dpkt.mrt), 42

tag (dpkt.aoe.AOE attribute), 11

tc (dpkt.dns.DNS attribute), 21

tc (dpkt.ipx.IPX attribute), 41

TCP (class in dpkt.tcp), 78

tcp_flags	(dpkt.netflow.Netflow1.NetflowRecord attribute), 46	at-	(dpkt.decorators.TestDeprecatedDecorator method), 18
tcp_flags	(dpkt.netflow.Netflow5.NetflowRecord attribute), 47	at-	test_dhcp() (in module dpkt.dhcp), 19
tcp_flags	(dpkt.netflow.Netflow6.NetflowRecord attribute), 49	at-	test_eth() (in module dpkt.ethernet), 25
tcp_flags	(dpkt.netflow.Netflow7.NetflowRecord attribute), 50	at-	test_eth_802dot1q() (in module dpkt.ethernet), 25
			test_eth_802dot1q_stacked() (in module dpkt.ethernet), 25
test()	(in module examples.print_http_requests), 8		test_eth_init_with_data() (in module dpkt.ethernet), 25
test()	(in module examples.print_icmp), 6		test_eth_llc_ipx() (in module dpkt.ethernet), 25
test()	(in module examples.print_packets), 5		test_eth_llc_snap_cdp() (in module dpkt.ethernet), 25
test_80211_beacon()	(in module dpkt.ieee80211), 37		test_eth_mpls_stacked() (in module dpkt.ethernet), 25
test_80211_data()	(in module dpkt.ieee80211), 37		test_eth_pppoe() (in module dpkt.ethernet), 25
test_80211_data_qos()	(in module dpkt.ieee80211), 37		test_fcs() (in module dpkt.radiotap), 59
test_802211_ack()	(in module dpkt.ieee80211), 37		test_filename() (dpkt.gzip.TestGzip method), 27
test_802dot1q_tag()	(in module dpkt.ethernet), 25		test_first_msg_data() (dpkt.ssl.TestTLSTLSMultiFactory method), 76
test_action_block_ack_request()	(in module dpkt.ieee80211), 37	test_flags()	(dpkt.gzip.TestGzip method), 27
test_action_block_ack_response()	(in module dpkt.ieee80211), 37	test_format_request()	(in module dpkt.http), 29
test_aoeata()	(in module dpkt.aoeata), 12	test_frag()	(in module dpkt.ip), 39
test_aoecfg()	(in module dpkt.aoecfg), 12	test_gre_v1()	(in module dpkt.gre), 26
test_asn1()	(in module dpkt.asn1), 13	test_gzip_response()	(in module dpkt.http), 29
test_auth()	(dpkt.ssl.ciphersuites.TestCipherSuites method), 77	test_hl()	(in module dpkt.ip), 39
test_basic()	(in module dpkt.dns), 22	test_icmp()	(in module dpkt.icmp), 30
test_body_forbidden_response()	(in module dpkt.http), 29	test_initial_flags()	(dpkt.ssl.TestTLSRecord method), 75
test_bug()	(in module dpkt.ieee80211), 37	test_invalid_header()	(in module dpkt.http), 29
