Pcapng文件

3.通用文件结构

3.1通用块结构

文件是由块组成的，一个追加一个块形成文件。所有块使用相同的形式，如图1

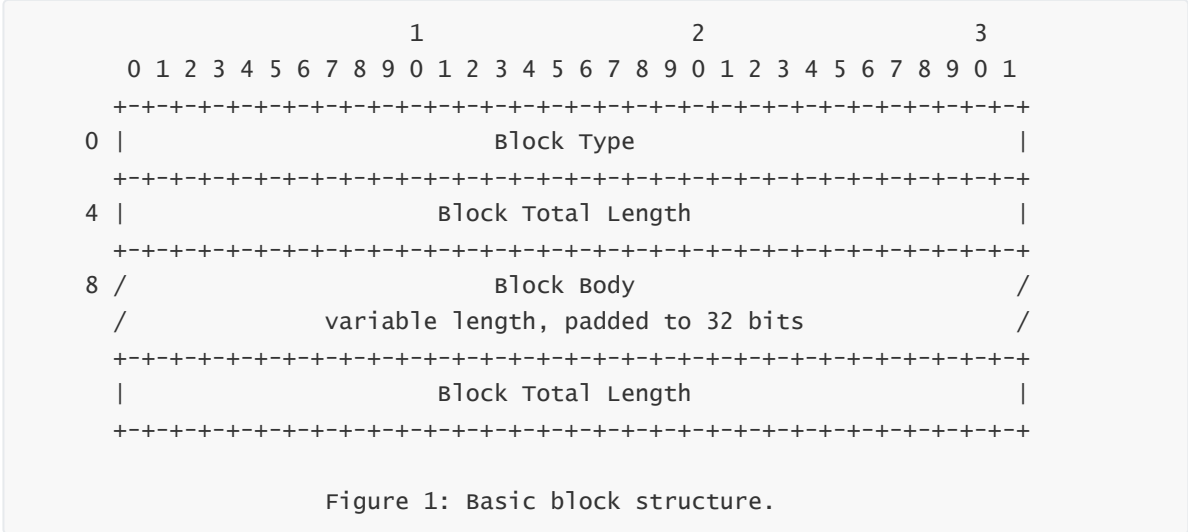


Figure 1: Basic block structure.

- **Block Type**：32位(xx xx)，用于标识一个块的特殊无符号值。最高有效位是1的值保留供本地使用。它们可以用于扩展文件格式，以将私有数据保存到文件中。当前定义的类型可以在11.1节中找到.
- **Block Total Length**：32位，表示这个块总大小的无符号值，单位是字节。例如，一个没有Block Body的块的长度是12字节：4字节Block Type，4字节初始Block Total Length，4字节结尾Block Total Lengh。
- **Block Body**：块的内容
- **Block Total Length**：块总大小的字节数。重复一次这个字段从而方便向后浏览文件。

所有的块都使用这个结构，轻松处理文件，并跳过不需要或未知的块。一些块可以在内部包含另一个块（块嵌套）。一些块是强制性的，比如，如果捕获文件不存在，则它们不存在，其他是可选的。

通用块结构允许需要时定义其他块，解析器不能理解的部分会简单的忽略它们的内容。

3.2 块类型

当前标准块类型的代码在Section11.1中指明，他们分为以下四个类别：

下列强制（MANDATORY）块必须在每个文件中至少出现一次：

- **Section Header Block (4.1节)**：定义了捕获文件的最重要的特征。

下列可选（OPTIONAL）块可以在文件中出现：

- **Interface Description Block（4.2节）**：它定义了用于捕获流量接口的最重要的特征。这个块在某些情况下需要使用，后续将进行描述。

