Network Working Group
Internet-Draft
Intended status: Informational
Expires: 25 December 2021

G. Harris, Ed.

M. Richardson
Sandelman

23 June 2021

PCAP Capture File Format
draft-gharris-opsawg-pcap-02

Abstract

This document describes the format used by the libpcap library to record captured packets to a file. Programs using the libpcap library to read and write those files, and thus reading and writing files in that format, include tcpdump.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the OPSAWG Working Group mailing list (opsawg@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/opsawg/.

Source for this draft and an issue tracker can be found at https://github.com/pcapng/pcapng.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 25 December 2021.

Table of Contents

1. Introduction

In the late 1980's, Van Jacobson, Steve McCanne, and others at the Network Research Group at Lawrence Berkeley National Laboratory developed the tcpdump program to capture and dissect network traces. The code to capture traffic, using low-level mechanisms in various operating systems, and to read and write network traces to a file was later put into a library named libpcap.

This document describes the format used by tcpdump, and other programs using libpcap, to read and write network traces.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. General File Structure

A capture file begins with a File Header, followed by zero or more Packet Records, one per packet.

All fields in the File Header and in Packet Records will always be saved according to the characteristics (little endian / big endian) of the capturing machine. This refers to all the fields that are saved as numbers and that span over two or more octets.

The approach of having the file saved in the native format of the generating host is more efficient because it avoids translation of data when reading / writing on the host itself, which is the most common case when generating/processing capture captures.

The packets are shown in traditional IETF diagram, with the bits numbered from the left to the right. The bit numbering does not reflect the binary value position, as IETF protocols are traditionally in big-endian network-byte order. The most significant bit is therefore on the left in this diagram as if the file is being stored on a big-endian system.

4. File Header

The File Header has the following format:

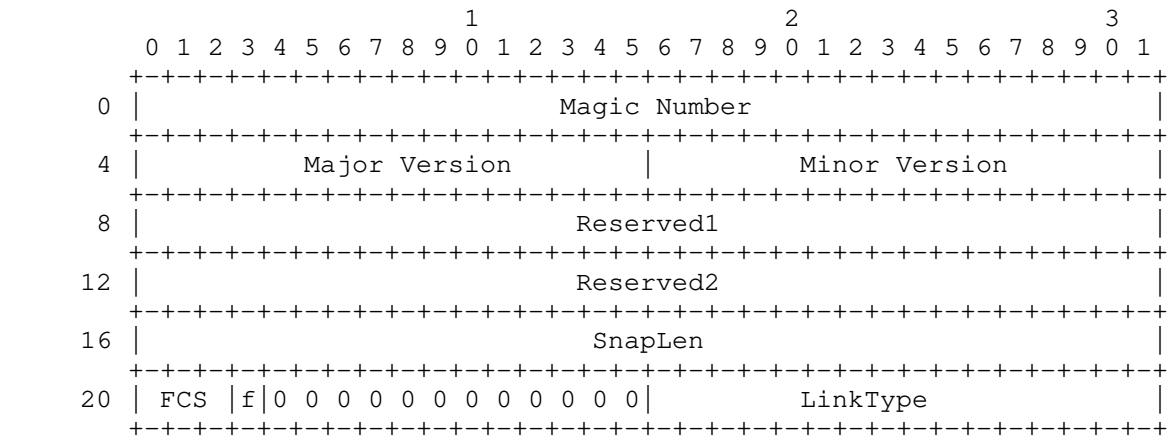

Figure 1: File Header

The File Header length is 24 octets.

The meaning of the fields in the File Header is:

Magic Number (32 bits): an unsigned magic number, whose value is either the hexadecimal number 0xA1B2C3D4 or the hexadecimal number 0xA1B23C4D.

If the value is 0xA1B2C3D4, time stamps in Packet Records (see Figure 2) are in seconds and microseconds; if it is 0xA1B23C4D, time stamps in Packet Records are in seconds and nanoseconds.

These numbers can be used to distinguish sessions that have been saved on little-endian machines from the ones saved on big-endian machines, and to heuristically identify pcap files.

Major Version (16 bits): an unsigned value, giving the number of the current major version of the format. The value for the current version of the format is 2. This value should change if the format changes in such a way that code that reads the new format could not read the old format (i.e., code to read both formats would have to check the version number and use different code paths for the two formats) and code that reads the old format could not read the new format.

Minor Version (16 bits): an unsigned value, giving the number of the

current minor version of the format. The value is for the current version of the format is 4. This value should change if the format changes in such a way that code that reads the new format could read the old format without checking the version number but code that reads the old format could not read all files in the new format.

Reserved1 (32 bits): not used - SHOULD be filled with 0 by pcap file writers, and MUST be ignored by pcap file readers. This value was documented by some older implementations as "gmt to local correction". Some older pcap file writers stored non-zero values in this field.

Reserved2 (32 bits): not used - SHOULD be filled with 0 by pcap file writers, and MUST be ignored by pcap file readers. This value was documented by some older implementations as "accuracy of timestamps". Some older pcap file writers stored non-zero values in this field.