test_bytes_parsed()	(dpkt.ssl.TestTLSTLSMultiFactory method), 76	test_ip()	(in module dpkt.ip), 38
test_chunked_response()	(in module dpkt.http), 29	test_ip6_ah_header()	(in module dpkt.ip6), 41
test_cipher_suite()	(dpkt.ssl.TestServerHello method), 75	test_ip6_esp_header()	(in module dpkt.ip6), 41
test_cipher_suite_length()	(dpkt.ssl.TestClientHello method), 75	test_ip6_extension_headers()	(in module dpkt.ip6), 41
test_circular_pointers()	(in module dpkt.dns), 22	test_ip6_fragment_header()	(in module dpkt.ip6), 41
test_client_hello_constructed()	(dpkt.ssl.TestClientHello method), 75	test_ip6_options_header()	(in module dpkt.ip6), 41
test_client_random_correct()	(dpkt.ssl.TestClientHello method), 75	test_ip6_routing_header()	(in module dpkt.ip6), 41
test_compressed_block_ack()	(in module dpkt.ieee80211), 37	test_ipg()	(in module dpkt.ip6), 41
test_compression_methods()	(dpkt.ssl.TestClientHello method), 75	test_isl_eth_llc_stp()	(in module dpkt.ethernet), 25
test_constructed()	(dpkt.ssl.TestServerHello method), 75	test_isl_tag()	(in module dpkt.ethernet), 25
test_constructor()	(in module dpkt.ip), 39	test_kx()	(dpkt.ssl.ciphersuites.TestCipherSuites method), 77
test_content_type()	(dpkt.ssl.TestTLSRecord method), 74	test_length()	(dpkt.ssl.TestTLSHandshake method), 75
test_crc32c()	(in module dpkt.crc32c), 18	test_length()	(dpkt.ssl.TestTLSRecord method), 74
test_created_inside_message()	(dpkt.ssl.TestTLSHandshake method), 75	test_llc()	(in module dpkt.llc), 42
test_data()	(dpkt.ssl.TestTLSRecord method), 74	test_method()	(dpkt.gzip.TestGzip method), 27
test_data_ds()	(in module dpkt.ieee80211), 37	test_mpls_label()	(in module dpkt.ethernet), 25
test_decompress()	(dpkt.gzip.TestGzip method), 27	test_mtime()	(dpkt.gzip.TestGzip method), 27
test_deprecated_decorator()		test_multicookie_response()	(in module dpkt.http), 29
		test_name()	(dpkt.ssl.ciphersuites.TestCipherSuites method), 77
		test_net_flow_v1_pack()	(in module dpkt.netflow), 50
		test_net_flow_v1_unpack()	(in module dpkt.netflow), 50
		test_net_flow_v5_pack()	(in module dpkt.netflow), 50
		test_net_flow_v5_unpack()	(in module dpkt.netflow), 50
		test_noreason_response()	(in module dpkt.http), 29
		test_ntp_pack()	(in module dpkt.ntp), 51
		test_ntp_unpack()	(in module dpkt.ntp), 51
		test_null_response()	(in module dpkt.dns), 22