- **Enhanced Packet Block（4.3节）**：它包含一个捕获的数据包，或它的一部分，它代表了原始Packet Block（现在已过时）的演变（附录A）。如果在文件中有此块，则也应该有Interface Description Block，且放在此块之前。
- **Simple Packet Block（4.4节）**：它包含一个捕获的数据包，或它的一部分，只有很少的信息。如果文件中有此块，则Interface Description Block也应该存在，且放在此块之前。
- **Name Resolution Block（4.5节）**：它定义了从数据包捕获中存在的数字地址和对应的规范名称的映射。
- **Interface Statics Block（4.6节）**：它定义了如何存储一些统计数据（如，数据包丢失，等等），对于了解捕获条件很有用。如果它出现在一个文件中，Interface Description Block也应该出现在文件中，放在此块之前。
- **Custom Block（4.9节）**：它以可移植的方式包含特定于供应商的数据。

下列过时的（OBSOLETE）块不应该出现在现在的文件中（但是记录在附录中以供参考）

- **Packet Block（附录A）**：它包含了一个捕获的数据包或它的一部分。它是过时的，被Enhanced Packet Block取代（4.3节）

下列试验块在考虑范围，但作者认为他们在定义前需要更深入的讨论：

- Alternative Packet Blocks
- Compression Block
- Encryption Block
- Fixed Length Block
- Directory Block
- Traffic Statistics and Monitoring Blocks
- Event/Security Blocks

对新的标准化块类型代码的请求请发送到pcap-ng-format邮件列表[1]。

3.3 逻辑块的层次结构

这些块相互参照时建立了逻辑层次结构。

图2 用树形图展示了当前块定义的逻辑层次结构

```
Section Header
|
+- Interface Description
|  +- Simple Packet
|  +- Enhanced Packet
|  +- Interface Statistics
|
+- Name Resolution

          Figure 2: Logical Block Hierarchy of a pcapng File
```

例如：每个捕获的数据包参考一个特定的捕获接口，接口本身参考一个特定的部分。

3.4 物理文件布局

文件必须以**Section Header Block**开始。但是，一个文件中可以出现一个以上的Section Header Block，每一个覆盖其后的数据知道下一个（或文件末尾）为止。 一个节包含被两个Section Header Blocks（或被一个Section Header Block 和文件结尾）限制的数据，包括第一个Section Header Block。

在版本号不同的情况下，一个应用不能读取一个Section，它必须跳过全部内容，直到下一个Section Header Block。 需要注意的是，为了正好跳到下一个部分，所有的块在起始处都必须有Type字段和Length字段。为了正确地向后跳过块，所有的块都必须在块结尾重复一次Length字段。 这些强制要求在未来的块形式中也必须被维护。

图3 展示了一个经典的文件布局，用一个单个的Section Header覆盖了整个文件。

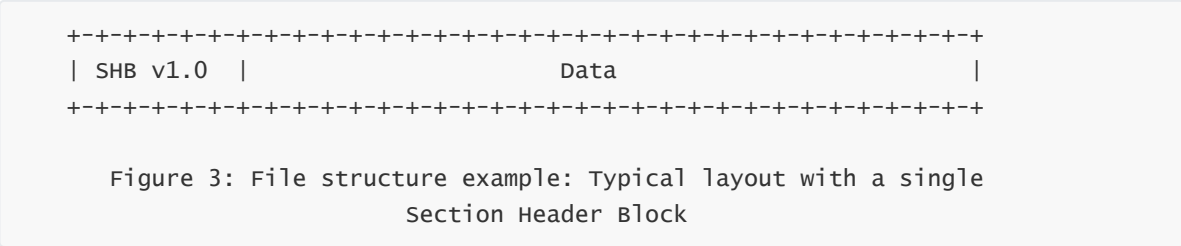


Figure 3: File structure example: Typical layout with a single Section Header Block