SnapLen (32 bits): an unsigned value indicating the maximum number of octets captured from each packet. The portion of each packet that exceeds this value will not be stored in the file. This value MUST NOT be zero; if no limit was specified, the value should be a number greater than or equal to the largest packet length in the file.

LinkType (16 bits): a 16-bit unsigned value that defines the link layer type of packets in the file. This field is defined in the Section 8.2 IANA registry.

Frame Cyclic Sequence (FCS) present (4 bits): if the "f" bit is set, then the 3 FCS bits provide the number of 16-bit (2 byte) words of FCS that are appended to each packet.

valid values are between 0 and 7, with ethernet typically having a length of 4 bytes, or a value of 2.

The bits marked as zero MUST be set to zero by pcap writers, and MUST be ignored by pcap readers.

5. Packet Record

A Packet Record is the standard container for storing the packets coming from the network.

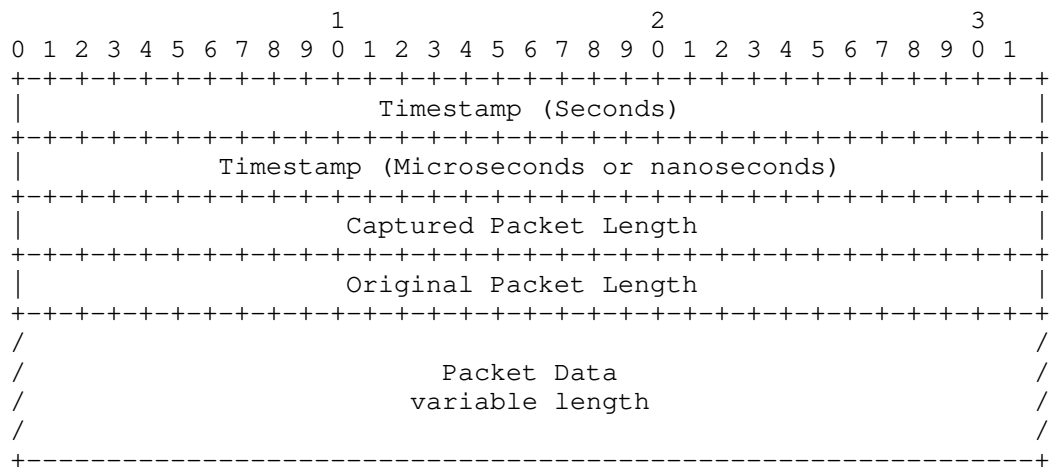

Figure 2: Packet Record

The Packet Header length is 16 octets.

The meaning of the fields in the Packet Record is:

Timestamp (Seconds) and Timestamp (Microseconds or nanoseconds): seconds and fraction of a seconds values of a timestamp.

The seconds value is a 32-bit unsigned integer that represents the number of seconds that have elapsed since 1970-01-01 00:00:00 UTC, and the microseconds or nanoseconds value represents the number of microseconds or nanoseconds that have elapsed since that seconds.

Whether the value represents microseconds or nanoseconds is specified by the magic number in the File Header.

Captured Packet Length (32 bits): an unsigned value that indicates the number of octets captured from the packet (i.e. the length of the Packet Data field). It will be the minimum value among the Original Packet Length and the snapshot length for the interface (SnapLen, defined in Figure 1).

Original Packet Length (32 bits): an unsigned value that indicates the actual length of the packet when it was transmitted on the network. It can be different from the Captured Packet Length if the packet has been truncated by the capture process.

Packet Data: the data coming from the network, including link-layer

headers. The actual length of this field is Captured Packet Length. The format of the link-layer headers depends on the LinkType field specified in the file header (see Figure 1) and it is specified in the entry for that format in [LINKTYPES].

6. Recommended File Name Extension: .pcap

The recommended file name extension for the "PCAP Capture File Format" specified in this document is ".pcap".

On Windows and macOS, files are distinguished by an extension to their filename. Such an extension is technically not actually required, as applications should be able to automatically detect the pcap file format through the "magic bytes" at the beginning of the file, as some other UN*X desktop environments do. However, using name extensions makes it easier to work with files (e.g. visually distinguish file formats) so it is recommended - though not required - to use .pcap as the name extension for files following this specification.

Please note: To avoid confusion (such as the current usage of .cap for a plethora of different capture file formats) file name extensions other than .pcap should be avoided.

There is new work to create the PCAP Next Generation capture File Format (see [I-D.tuexen-opsawg-pcapng]). The new file format is not compatible with this specification, but many programs read both transparently. Files of that type will usually start with a Section Header Block, with a magic number of 0x0A0D0D0A.

7. Security Considerations

TBD.

8. IANA Considerations

This document requires the following IANA actions:

8.1. Media-Type Registry

This section registers the the 'application/pcap' in the "Media Types" registry. These media types are used to indicate that the content is packet capture as described in this document.