test_num_certs() (dpkt.ssl.TestTLSCertificate method), 76

test_num_messages() (dpkt.ssl.TestTLSMultiFactory method), 76

test_offset() (in module dpkt.tcp), 79

test_op_data() (in module dpkt.tftp), 79

test_op_err() (in module dpkt.tftp), 79

test_op_rrq() (in module dpkt.tftp), 79

test_OPT() (in module dpkt.dns), 22

test_opt() (in module dpkt.ip), 39

test_os() (dpkt.gzip.TestGzip method), 27

test_pack() (in module dpkt.bgp), 17

test_pack() (in module dpkt.diameter), 20

test_pack() (in module dpkt.h225), 27

test_pack_name() (in module dpkt.dns), 22

test_packing() (in module dpkt.ppp), 55

test_parse_opts() (in module dpkt.tcp), 79

test_parse_request() (in module dpkt.http), 29

test_parses() (dpkt.ssl.TestTLSChangeCipherSpec method), 75

test_pcap_endian() (in module dpkt.pcap), 54

test_pim() (in module dpkt.pim), 54

test_ppp() (in module dpkt.ppp), 55

test_ppp_packing() (in module dpkt.pppoe), 56

test_ppp_short() (in module dpkt.ppp), 55

test_ppp_short() (in module dpkt.pppoe), 56

test_pppoe_discovery() (in module dpkt.pppoe), 56

test_pppoe_session() (in module dpkt.pppoe), 56

test_PTR() (in module dpkt.dns), 22

test_Radiotap() (in module dpkt.radiotap), 59

test_raises_need_data() (dpkt.ssl.TestTLSHandshake method), 75

test_raises_need_data_when_buf_is_short() (dpkt.ssl.TestTLSRecord method), 75

test_random_correct() (dpkt.ssl.TestServerHello method), 75

test_random_data() (in module dpkt.dns), 22

test_reader() (in module dpkt.pcap), 54

test_record_type() (dpkt.ssl.TestTLS method), 74

test_record_version() (dpkt.ssl.TestTLS method), 74

test_records_length() (dpkt.ssl.TestTLS method), 74

test_repack() (dpkt.ssl.TestTLSRecord method), 75

test_request_version() (in module dpkt.http), 29

test_rtp() (in module dpkt.rtp), 64

test_rtp_pack() (in module dpkt.rtp), 62

test_rtp_unpack() (in module dpkt.rtp), 62

test_sctp_pack() (in module dpkt.sctp), 69

test_sctp_unpack() (in module dpkt.sctp), 69

test_second_msg_data() (dpkt.ssl.TestTLSMultiFactory method), 76

test_session_id() (dpkt.ssl.TestClientHello method), 75

test_sll() (in module dpkt.sll), 70

test_smb() (in module dpkt.smb), 70

test_stp() (in module dpkt.stp), 77

test_stun_padded() (in module dpkt.stun), 78

test_stun_response() (in module dpkt.stun), 78

test_telnet() (in module dpkt.telnet), 79

test_tns() (in module dpkt.tns), 80

test_total_length() (dpkt.ssl.TestClientHello method), 75

test_total_length() (dpkt.ssl.TestServerHello method), 76

test_total_length() (dpkt.ssl.TestTLSChangeCipherSpec method), 75

test_total_length() (dpkt.ssl.TestTLSRecord method), 75

test_txt_response() (in module dpkt.dns), 22

test_unpack() (in module dpkt.bgp), 17

test_unpack() (in module dpkt.diameter), 20

test_unpack() (in module dpkt.h225), 28

test_utils() (in module dpkt.dpkt), 23

test_value() (dpkt.ssl.TestTLSAppData method), 75

test_version() (dpkt.ssl.TestTLSRecord method), 74

test_very_long_name() (in module dpkt.dns), 22

test_vrrp() (in module dpkt.vrrp), 81

test_writer_precision() (in module dpkt.pcap), 54

