Workgroup: Network Working Group
Internet-Draft: draft-gharris-opsawg-pcap-00
Published: 22 December 2020
Intended Status: Informational
Expires: 25 June 2021
Authors: G. Harris, Ed. M. Richardson
Sandelman

PCAP Capture File Format

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 June 2021.

Copyright Notice

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:

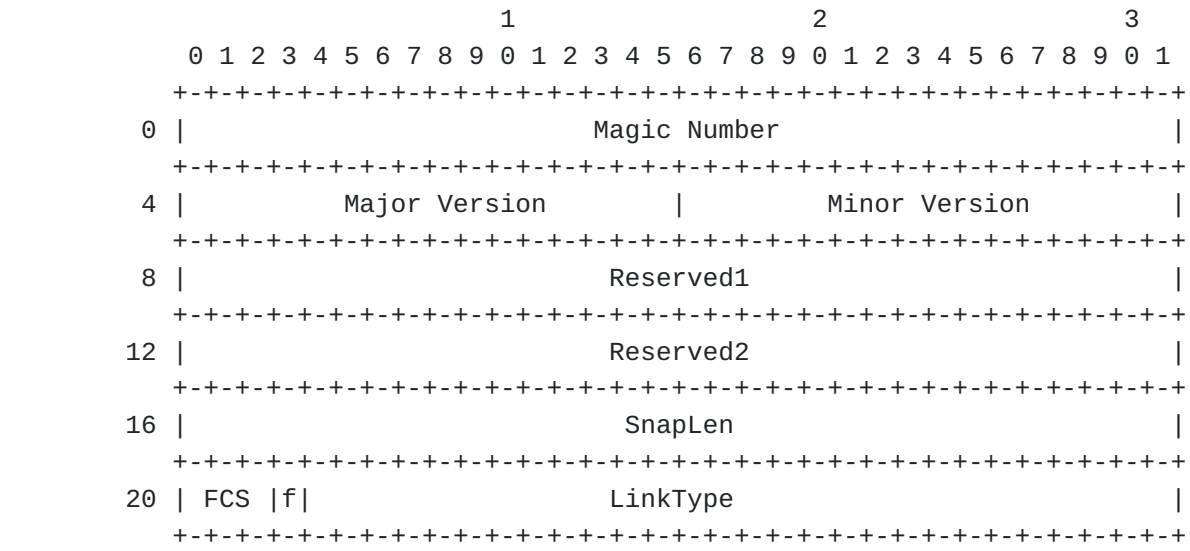

Figure 1: File Header

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 sections 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 (32 bits): an unsigned value that defines, in the lower 28 bits, the link layer type of packets in the file.

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

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

5. Packet Record

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

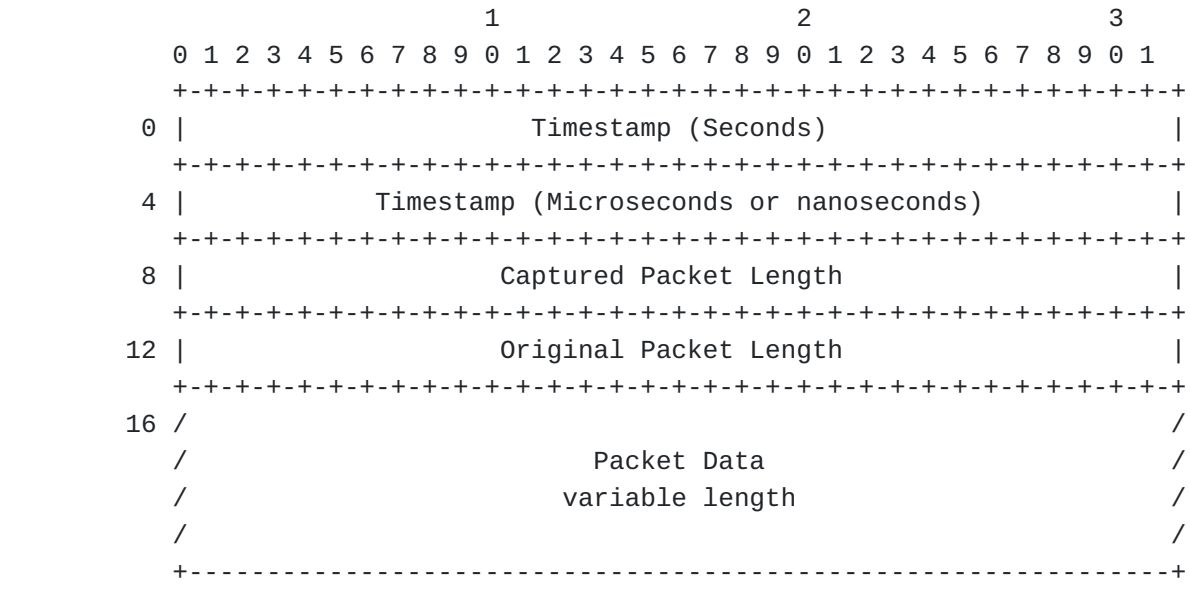

Figure 2: Packet Record

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

TBD.

[Open issue: decide whether the LinkType values should be IANA registries. And if so, what the IANA policy for each should be (see RFC 5226)]

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, <http://www.ietf.org/internet-drafts/draft-tuexen-opsawg-pcapng-02.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/