8.1.1. application/pcap

Type name: application
Subtype name: pcap
Required parameters: none
Optional parameters: none
Encoding considerations: PCAP files contain network packets
Security considerations: See Security Considerations, Section
Interoperability considerations: The format is designed to be broadly intero
perable.
Published specification: THIS RFC.
Applications that use this media type: tcpdump, wireshark, others.
Additional information:
Magic number(s): 0xA1B2C3D4, and 0xA1B23C4D in both endian orders
File extension(s): .pcap
Macintosh file type code(s): none
Person & email address to contact for further information: The Tcpdump Group,
www.tcpdump.org
Intended usage: LIMITED
Restrictions on usage: NONE
Author: Guy Harris and Michael Richardson
Change controller: The Tcpdump Group
Provisional registration? (standards tree only): NO

8.2. LinkType Registry

IANA is requested to create a new Registry entitled: "The PCAP Registry", and within that Registry to create a table called: "PCAP LinkType List".

The LinkType Registry is a table of 16-bit numbers. The Registry has three sections with different [RFC8126] rules:

* values from 0 to 32767 are marked as Specification Required.

* values from 32768 to 65000 are marked as First-Come First-Served.

* values from 65000 to 65535 are marked as Private Use.

The Registry has four columns: the symbolic name (LINKTYPE_something), the integer value, a very short description, and the document/requestor reference.

The Registry shall be populated as follows in the table below. In each case here, the reference should be http://www.tcpdump.org/linktypes.html, which is not repeated.

The initial value of table is base upon the Link type list maintained by libpcap, and published on the tcpdump.org web site as http://www.tcpdump.org/linktypes.html.

There is often an associated DLT value which are often identical in value, but not universally so. DLT values are associated with specific operation system captures, and are operating system specific.

LINKTYPE name	LINKTYPE value	description
LINKTYPE_NULL	0	BSD loopback encapsulation
LINKTYPE_ETHERNET	1	IEEE 802.3 Ethernet
LINKTYPE_EXP_ETHERNET	2	Xerox experimental 3Mb Ethernet
LINKTYPE_AX25	3	AX.25 packet
LINKTYPE_PRONET	4	Reserved for PRONET
LINKTYPE_CHAOS	5	Reserved for MIT CHAOSNET
LINKTYPE_IEEE802_5	6	IEEE 802.5 Token Ring
LINKTYPE_ARCNET_BSD	7	ARCNET Data Packets with BSD encapsulation
LINKTYPE_SLIP	8	SLIP, w/ LINKTYPE_SLIP header.
LINKTYPE_PPP	9	PPP, as per RFC 1661/RFC 1662
LINKTYPE_FDDI	10	FDDI: per ANSI INCITS 239-1994.
not to be used	11-49	Do not use these values
LINKTYPE_PPP_HDLC	50	PPP in HDLC-like framing, as per RFC 1662

LINKTYPE_PPP_ETHER	51	PPPoE; per RFC 2516
not to be used	52-98	Do not use these values
LINKTYPE_SYMANTEC_FIREWALL	99	Reserved for Symantec Enterprise Firewall
LINKTYPE_ATM_RFC1483	100	RFC 1483 LLC/SNAP-encapsulated ATM
LINKTYPE_RAW	101	Raw IP; begins with an IPv4 or IPv6 header
LINKTYPE_SLIP_BSDOS	102	Reserved for BSD/OS SLIP BPF header
LINKTYPE_PPP_BSDOS	103	Reserved for BSD/OS PPP BPF header
LINKTYPE_C_HDLC	104	Cisco PPP with HDLC framing, as per section 4.3.1 of RFC 1547
LINKTYPE_IEEE802_11	105	IEEE 802.11 wireless LAN.
LINKTYPE_ATM_CLIP	106	ATM Classical IP, with no header preceding IP
LINKTYPE_FRELAY	107	Frame Relay LAPF frames
LINKTYPE_LOOP	108	OpenBSD loopback encapsulation
LINKTYPE_ENC	109	Reserved for OpenBSD IPSEC encapsulation
LINKTYPE_LANE8023	110	Reserved for ATM LANE + 802.3
LINKTYPE_HIPPI	111	Reserved for NetBSD HIPPI

LINKTYPE_HDLC	112	Reserved for NetBSD HDLC framing
LINKTYPE_LINUX_SLL	113	Linux "cooked" capture encapsulation
LINKTYPE_LTALK	114	Apple LocalTalk
LINKTYPE_ECONET	115	Reserved for Acorn Econet
LINKTYPE_IPFILTER	116	Reserved for OpenBSD ipfilter
LINKTYPE_PFLOG	117	OpenBSD pflog; "struct pfloghdr" structure
LINKTYPE_CISCO_IOS	118	Reserved for Cisco-internal use
LINKTYPE_IEEE802_11_PRISM	119	Prism monitor mode
LINKTYPE_IEEE802_11_AIRONET	120	Reserved for 802.11 + FreeFreeBSD Aironet radio metadata
LINKTYPE_HHDLC	121	Reserved for Siemens HiPath HDLC
LINKTYPE_IP_OVER_FC	122	RFC 2625 IP-over-Fibre Channel
LINKTYPE_SUNATM	123	ATM traffic, / per SunATM devices
LINKTYPE_RIO	124	Reserved for RapidIO
LINKTYPE_PCI_EXP	125	Reserved for PCI Express
LINKTYPE_AURORA	126	Reserved for Xilinx Aurora link layer
LINKTYPE_IEEE802_11_RADIOTAP	127	Radiotap

