Network Working Group
Internet-Draft
Intended status: Historic
Expires: 15 November 2026

G. Harris, Ed.

M. Richardson
Sandelman

14 May 2026

# PCAP Capture File Format
# draft-ietf-opsawg-pcap-08

## 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.

## About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-opsawg-pcap/.

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

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

## Status of This Memo

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 historical format used by tcpdump, and other programs using libpcap, to read and write network traces. This document describes version 2 of the pcap format.

This document is published as historical, as there has existed for some time, an updated format called "pcapng", that replaces this file format. See [I-D.ietf-opsawg-pcapng]. No new extensions for this format are expected, although new LINKLAYER types that are registered using [I-D.ietf-opsawg-pcaplinktype] can be included in pcap files.

A major limitation of the pcap v2 format described here is that files consist of a header which is different than the other blocks in the file. This prevents pcap v2 files from being simply concatenated for processing. It is also difficult to break pcap v2 files apart, as a new header always needs to be placed at the beginning of any new file. The pcapng format does not suffer from these problems.

More significantly, pcap v2 files can only contain packets in a single LINKTYPE format, and this often means that packets are often from a single network interface as not all LINKTYPEs include a way to indicate which interface a packet is from.

## 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.

This document uses the term octet in a way consistent with the word "byte"

## 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 the headers of Packet Records will always be written according to the characteristics (little-endian / big- endian) of the machine that is writing the file. This refers to all the fields that are written as numbers and that span over two or more octets.

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

When hosts with a different native endian format read a file, they must swap octets as appropriate. This is less efficient, but less common, and if repeated access to the files are important, then files can be translated and saved.

## 4. File Header

The File Header has the following format, with the octet offset of fields shown to the left of the field:

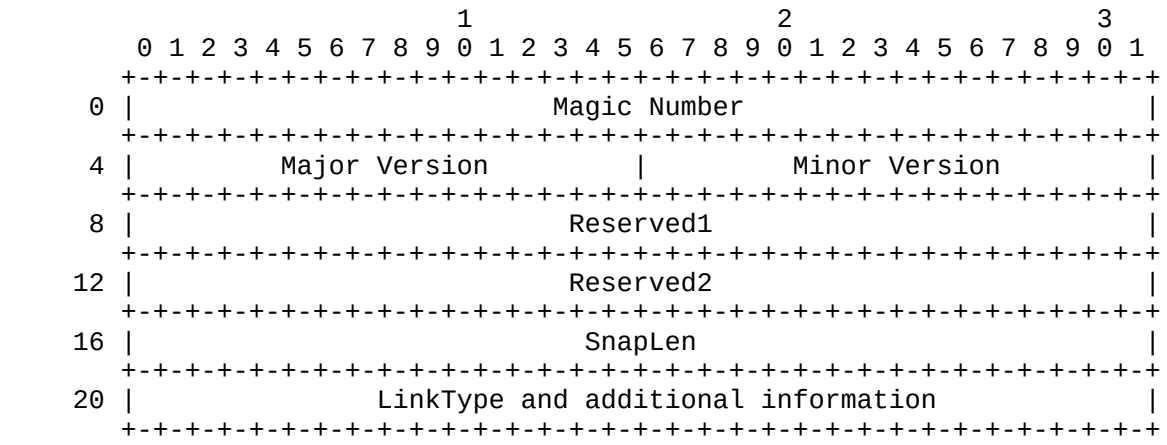


Figure 1: File Header

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, timestamps in Packet Records (see Figure 3) are in seconds and microseconds; if it is 0xA1B23C4D, timestamps in Packet Records are in seconds and nanoseconds.

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

Major Version (16 bits): an unsigned integer, giving the number of the current major version of the format. The value for the current version of the format is 2 (big-endian 0x00 0x02 or little-endian 0x02 0x00). 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. As this document is historical, and no newer formats were publicly released, this value will not change again.

Minor Version (16 bits): an unsigned integer, giving the number of the current minor version of the format. The value for the current version of the format is 4 (big-endian 0x00 0x04 or little-endian 0x04 0x00). 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. As this document is historical, and no newer formats exist, this value will not change again.