图4 展示了包含3个头部的文件，通常是文件串联的结果。仅理解1.0版本的应用程序跳过了中间的部分并在第三个Section Header之后重新开始处理数据包。

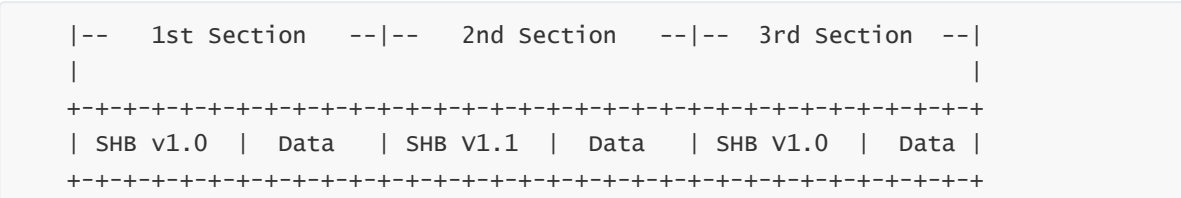


Figure 4: File structure example: three Section Header Blocks in a single file

图5展示了和“经典libpcap”文件相当的文件——最小的有用的捕获文件。 它包含了一个Section Header Block(SHB)，一个Interface Description Block（IDB）和一些Enhanced Packet Blocks（EPB）。

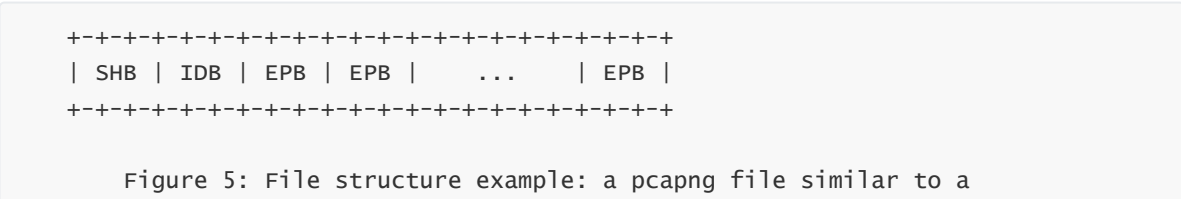


Figure 5: File structure example: a pcapng file similar to a classical libpcap file

图6展示了一个复杂的文件示例。除了上述的最小文件，它包含了从三个接口捕获的数据包，当数据包到达其他接口时，第三个开始捕获，并且还包括了一些Name Resolution Blocks（NRB）和一个Interface Statics Block（ISB）。

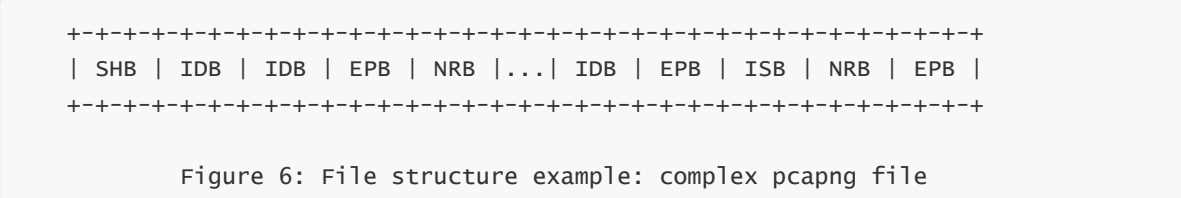


Figure 6: File structure example: complex pcapng file

最后一个例子应该可以看出相较于传统的libpcap形式，块结构使文件格式非常灵活。

3.5 选项

所有的块都可以嵌入可选字段。可选字段可以用来插入一些对于阅读数据有用的信息，但是对于数据包处理来说并不是必须的。

因此，每个工具可以要么读取可选字段的内容，要么跳过其中一些甚至全部内容。

必须对可能包含选项的块进行结构化，以便可以从该数据中确定该选项之前的Block Body中数据的字节数；这样可以发现选项的起点。

所有标准块都支持选项；对于支持选项的Custom Blocks，Custom Data必须以这样的方式进行结构化。这意味着Block Length字段（在通用块结构中，3.1节）可以用来确定可选字段的字节数有多少（如果有可选字段的话）。这个数可以用来确定块是否有可选字段（如果是0，则没有可选字段），检查当处理可选字段时，是否有任何剩余的可选字段，和一次跳过所有的可选字段。