		header[Radiotap], followed by an 802.11 header
LINKTYPE_TZSP	128	Reserved for Tazmen Sniffer Protocol
LINKTYPE_ARCNET_LINUX	129	ARCNET Data Packets, per RFC 1051 frames w/variations
LINKTYPE_JUNIPER_MLPPP	130	Reserved for Juniper Networks
LINKTYPE_JUNIPER_MLFR	131	Reserved for Juniper Networks
LINKTYPE_JUNIPER_ES	132	Reserved for Juniper Networks
LINKTYPE_JUNIPER_GGSN	133	Reserved for Juniper Networks
LINKTYPE_JUNIPER_MFR	134	Reserved for Juniper Networks
LINKTYPE_JUNIPER_ATM2	135	Reserved for Juniper Networks
LINKTYPE_JUNIPER_SERVICES	136	Reserved for Juniper Networks
LINKTYPE_JUNIPER_ATM1	137	Reserved for Juniper Networks
LINKTYPE_APPLE_IP_OVER_IEEE1394	138	Apple IP-over-IEEE 1394 cooked header
LINKTYPE_MTP2_WITH_PHDR	139	Signaling System 7 (SS7) Message Transfer Part Level ITU-T Q.703
LINKTYPE_MTP2	140	SS7 Level 2, Q.703
LINKTYPE_MTP3	141	SS7 Level 3, Q.704
LINKTYPE_SCCP	142	SS7 Control Part,

		ITU-T Q.711/Q.712/Q.713/ Q.714
LINKTYPE_DOCSIS	143	DOCSIS MAC frames, DOCSIS 3.1
LINKTYPE_LINUX_IRDA	144	Linux-IrDA packets w/ LINKTYPE_LINUX_IRDA header
LINKTYPE_IBM_SP	145	Reserved for IBM SP switch
LINKTYPE_IBM_SN	146	Reserved for IBM Next Federation switch
LINKTYPE_RESERVED_01	147	For private use
LINKTYPE_RESERVED_02	148	For private use
LINKTYPE_RESERVED_03	149	For private use
LINKTYPE_RESERVED_04	150	For private use
LINKTYPE_RESERVED_05	151	For private use
LINKTYPE_RESERVED_06	152	For private use
LINKTYPE_RESERVED_07	153	For private use
LINKTYPE_RESERVED_08	154	For private use
LINKTYPE_RESERVED_09	155	For private use
LINKTYPE_RESERVED_10	156	For private use
LINKTYPE_RESERVED_11	157	For private use
LINKTYPE_RESERVED_12	158	For private use
LINKTYPE_RESERVED_13	159	For private use
LINKTYPE_RESERVED_14	160	For private use
LINKTYPE_RESERVED_15	161	For private use

LINKTYPE_RESERVED_16	162	For private use
LINKTYPE_IEEE802_11_AVS	163	AVS header[AVS], followed by an 802.11 header
LINKTYPE_JUNIPER_MONITOR	164	Reserved for Juniper Networks
LINKTYPE_BACNET_MS_TP	165	BACnet MS/TP frames, per 9.3 MS/TP Frame Format ANSI 135
LINKTYPE_PPP_PPPD	166	PPP in HDLC-like encapsulation, like LINKTYPE_PPP_HDLC, different stuffing
LINKTYPE_JUNIPER_PPPOE	167	Reserved for Juniper Networks
LINKTYPE_JUNIPER_PPPOE_ATM	168	Reserved for Juniper Networks
LINKTYPE_GPRS_LLC	169	General Packet Radio Service Logical Link Control, as per 3GPP TS 04.64
LINKTYPE_GPF_T	170	Transparent-mapped generic framing procedure, as specified by ITU-T Recommendation G.7041/Y.1303
LINKTYPE_GPF_F	171	Frame-mapped generic framing procedure, as specified by ITU-T Recommendation G.7041/Y.1303
LINKTYPE_GCOM_T1E1	172	Reserved for Gcom T1/E1 line monitoring equipment
LINKTYPE_GCOM_SERIAL	173	Reserved for Gcom