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" or "time zone offset". 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 integer that indicates 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 and additional information (32 bits): an unsigned integer that contains the link-layer type of packets in the file and may contain additional information.

The LinkType and additional information field is in the form

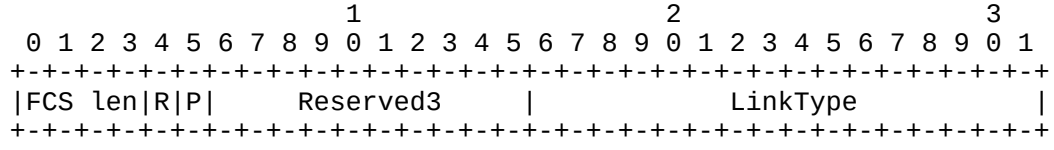

Figure 2: LinkType and additional information

The field is shown as if it were in the octet order of the host reading or writing the file, with bit 0 being the most-significant bit of the field and bit 31 being the least-significant bit of the field.

FCS len (4 bits): an unsigned integer that indicates the number of 16-bit (2-octet) words of FCS that are appended to each packet, if the P bit is set; if the P bit is not set, and the FCS length is not indicated by the link-layer type value, the FCS length is unknown. The valid values of the FCS len field are between 0 and 15; Ethernet, for example, would have an FCS length value of 2, corresponding to a 4-octet FCS.

R (1 bit): not used - MUST be set to zero by pcap writers, and MUST NOT be interpreted by pcap readers; a reader SHOULD treat a non-zero value as an error.

P (1 bit): a bit that, if set, indicates that the Frame Check Sequence (FCS) length value is present and, if not set, indicates that the FCS value is not present.

Reserved3 (10 bits): not used - MUST be set to zero by pcap writers, and MUST NOT be interpreted by pcap readers; a reader SHOULD treat a non-zero value as an error.

LinkType (16 bits): an unsigned integer that indicates the link layer type for packets in the file; it is a value as defined in the PCAP-related LinkType List registry, as defined in [I-D.ietf-opsawg-pcaplinktype].

## 4.1. File Endian Information

The magic number is stored in native endian format, so all the octet sequences below are magic numbers.

* 0xA1,0xB2,0xC3,0xD4: little endian file, with timestamps in seconds/microseconds.

* 0x1A,0x2B,0x3C,0x4D: little endian file, with timestamps in seconds/nanoseconds.

* 0xD4,0xC3,0xB2,0xA1: big endian file, with timestamps in seconds/microseconds.

* 0x4D,0x3C,0x2B,0x1A: big endian file, with timestamps in seconds/nanoseconds.

## 5. Packet Record

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

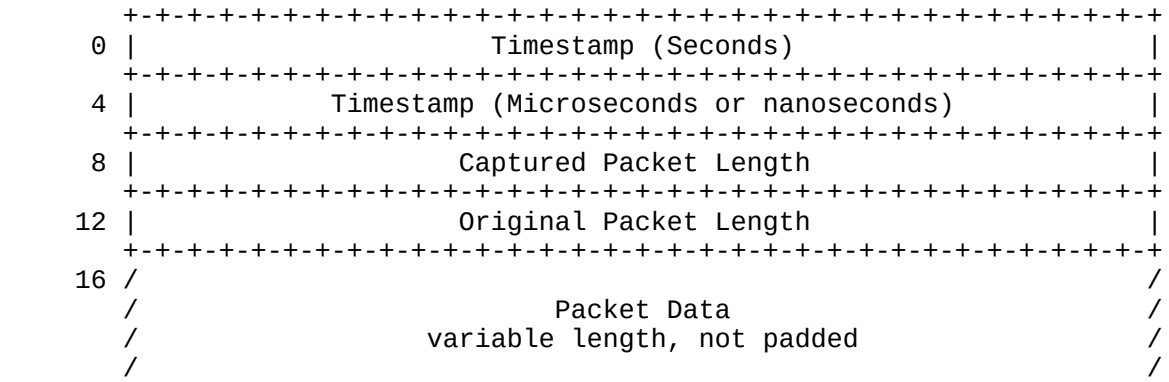


Figure 3: Packet Record

The Packet Record begins with a 16-octet header, followed by data from the packet.