test_xflags() (dpkt.gzip.TestGzip method), 27

test_zerolen() (in module dpkt.ip), 39

testAIM() (in module dpkt.aim), 10

TestCipherSuites (class in dpkt.ssl_ciphersuites), 77

TestClientHello (class in dpkt.ssl), 75

TestDeprecatedDecorator (class in dpkt.decorators), 18

testExceptions() (in module dpkt.aim), 10

TestGzip (class in dpkt.gzip), 27

TestServerHello (class in dpkt.ssl), 75

TestTLS (class in dpkt.ssl), 74

TestTLSAppData (class in dpkt.ssl), 75

TestTLSCertificate (class in dpkt.ssl), 76

TestTLSChangeCipherSpec (class in dpkt.ssl), 75

TestTLSHandshake (class in dpkt.ssl), 75

TestTLSMultiFactory (class in dpkt.ssl), 76

TestTLSRecord (class in dpkt.ssl), 74

TFTP (class in dpkt.tftp), 79

tha (dpkt.arp.ARP attribute), 13

thiszone (dpkt.pcap.FileHdr attribute), 53

thiszone (dpkt.pcap.LEFileHdr attribute), 53

tid (dpkt.ieee80211.IEEE80211.BlockAck attribute), 33

tid (dpkt.smb.SMB attribute), 70

timeout (dpkt.ieee80211.IEEE80211.BlockAckActionRequest attribute), 35

timeout (dpkt.ieee80211.IEEE80211.BlockAckActionResponse attribute), 35

timestamp (dpkt.ieee80211.IEEE80211.Beacon attribute), 34

TLS (class in dpkt.ssl), 72

tls_multi_factory() (in module dpkt.ssl), 74

TLAlert (class in dpkt.ssl), 73

TLSAppData (class in dpkt.ssl), 73

TLSCertificate (class in dpkt.ssl), 73

TLSCertificateRequest (in module dpkt.ssl), 73

TLSCertificateVerify (in module dpkt.ssl), 73

- TLSCipherSpec (class in dpkt.ssl), 72
 - TLSCClientHello (class in dpkt.ssl), 73
 - TLSCClientKeyExchange (in module dpkt.ssl), 73
 - TLSCFinished (in module dpkt.ssl), 74
 - TLShHandshake (class in dpkt.ssl), 74
 - TLShHelloRequest (class in dpkt.ssl), 73
 - TLSSRecord (class in dpkt.ssl), 72
 - TLSSServerHello (class in dpkt.ssl), 73
 - TLSSServerHelloDone (in module dpkt.ssl), 73
 - TLSSServerKeyExchange (in module dpkt.ssl), 73
 - TLSSUnknownHandshake (class in dpkt.ssl), 73
 - tlv() (in module dpkt.aim), 10
 - tlv() (in module dpkt.stun), 78
 - TNS (class in dpkt.tns), 79
 - to_ds (dpkt.ieee80211.IEEE80211 attribute), 32
 - tone (dpkt.sccp.StartTone attribute), 67
 - tos (dpkt.ip.IP attribute), 38
 - tos (dpkt.netflow.Netflow1.NetflowRecord attribute), 46
 - tos (dpkt.netflow.Netflow5.NetflowRecord attribute), 47
 - tos (dpkt.netflow.Netflow6.NetflowRecord attribute), 49
 - tos (dpkt.netflow.Netflow7.NetflowRecord attribute), 50
 - tpa (dpkt.arp.ARP attribute), 13
 - TPKT (class in dpkt.tpkt), 80
 - transitive (dpkt.bgp.BGP.Update.Attribute attribute), 14
 - transmit_time (dpkt.ntp.NTP attribute), 51
 - ts (dpkt.mrt.MRTHeader attribute), 42
 - ts (dpkt.rtp.RTP attribute), 64
 - ts_sec (dpkt.snoop.PktHdr attribute), 71
 - ts_usec (dpkt.snoop.PktHdr attribute), 71
 - tsft_present (dpkt.radiotap.Radiotap attribute), 57
 - ttl (dpkt.cdp.CDP attribute), 18
 - ttl (dpkt.dns.DNS.RR attribute), 21
 - ttl (dpkt.ip.IP attribute), 38
 - ttl (dpkt.netbios.NS.RR attribute), 44
 - tu (dpkt.ieee80211.IEEE80211.FH attribute), 36
 - tv_sec (dpkt.pcap.LEPktHdr attribute), 52
 - tv_sec (dpkt.pcap.PktHdr attribute), 52