		T1/E1 line monitoring equipment
LINKTYPE_JUNIPER_PIC_PEER	174	Reserved for Juniper Networks
LINKTYPE_ERF_ETH	175	Endace ERF header followed by 802.3 Ethernet
LINKTYPE_ERF_POS	176	Endace ERF header followed by Packet-over-SONET
LINKTYPE_LINUX_LAPD	177	Link Access Procedures on the D Channel (LAPD) frames, as specified by ITU-T Recommendation Q.920 and ITU-T Recommendation Q.921 , captured via vISDN, with a LINKTYPE_LINUX_LAPD header , followed by the Q.921 frame, starting with the address field.
LINKTYPE_JUNIPER_ETHER	178	Reserved for Juniper Networks
LINKTYPE_JUNIPER_PPP	179	Reserved for Juniper Networks
LINKTYPE_JUNIPER_FRELAY	180	Reserved for Juniper Networks
LINKTYPE_JUNIPER_CHDLC	181	Reserved for Juniper Networks
LINKTYPE_MFR	182	FRF.16.1 Multi-Link Frame Relay frames, beginning with an FRF.12 Interface fragmentation format fragmentation

		header.
LINKTYPE_JUNIPER_VP	182	Reserved for Juniper Networks
LINKTYPE_A653_ICM	185	Reserved for Arinc 653 Interpartition Communication messages
LINKTYPE_USB_FREEBSD	186	USB packets, beginning with a FreeBSD USB header
LINKTYPE_BLUETOOTH_HCI_H4	187	Bluetooth HCI UART transport layer; the frame contains an HCI packet indicator byte, as specified by the UART Transport Layer portion of the most recent Bluetooth Core specification , followed by an HCI packet of the specified packet type, as specified by the Host Controller Interface Functional Specification portion of the most recent Bluetooth Core Specification.
LINKTYPE_IEEE802_16_MAC_CPS	188	Reserved for IEEE 802.16 MAC Common Part Sublayer
LINKTYPE_USB_LINUX	189	USB packets, beginning with a Linux USB header, as specified by the struct usbmon_packet in the Documentation/usb/ usbmon.txt file in

		the Linux source tree. Only the first 48 bytes of that header are present. All fields in the header are in host byte order. When performing a live capture, the host byte order is the byte order of the machine on which the packets are captured. When reading a pcap file, the byte order is the byte order for the file, as specified by the file's magic number; when reading a pcapng file, the byte order is the byte order for the section of the pcapng file, as specified by the Section Header Block.
LINKTYPE_CAN20B	190	Reserved for Controller Area Network (CAN) v. 2.0B packets
LINKTYPE_IEEE802_15_4_LINUX	191	IEEE 802.15.4, with address fields padded, as is done by Linux drivers
LINKTYPE_PPI	192	Per-Packet Information information, as specified by the Per-Packet Information Header Specification , followed by a packet

		with the LINKTYPE_ value specified by the pph_dlt field of that header.
LINKTYPE_IEEE802_16_MAC_CPS_RADIO	193	Reserved for 802.16 MAC Common Part Sublayer plus radio header
LINKTYPE_JUNIPER_ISM	194	Reserved for Juniper Networks
LINKTYPE_IEEE802_15_4_WITHFCS	195	IEEE 802.15.4 Low-Rate Wireless Networks, with each packet having the FCS at the end of the frame.
LINKTYPE_SITA	196	Various link-layer types, with a pseudo-header , for SITA
LINKTYPE_ERF	197	Various link-layer types, with a pseudo-header, for Endace DAG cards; encapsulates Endace ERF records.
LINKTYPE_RAIF1	198	Reserved for Ethernet packets captured from a u10 Networks board
LINKTYPE_IPMB_KONTRON	199	Reserved for IPMB packet for IPMI, with a 2-byte header
LINKTYPE_JUNIPER_ST	200	Reserved for Juniper Networks
LINKTYPE_BLUETOOTH_HCI_H4_WITH_PHDR	201	Bluetooth HCI UART transport layer; the frame contains a 4-byte direction

		field, in network byte order (big-endian), the low-order bit of which is set if the frame was sent from the host to the controller and clear if the frame was received by the host from the controller, followed by an HCI packet indicator byte, as specified by the UART Transport Layer portion of the most recent Bluetooth Core specification , followed by an HCI packet of the specified packet type, as specified by the Host Controller Interface Functional Specification portion of the most recent Bluetooth Core Specification.
LINKTYPE_AX25_KISS	202	AX.25 packet, with a 1-byte KISS header containing a type indicator.
LINKTYPE_LAPD	203	Link Access Procedures on the D Channel (LAPD) frames, as specified by ITU-T Recommendation Q.920 and ITU-T Recommendation Q.921 , starting with the address field, with no pseudo-header.

LINKTYPE_PPP_WITH_DIR	204	PPP, as per RFC 1661 and RFC 1662 , preceded with a one-byte pseudo-header with a zero value meaning received by this host and a non-zero value meaning sent by this host; if the first 2 bytes are 0xff and 0x03, it's PPP in HDLC-like framing, with the PPP header following those two bytes, otherwise it's PPP without framing, and the packet begins with the PPP header. The data in the frame is not octet-stuffed or bit-stuffed.
LINKTYPE_C_HDLC_WITH_DIR	205	Cisco PPP with HDLC framing, as per section 4.3.1 of RFC 1547 , preceded with a one-byte pseudo-header with a zero value meaning received by this host and a non-zero value meaning sent by this host.
LINKTYPE_FRELAY_WITH_DIR	206	Frame Relay LAPF frames, beginning with a one-byte pseudo-header with a zero value meaning received by this host (DCE->DTE) and a non-zero value meaning sent by this host (DTE->DCE), followed by an ITU-T Recommendation Q.922