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 is a 32-bit unsigned integer that represents the number of microseconds or nanoseconds that have elapsed since that seconds.

The Magic Number field in the File Header of a file indicates whether the values of the Timestamp (Microseconds or nanoseconds) fields of packets in that file are in units of microseconds or nanoseconds.

Captured Packet Length (32 bits): an unsigned integer 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 integer that indicates

the number of octets of packet data that would have been provided had the packet not been truncated to the snapshot length for the interface or to a length limit imposed by the capture mechanism. If no truncation was done, it will be the same as the Captured Packet Length, but it will be different from the Captured Packet Length if the packet has been truncated by the capture process. It SHOULD NOT be less than the Captured Packet Length.

A pcap file writer MAY write an Original Packet Length that is less than the Captured Packet Length if both the Captured Packet Length and the Original Packet length came from a file in which a packet had an Original Packet Length less than the Captured Packet Length; otherwise, it MUST write an Original Packet Length that is greater than or equal to the Captured Packet Length.

A pcap file reader MAY convert an Original Packet Length that is less than the Captured Packet Length to a value that is greater than or equal to the Captured Packet Length.

Packet Data: the data coming from the network, including link-layer headers. The actual length of this field is the 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 [I-D.ietf-opsawg-pcaplinktype].

Packet Records are not padded to a 4-octet boundary; if the number of octets of packet data is not a multiple of 4, there are no padding octets following it, so Packet Records are not guaranteed to begin on a 4-octet boundary within a file.

## 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 Number field in the File Header, as some desktop environments other than those of Windows and macOS 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 Now Generic capture File Format (see [I-D.ietf-opsawg-pcapng]). The new file format is not compatible with this specification, but many programs read both transparently. Files of that type will start with a Section Header Block, the first four octets of which are 0x0A 0x0D 0x0D 0x0A, which does not match any of the Magic Number values in a pcap File Header, allowing code that reads both file formats to determine the format of a file.

## 7. Security Considerations

A pcap file reader MUST validate the File Header and Packet Record headers. If it analyzes the Packet Data according to the LINKTYPE for the packets, it must also validate all of that data. A reader can receive as input not only valid headers or packets, but any arbitrary random sequence of octets: Headers or packets may be intentionally malformed by a sender, and capture files from outside sources may contain intentionally malformed contents, for malicious reasons.

See also: https://www.iana.org/assignments/media-types/application/vnd.tcpdump.pcap

## 8. IANA Considerations

This document requires one IANA action:

### 8.1. Media-Type Registry

This section registers 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 interoperable.
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

## 9. Acknowledgments

The authors wish to thank Michael Tuexen for document shepherding as well as being the original impetous for starting this work.

Carsten Bormann, Joe Clarke, Mohamed Boucadair, and John Thacker provided review comments and suggested text and diagrams.

The TCPDUMP Group team, led by Denis Ovsienko, and Francois-Xavier Le Bail contributed to this document and helped motivate its forward progress.

## 10. References

### 10.1. Normative References

[I-D.ietf-opsawg-pcaplinktype] Harris, G. and M. Richardson, "Link-Layer Types for PCAP-related Capture File Formats", Work in Progress, Internet-Draft, draft-ietf-opsawg-pcaplinktype-18, 6 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-pcaplinktype-18>.

[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/rfc/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/rfc/rfc8174>.

## 10.2. Informative References

[I-D.ietf-opsawg-pcapng]
T端xen, M., Risso, F., Bongertz, J., Combs, G., Harris, G., Chaudron, E., and M. Richardson, "PCAP Now Generic (pcapng) Capture File Format", Work in Progress, Internet-Draft, draft-ietf-opsawg-pcapng-05, 16 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-pcapng-05>.

## 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/