 - tv_usec (dpkt.pcap.LEPktHdr attribute), 52
 - tv_usec (dpkt.pcap.PktHdr attribute), 52
 - tx_attn_present (dpkt.radiotap.Radiotap attribute), 57
 - type (dpkt.aim.FLAP attribute), 10
 - type (dpkt.bgp.BGP attribute), 17
 - type (dpkt.bgp.BGP.Open.Parameter attribute), 14
 - type (dpkt.bgp.BGP.Update.Attribute attribute), 16
 - type (dpkt.bgp.BGP.Update.Attribute.ASPath.ASPathSegment attribute), 15
 - type (dpkt.bgp.BGP.Update.Attribute.Origin attribute), 14
 - type (dpkt.cdp.CDP.TLV attribute), 18
 - type (dpkt.dns.DNS.Q attribute), 21
 - type (dpkt.dns.DNS.RR attribute), 21
 - type (dpkt.ethernet.Ethernet attribute), 24
 - type (dpkt.ethernet.VLANtag8021Q attribute), 25
 - type (dpkt.h225.H225.IE attribute), 27
 - type (dpkt.icmp.ICMP attribute), 30
 - type (dpkt.icmp6.ICMP6 attribute), 32
 - type (dpkt.ieee80211.IEEE80211 attribute), 32
 - type (dpkt.igmp.IGMP attribute), 38
 - type (dpkt.ip6.IP6RoutingHeader attribute), 40
 - type (dpkt.mrt.MRTHeader attribute), 42
 - type (dpkt.netbios.Datagram attribute), 45
 - type (dpkt.netbios.NS.Q attribute), 44
 - type (dpkt.netbios.NS.RR attribute), 44
 - type (dpkt.netbios.Session attribute), 44
 - type (dpkt.ospf.OSPF attribute), 52
 - type (dpkt.pim.PIM attribute), 54
 - type (dpkt.pppoe.PPPoE attribute), 55
 - type (dpkt.rfb.RFB attribute), 60
 - type (dpkt.rip.Auth attribute), 62
 - type (dpkt.rx.Rx attribute), 65
 - type (dpkt.sctp.Chunk attribute), 69
 - type (dpkt.sll.SLL attribute), 69
 - type (dpkt.ssl.TLS attribute), 72
 - type (dpkt.ssl.TLSCipherSpec attribute), 73
 - type (dpkt.ssl.TLShHandshake attribute), 74
 - type (dpkt.ssl.TLSSRecord attribute), 72
 - type (dpkt.stp.STP attribute), 77
 - type (dpkt.stun.STUN attribute), 78
 - type (dpkt.tns.TNS attribute), 80
 - type (dpkt.vrrp.VRRP attribute), 81
 - type (dpkt.yahoo.YHOO attribute), 82
 - type (dpkt.yahoo.YMSG attribute), 82
- ## U
- UDP (class in dpkt.udp), 80
 - uid (dpkt.smb.SMB attribute), 70
 - ulen (dpkt.udp.UDP attribute), 81
 - unix_nsec (dpkt.netflow.Netflow1 attribute), 46
 - unix_nsec (dpkt.netflow.Netflow5 attribute), 48
 - unix_nsec (dpkt.netflow.Netflow6 attribute), 49
 - unix_nsec (dpkt.netflow.Netflow7 attribute), 50
 - unix_nsec (dpkt.netflow.NetflowBase attribute), 45
 - unix_sec (dpkt.netflow.Netflow1 attribute), 46
 - unix_sec (dpkt.netflow.Netflow5 attribute), 48
 - unix_sec (dpkt.netflow.Netflow6 attribute), 49
 - unix_sec (dpkt.netflow.Netflow7 attribute), 50
 - unix_sec (dpkt.netflow.NetflowBase attribute), 45
 - unknown (dpkt.qq.QQ5Packet attribute), 57
 - unknown (dpkt.yahoo.YHOO attribute), 82
 - unknown1 (dpkt.qq.QQ3Packet attribute), 56
 - unknown1 (dpkt.yahoo.YMSG attribute), 82
 - unknown10 (dpkt.qq.QQ3Packet attribute), 56
 - unknown11 (dpkt.qq.QQ3Packet attribute), 56
 - unknown12 (dpkt.qq.QQ3Packet attribute), 56
 - unknown13 (dpkt.qq.QQ3Packet attribute), 56
 - unknown2 (dpkt.qq.QQ3Packet attribute), 56
 - unknown2 (dpkt.yahoo.YMSG attribute), 82
 - unknown3 (dpkt.qq.QQ3Packet attribute), 57
 - unknown4 (dpkt.qq.QQ3Packet attribute), 57