		LAPF header starting with the address field, and without an FCS at the end of the frame.
LINKTYPE_LAPB_WITH_DIR	207	Link Access Procedure, Balanced (LAPB), as specified by ITU-T Recommendation X.25 , preceded with a one-byte pseudo-header with a zero value meaning received by this host (DCE->DTE) and a non-zero value meaning sent by this host (DTE->DCE).
Reserved	208	Reserved for an unspecified link-layer type
LINKTYPE_IPMB_LINUX	209	IPMB over an I2C circuit, with a Linux-specific pseudo-header
LINKTYPE_FLEXRAY	210	Reserved for FlexRay automotive bus
LINKTYPE_MOST	211	Reserved for Media Oriented Systems Transport (MOST) bus
LINKTYPE_LIN	212	Reserved for Local Interconnect Network (LIN) bus for vehicle networks
LINKTYPE_X2E_SERIAL	213	Reserved for X2E serial line captures
LINKTYPE_X2E_XORAYA	214	Reserved for X2E Xoraya data loggers

LINKTYPE_IEEE802_15_4_NONASK_PHY	215	IEEE 802.15.4 Low-Rate Wireless Networks, with each packet having the FCS at the end of the frame, and with the PHY-level data for the O-QPSK, BPSK, GFSK, MSK, and RCC DSS BPSK PHYs (4 octets of 0 as preamble, one octet of SFD, one octet of frame length + reserved bit) preceding the MAC-layer data (starting with the frame control field).
LINKTYPE_LINUX_EVDEV	216	Reserved for Linux evdev messages
LINKTYPE_GSMTAP_UM	217	Reserved for GSM Um interface, with gsmtap header
LINKTYPE_GSMTAP_ABIS	218	Reserved for GSM Abis interface, with gsmtap header
LINKTYPE_MPLS	219	MPLS packets with MPLS label as the header
LINKTYPE_USB_LINUX_MMAPPED	220	USB packets, beginning with a Linux USB header, as specified by the struct usbmon_packet in the Documentation/usb/usbmon.txt file in the Linux source tree. All 64 bytes of the header are present. All fields in the header are in

		host byte order. When performing a live capture, the host byte order is the byte order of the machine on which the packets are captured. When reading a pcap file, the byte order is the byte order for the file, as specified by the file's magic number; when reading a pcapng file, the byte order is the byte order for the section of the pcapng file, as specified by the Section Header Block. For isochronous transfers, the ndesc field specifies the number of isochronous descriptors that follow.
LINKTYPE_DECT	221	Reserved for DECT packets, with a pseudo-header
LINKTYPE_AOS	222	Reserved for OS Space Data Link Protocol
LINKTYPE_WIHART	223	Reserved for Wireless HART (Highway Addressable Remote Transducer)
LINKTYPE_FC_2	224	Fibre Channel FC-2 frames, beginning with a Frame_Header.

LINKTYPE_FC_2_WITH_FRAME_DELIMS	225	Fibre Channel FC-2 frames, beginning an encoding of the SOF, followed by a Frame_Header, and ending with an encoding of the SOF. The encodings represent the frame delimiters as 4-byte sequences representing the corresponding ordered sets, with K28.5 represented as 0xBC, and the D symbols as the corresponding byte values; for example, SOFi2, which is K28.5 - D21.5 - D1.2 - D21.2, is represented as 0xBC 0xB5 0x55 0x55.
LINKTYPE_IPNET	226	Solaris ipnet pseudo-header , followed by an IPv4 or IPv6 datagram.
LINKTYPE_CAN_SOCKETCAN	227	CAN (Controller Area Network) frames, with a pseudo-header followed by the frame payload.
LINKTYPE_IPV4	228	Raw IPv4; the packet begins with an IPv4 header.
LINKTYPE_IPV6	229	Raw IPv6; the packet begins with an IPv6 header.
LINKTYPE_IEEE802_15_4_NOFCS	230	IEEE 802.15.4 Low-Rate Wireless Network, without the FCS at the end of

		the frame.
LINKTYPE_DBUS	231	Raw D-Bus messages , starting with the endianness flag, followed by the message type, etc., but without the authentication handshake before the message sequence.
LINKTYPE_JUNIPER_VS	232	Reserved for Juniper Networks
LINKTYPE_JUNIPER_SRX_E2E	233	Reserved for Juniper Networks
LINKTYPE_JUNIPER_FIBRECHANNEL	234	Reserved for Juniper Networks
LINKTYPE_DVB_CI	235	DVB-CI (DVB Common Interface for communication between a PC Card module and a DVB receiver), with the message format specified by the PCAP format for DVB-CI specification
LINKTYPE_MUX27010	236	Variant of 3GPP TS 27.010 multiplexing protocol (similar to, but not the same as, 27.010).
LINKTYPE_STANAG_5066_D_PDU	237	D_PDUs as described by NATO standard STANAG 5066, starting with the synchronization sequence, and including both header and data CRCs. The current version of STANAG