unknown5 (dpkt.qq.QQ3Packet attribute), 57
 unknown6 (dpkt.qq.QQ3Packet attribute), 57
 unknown7 (dpkt.qq.QQ3Packet attribute), 57
 unknown8 (dpkt.qq.QQ3Packet attribute), 57
 unknown9 (dpkt.qq.QQ3Packet attribute), 57
 unpack() (dpkt.ah.AH method), 9
 unpack() (dpkt.aim.FLAP method), 10
 unpack() (dpkt.aoe.AOE method), 11
 unpack() (dpkt.bgp.BGP method), 13
 unpack() (dpkt.bgp.BGP.Keepalive method), 16
 unpack() (dpkt.bgp.BGP.Notification method), 16
 unpack() (dpkt.bgp.BGP.Open method), 14
 unpack() (dpkt.bgp.BGP.Open.Parameter method), 14
 unpack() (dpkt.bgp.BGP.Open.Parameter.Capability method), 14
 unpack() (dpkt.bgp.BGP.Update method), 14
 unpack() (dpkt.bgp.BGP.Update.Attribute method), 14
 unpack() (dpkt.bgp.BGP.Update.Attribute.ASPath method), 15
 unpack() (dpkt.bgp.BGP.Update.Attribute.ASPath.ASPathSegment method), 15
 unpack() (dpkt.bgp.BGP.Update.Attribute.AtomicAggregate method), 15
 unpack() (dpkt.bgp.BGP.Update.Attribute.ClusterList method), 16
 unpack() (dpkt.bgp.BGP.Update.Attribute.Communities method), 15
 unpack() (dpkt.bgp.BGP.Update.Attribute.MPReachNLRI method), 16
 unpack() (dpkt.bgp.BGP.Update.Attribute.MPReachNLRI.SNPA method), 16
 unpack() (dpkt.bgp.BGP.Update.Attribute.MPUnreachNLRI method), 16
 unpack() (dpkt.bgp.RouteGeneric method), 17
 unpack() (dpkt.bgp.RouteIPV4 method), 17
 unpack() (dpkt.bgp.RouteIPV6 method), 17
 unpack() (dpkt.cdp.CDP method), 18
 unpack() (dpkt.cdp.CDP.Address method), 18
 unpack() (dpkt.cdp.CDP.TLV method), 18
 unpack() (dpkt.dhcp.DHCP method), 19
 unpack() (dpkt.diameter.AVP method), 20
 unpack() (dpkt.diameter.Diameter method), 20
 unpack() (dpkt.dns.DNS method), 22
 unpack() (dpkt.dns.DNS.Q method), 21
 unpack() (dpkt.dpkt.Packet method), 23
 unpack() (dpkt.dtp.DTP method), 23
 unpack() (dpkt.ethernet.Ethernet method), 24
 unpack() (dpkt.ethernet.MPLSlabel method), 24
 unpack() (dpkt.ethernet.VLANtag8021Q method), 24
 unpack() (dpkt.ethernet.VLANtagISL method), 25
 unpack() (dpkt.gre.GRE method), 26
 unpack() (dpkt.gre.GRE.SRE method), 26
 unpack() (dpkt.gzip.Gzip method), 26
 unpack() (dpkt.h225.H225 method), 27
 unpack() (dpkt.h225.H225.IE method), 27
 unpack() (dpkt.http.Message method), 29
 unpack() (dpkt.http.Request method), 29
 unpack() (dpkt.http.Response method), 29
 unpack() (dpkt.icmp.ICMP method), 30
 unpack() (dpkt.icmp.ICMP.Quote method), 30
 unpack() (dpkt.icmp6.ICMP6 method), 31
 unpack() (dpkt.icmp6.ICMP6.Error method), 31
 unpack() (dpkt.ieee80211.IEEE80211 method), 32
 unpack() (dpkt.ieee80211.IEEE80211.Action method), 34
 unpack() (dpkt.ieee80211.IEEE80211.BlockAck method), 33
 unpack() (dpkt.ieee80211.IEEE80211.IE method), 36
 unpack() (dpkt.ieee80211.IEEE80211.TIM method), 37
 unpack() (dpkt.ip.IP method), 38
 unpack() (dpkt.ip6.IP6 method), 39
 unpack() (dpkt.ip6.IP6AHHeader method), 40
 unpack() (dpkt.ip6.IP6ESPHeader method), 41
 unpack() (dpkt.ip6.IP6FragmentHeader method), 40
 unpack() (dpkt.ip6.IP6OptsHeader method), 39
 unpack() (dpkt.ip6.IP6RoutingHeader method), 40
 unpack() (dpkt.llc.LLC method), 42
 unpack() (dpkt.loopback.Loopback method), 42
 unpack() (dpkt.mrt.TableDump method), 42
 unpack() (dpkt.netflow.NetflowBase method), 45
 unpack() (dpkt.netflow.NetflowBase.NetflowRecordBase method), 45
 unpack() (dpkt.ppp.PPP method), 55
 unpack() (dpkt.pppoe.PPP method), 56
 unpack() (dpkt.pppoe.PPPoE method), 55
 unpack() (dpkt.radiotap.Radiotap method), 58
 unpack() (dpkt.radius.RADIUS method), 60
 unpack() (dpkt.rip.RIP method), 62
 unpack() (dpkt.rpc.RPC method), 63
 unpack() (dpkt.rpc.RPC.Auth method), 63
 unpack() (dpkt.rpc.RPC.Call method), 63
 unpack() (dpkt.rpc.RPC.Reply method), 63
 unpack() (dpkt.rpc.RPC.Reply.Accept method), 63
 unpack() (dpkt.rpc.RPC.Reply.Reject method), 63
 unpack() (dpkt.rtp.RTP method), 64
 unpack() (dpkt.sccp.SCCP method), 68
 unpack() (dpkt.sctp.Chunk method), 68
 unpack() (dpkt.sctp.SCTP method), 68
 unpack() (dpkt.sll.SLL method), 69
 unpack() (dpkt.ssl.SSL2 method), 72
 unpack() (dpkt.ssl.TLS method), 72
 unpack() (dpkt.ssl.TLSCertificate method), 73
 unpack() (dpkt.ssl.TLSClientHello method), 73
 unpack() (dpkt.ssl.TLSHandshake method), 74
 unpack() (dpkt.ssl.TLSRecord method), 72
 unpack() (dpkt.ssl.TLSServerHello method), 73
 unpack() (dpkt.tcp.TCP method), 78
 unpack() (dpkt.tftp.TFTP method), 79