		5066 is backwards-compatible with the 1.0.2 version , although newer versions are classified.
LINKTYPE_JUNIPER_ATM_CEMIC	238	Reserved for Juniper Networks
LINKTYPE_NFLOG	239	Linux netlink NETLINK NFLOG socket log messages.
LINKTYPE_NETANALYZER	240	Pseudo-header for Hilscher Gesellschaft fuer Systemautomation mbH netANALYZER devices , followed by an Ethernet frame, beginning with the MAC header and ending with the FCS.
LINKTYPE_NETANALYZER_TRANSPARENT	241	Pseudo-header for Hilscher Gesellschaft fuer Systemautomation mbH netANALYZER devices , followed by an Ethernet frame, beginning with the preamble, SFD, and MAC header, and ending with the FCS.
LINKTYPE_IPOIB	242	IP-over-InfiniBand, as specified by RFC 4391 section 6
LINKTYPE_MPEG_2_TS	243	MPEG-2 Transport Stream transport packets, as specified by ISO 13818-1/ ITU-T Recommendation H.222.0 (see table

		2-2 of section 2.4.3.2 Transport Stream packet layer).
LINKTYPE_NG40	244	Pseudo-header for ng4T GmbH's UMTS Iub/Iur-over-ATM and Iub/Iur-over-IP format as used by their ng40 protocol tester , followed by frames for the Frame Protocol as specified by 3GPP TS 25.427 for dedicated channels and 3GPP TS 25.435 for common/ shared channels in the case of ATM AAL2 or UDP traffic, by SSCOP packets as specified by ITU-T Recommendation Q.2110 for ATM AAL5 traffic, and by NBAP packets for SCTP traffic.
LINKTYPE_NFC_LLCP	245	Pseudo-header for NFC LLCP packet captures , followed by frame data for the LLCP Protocol as specified by NFCForum-TS-LLCP_1.1
LINKTYPE_PFSYNC	246	Reserved for pfsync output
LINKTYPE_INFINIBAND	247	Raw InfiniBand frames, starting with the Local Routing Header, as specified in Chapter 5 Data packet format of InfiniBand[TM] Architectural

		Specification Release 1.2.1 Volume 1 - General Specifications
LINKTYPE_SCTP	248	SCTP packets, as defined by RFC 4960 , with no lower-level protocols such as IPv4 or IPv6.
LINKTYPE_USBPCAP	249	USB packets, beginning with a USBPcap header
LINKTYPE_RTAC_SERIAL	250	Serial-line packet header for the Schweitzer Engineering Laboratories RTAC product , followed by a payload for one of a number of industrial control protocols.
LINKTYPE_BLUETOOTH_LE_LL	251	Bluetooth Low Energy air interface Link Layer packets, in the format described in section 2.1 PACKET FORMAT of volume 6 of the Bluetooth Specification Version 4.0 (see PDF page 2200), but without the Preamble.
LINKTYPE_WIRESHARK_UPPER_PDU	252	Reserved for Wireshark
LINKTYPE_NETLINK	253	Linux Netlink capture encapsulation
LINKTYPE_BLUETOOTH_LINUX_MONITOR	254	Bluetooth Linux

		Monitor encapsulation of traffic for the BlueZ stack
LINKTYPE_BLUETOOTH_BREDR_BB	255	Bluetooth Basic Rate and Enhanced Data Rate baseband packets
LINKTYPE_BLUETOOTH_LE_LL_WITH_PHDR	256	Bluetooth Low Energy link-layer packets
LINKTYPE_PROFIBUS_DL	257	PROFIBUS data link layer packets, as specified by IEC standard 61158-4-3, beginning with the start delimiter, ending with the end delimiter, and including all octets between them.
LINKTYPE_PKTAP	258	Apple PKTAP capture encapsulation
LINKTYPE_EPON	259	Ethernet-over-passive-optical-network packets, starting with the last 6 octets of the modified preamble as specified by 65.1.3.2 Transmit in Clause 65 of Section 5 of IEEE 802.3 , followed immediately by an Ethernet frame.
LINKTYPE_IPMI_HPM_2	260	IPMI trace packets, as specified by Table 3-20 Trace Data Block Format in the PICMG HPM.2 specification The time stamps for

		packets in this format must match the time stamps in the Trace Data Blocks.
LINKTYPE_ZWAVE_R1_R2	261	Z-Wave RF profile R1 and R2 packets , as specified by ITU-T Recommendation G.9959 , with some MAC layer fields moved.
LINKTYPE_ZWAVE_R3	262	Z-Wave RF profile R3 packets , as specified by ITU-T Recommendation G.9959 , with some MAC layer fields moved.
LINKTYPE_WATTSTOPPER_DLM	263	Formats for WattStopper Digital Lighting Management (DLM) and Legrand Nitoo Open protocol common packet structure captures.
LINKTYPE_ISO_14443	264	Messages between ISO 14443 contactless smartcards (Proximity Integrated Circuit Card, PICC) and card readers (Proximity Coupling Device, PCD), with the message format specified by the PCAP format for ISO14443 specification
LINKTYPE_RDS	265	Radio data system (RDS) groups, as per IEC 62106,