unpack() (dpkt.tns.TNS method), 80
 unpack() (dpkt.vrrp.VRRP method), 81
 unpack_ies() (dpkt.ieee80211.IEEE80211 method), 32
 unpack_name() (dpkt.netbios.NS method), 44
 unpack_name() (in module dpkt.dns), 20
 unpack_q() (dpkt.dns.DNS method), 21
 unpack_rdata() (dpkt.dns.DNS.RR method), 21
 unpack_rdata() (dpkt.netbios.NS.RR method), 44
 unpack_rr() (dpkt.dns.DNS method), 22
 unpack_xdrlist() (in module dpkt.rpc), 64
 UnpackError, 22
 update_time (dpkt.ntp.NTP attribute), 51
 urp (dpkt.tcp.TCP attribute), 79
 usecs (dpkt.radiotap.Radiotap.TSFT attribute), 59
 utctime() (in module dpkt.asn1), 13

V

v (dpkt.bgp.BGP.Open attribute), 14
 v (dpkt.diameter.Diameter attribute), 20
 v (dpkt.dtp.DTP attribute), 23
 v (dpkt.gre.GRE attribute), 26
 v (dpkt.ip.IP attribute), 38
 v (dpkt.ip6.IP6 attribute), 39
 v (dpkt.ntp.NTP attribute), 51
 v (dpkt.ospf.OSPF attribute), 52
 v (dpkt.pim.PIM attribute), 54
 v (dpkt.pppoe.PPPoE attribute), 55
 v (dpkt.rip.RIP attribute), 62
 v (dpkt.snoop.FileHdr attribute), 71
 v (dpkt.stp.STP attribute), 77
 v (dpkt.tpkt.TPKT attribute), 80
 v (dpkt.vrrp.VRRP attribute), 81
 v_major (dpkt.pcap.FileHdr attribute), 53
 v_major (dpkt.pcap.LEFileHdr attribute), 53
 v_minor (dpkt.pcap.FileHdr attribute), 53
 v_minor (dpkt.pcap.LEFileHdr attribute), 53
 val (dpkt.radiotap.Radiotap.Flags attribute), 58
 val (dpkt.radiotap.Radiotap.LockQuality attribute), 58
 val (dpkt.radiotap.Radiotap.Rate attribute), 59
 val (dpkt.radiotap.Radiotap.RxFlags attribute), 59
 val (dpkt.radiotap.Radiotap.TxAttenuation attribute), 59
 value (dpkt.bgp.BGP.Update.Attribute.Communities.Community attribute), 15
 value (dpkt.bgp.BGP.Update.Attribute.Communities.ReservedCommunity attribute), 16
 value (dpkt.bgp.BGP.Update.Attribute.LocalPref attribute), 15
 value (dpkt.bgp.BGP.Update.Attribute.MultiExitDisc attribute), 15
 value (dpkt.bgp.BGP.Update.Attribute.OriginatorID attribute), 16
 vendor_flag (dpkt.diameter.AVP attribute), 20
 ver (dpkt.aoe.AOE attribute), 11
 ver_fl (dpkt.aoe.AOE attribute), 11

vers (dpkt.pmap.Pmap attribute), 55
 vers (dpkt.rpc.RPC.Call attribute), 63
 version (dpkt.cdp.CDP attribute), 18
 version (dpkt.hsrp.HSRP attribute), 28
 version (dpkt.ieee80211.IEEE80211 attribute), 32
 version (dpkt.netflow.Netflow1 attribute), 47
 version (dpkt.netflow.Netflow5 attribute), 48
 version (dpkt.netflow.Netflow6 attribute), 49
 version (dpkt.netflow.Netflow7 attribute), 50
 version (dpkt.netflow.NetflowBase attribute), 45
 version (dpkt.radiotap.Radiotap attribute), 59
 version (dpkt.rtp.RTP attribute), 64
 version (dpkt.ssl.TLS attribute), 72
 version (dpkt.ssl.TLSClientHello attribute), 73
 version (dpkt.ssl.TLSRecord attribute), 72
 version (dpkt.ssl.TLSServerHello attribute), 73
 version (dpkt.yahoo.YHOO attribute), 82
 version (dpkt.yahoo.YMSG attribute), 82
 view (dpkt.mrt.TableDump attribute), 43
 vip (dpkt.hsrp.HSRP attribute), 28
 VLANtag8021Q (class in dpkt.ethernet), 24
 VLANtagISL (class in dpkt.ethernet), 25
 vrid (dpkt.vrrp.VRRP attribute), 81
 VRRP (class in dpkt.vrrp), 81
 vtag (dpkt.sctp.SCTP attribute), 68

W

wep (dpkt.ieee80211.IEEE80211 attribute), 32
 width (dpkt.rfb.FramebufferUpdateRequest attribute), 61
 win (dpkt.tcp.TCP attribute), 79
 writepkt() (dpkt.pcap.Writer method), 53
 writepkt() (dpkt.snoop.Writer method), 71
 Writer (class in dpkt.pcap), 53
 Writer (class in dpkt.snoop), 71

X

x (dpkt.rtp.RTP attribute), 64
 x_position (dpkt.rfb.FramebufferUpdateRequest attribute), 61
 x_position (dpkt.rfb.PointerEvent attribute), 61
 xflags (dpkt.gzip.Gzip attribute), 27
 xid (dpkt.dhcp.DHCP attribute), 19
 xid (dpkt.rpc.RPC attribute), 64
 xid (dpkt.stun.STUN attribute), 78

Y

y_position (dpkt.rfb.FramebufferUpdateRequest attribute), 61
 y_position (dpkt.rfb.PointerEvent attribute), 61
 YHOO (class in dpkt.yahoo), 81
 yiaddr (dpkt.dhcp.DHCP attribute), 19
 YMSG (class in dpkt.yahoo), 82

Z

zero (dpkt.dns.DNS attribute), [21](#)