		encapsulated in this form
LINKTYPE_USB_DARWIN	266	USB packets, beginning with a Darwin (macOS, etc.) USB header
LINKTYPE_OPENFLOW	267	Reserved for OpenBSD DLT_OPENFLOW
LINKTYPE_SDLC	268	SDLC packets, as specified by Chapter 1, DLC Links, section Synchronous Data Link Control (SDLC) of Systems Network Architecture Formats, GA27-3136-20 , without the flag fields, zero-bit insertion, or Frame Check Sequence field, containing SNA path information units (PIUs) as the payload.
LINKTYPE_TI_LLN_SNIFFER	269	Reserved for Texas Instruments protocol sniffer
LINKTYPE_LORATAP	270	LoRaTap pseudo-header , followed by the payload, which is typically the PHYPayload from the LoRaWan specification
LINKTYPE_VSOCK	271	Protocol for communication between host and guest machines in VMware and KVM hypervisors.

LINKTYPE_NORDIC_BLE	272	Messages to and from a Nordic Semiconductor nRF Sniffer for Bluetooth LE packets, beginning with a pseudo-header
LINKTYPE_DOCSIS31_XRA31	273	DOCSIS packets and bursts, preceded by a pseudo-header giving metadata about the packet
LINKTYPE_ETHERNET_MPACKET	274	mPackets, as specified by IEEE 802.3br Figure 99-4, starting with the preamble and always ending with a CRC field.
LINKTYPE_DISPLAYPORT_AUX	275	DisplayPort AUX channel monitoring data as specified by VESA DisplayPort(DP) Standard preceded by a pseudo-header
LINKTYPE_LINUX_SLL2	276	Linux cooked capture encapsulation v2
LINKTYPE_SERCOS_MONITOR	277	Reserved for Sercos Monitor
LINKTYPE_OPENVIZSLA	278	Openvizsla FPGA-based USB sniffer
LINKTYPE_EBHSCR	279	Elektrobit High Speed Capture and Replay (EBHSCR) format
LINKTYPE_VPP_DISPATCH	280	Records in traces from the http://fd.io VPP graph dispatch tracer, in the the

		graph dispatcher trace format
LINKTYPE_DSA_TAG_BRCM	281	Ethernet frames, with a switch tag inserted between the source address field and the type/length field in the Ethernet header.
LINKTYPE_DSA_TAG_BRCM_PREPEND	282	Ethernet frames, with a switch tag inserted before the destination address in the Ethernet header.
LINKTYPE_IEEE802_15_4_TAP	283	IEEE 802.15.4 Low-Rate Wireless Networks, with a pseudo-header containing TLVs with metadata preceding the 802.15.4 header.
LINKTYPE_DSA_TAG_DSA	284	Ethernet frames, with a switch tag inserted between the source address field and the type/length field in the Ethernet header.
LINKTYPE_DSA_TAG_EDSA	285	Ethernet frames, with a programmable Ethernet type switch tag inserted between the source address field and the type/length field in the Ethernet header.
LINKTYPE_ELEE	286	Payload of lawful intercept packets using the ELEE protocol The packet begins with the ELEE

		header; it does not include any transport-layer or lower-layer headers for protcols used to transport ELEE packets.
LINKTYPE_Z_WAVE_SERIAL	287	Serial frames transmitted between a host and a Z-Wave chip over an RS-232 or USB serial connection, as described in section 5 of the Z-Wave Serial API Host Application Programming Guide
LINKTYPE_USB_2_0	288	USB 2.0, 1.1, or 1.0 packet, beginning with a PID, as described by Chapter 8 Protocol Layer of the the Universal Serial Bus Specification Revision 2.0
LINKTYPE_ATSC_ALP	289	ATSC Link-Layer Protocol frames, as described in section 5 of the A/330 Link-Layer Protocol specification, found at the ATSC 3.0 standards page , beginning with a Base Header

Table 1

9. Contributors

[Insert pcap developers etc. here].

10. Acknowledgments

The authors wish to thank [insert list here] and many others for their invaluable comments.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

[I-D.tuexen-opsawg-pcapng] Tuexen, M., Risso, F., Bongertz, J., Combs, G., Harris, G., and M. Richardson, "PCAP Next Generation (pcapng) Capture File Format", Work in Progress, Internet-Draft, draft-tuexen-opsawg-pcapng-02, 28 September 2020, <https://www.ietf.org/archive/id/draft-tuexen-opsawg-pcapng-02.txt>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[Radiotap] radiotap.org, "Radiotap Web site", n.d., <http://www.radiotap.org/>.

[AVS] Peachy, S., "Archived AVS specification", n.d., <http://web.archive.org/web/20040803232023/http://www.shaftnet.org/~pizza/software/capturefrm.txt>.

Authors' Addresses

Guy Harris (editor)

Email: gharris@sonic.net

Michael C. Richardson
Sandelman Software Works Inc

Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/