选项是一个类型-长度-值字段的列表，每一个包含一个单独的值：

- **Option Type（16位）**：包含当前特定的TLV记录类型的代码的无符号值。Option Types的最高有效位保留供本地使用；因此，不能保证所使用的代码在所有捕获文件（其他应用程序生成的）中是唯一的，且大多数都不可移植。对于跨平全球唯一的特定于供应商的扩展，必须使用Custom Option，如3.5.1节中定义
- **Option Length (16位)**：包含后续'Option Value'字段的实际长度的无符号值，不包括填充长度。
- **Option Value（可变长度）**：给定选项的值，填充至32位对齐。这个字段的实际长度由Option Length field确定

需要给定块的最新标准选项代码需要发送到pcap-ng-format的邮件列表[2]。

给定选项可能具有固定长度，该选项的所有实例都有和指定的固定长度相同，或具有可变长度，该长度有一个最小值，所有实例的选项都必须和它相同或比它大，固定长度或可变长度选项的最小值会在选项描述中指定。如果可变长度的最小值选项没有被指定，默认最小值为0。读取这些文件的软件应该像报错一样报告无效长度；如果发现非法长度的选项，软件可能停止处理文件，或忽略选项后继续处理文件。写入这些文件的软件不能写入具有非法长度选项的文件。

如果一个选项的值是一个字符串，不一定以0为结尾的。读取这些文件的软件不能假设这些字符串是以0结尾的，但必须将0值的字节视为字符串终止符。

一些选项可能被处理多次；例如，一个块可能有多个注释，如果有多个IPv4或IPv6地址分配给一个Interface Description Block，它可能为接口分配多个IPv4或IPv6地址。 其他选项可能在给定块中至多出现一次。

选项列表由使用特殊'End of Option'代码（opt_endofpoint）的选项终止。写pcapng文件的代码必须在选项列表的最后设置一个opt_endofopt选项。读取pcapng文件的代码不能假定选项列表的末尾存在一个opt_endofopt；它必须检查块的尾部，并且应该在选项列表没有opt_endofopt选项的情况下，像选项列表末尾有opt_endofopt选项一样对待块。 13 + 80 = 93

选项字段的形式如图7所示。

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Option Code              |          Option Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                        Option Value                           /
/              variable length, padded to 32 bits              /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

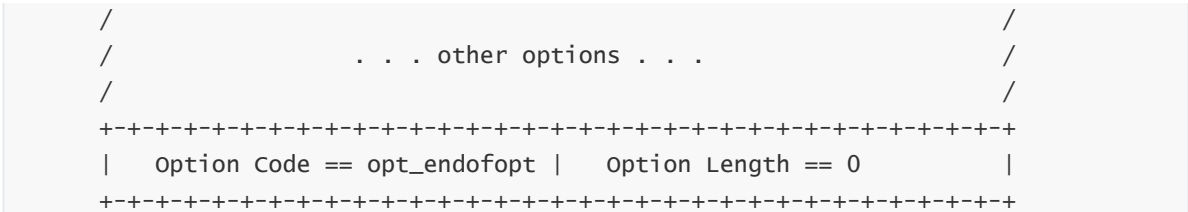


Figure 7: Options Format

下面的代码始终可以出现在任何可选字段中：

Name	Code	Length	Multiple allowed?
opt_endofopt	0	0	no
opt_comment	1	variable	yes
opt_custom	2988/2989/19372/19373	variable, minimum 4	yes

Table 1: Common Options

- opt_endofopt：opt_enofopt选项限制了可选字段的范围。在给定选项列表中不得重复此选项。
- opt_comment：opt_comment选项是一个UTF-8的字符串，它包含了与块相关的人类可读的注释文本。行分隔符应该是回车换行符（'\r\n'）或仅仅是回车符（'\n'）；两种形式都可能会出现，并被视为行分隔符。字符串不以0结尾。
- opt_custom：这个选项在3.5.1中详细描述

3.5.1 自定义选项

自定义选项用于可移植的，特定于与它们相关的供应商的数据。自定义选项可以是具有选项的任何块类型，在一个块中可以重复任意次，并可能在其他选项类型之前或之后 - 除了opt_endofopt（总是最后一个选项）之后。在相同的pcapng文件中可以使用不通的自定义选项，不同理性代码和/不同的私有企业编号，可以在第6节查看细节。

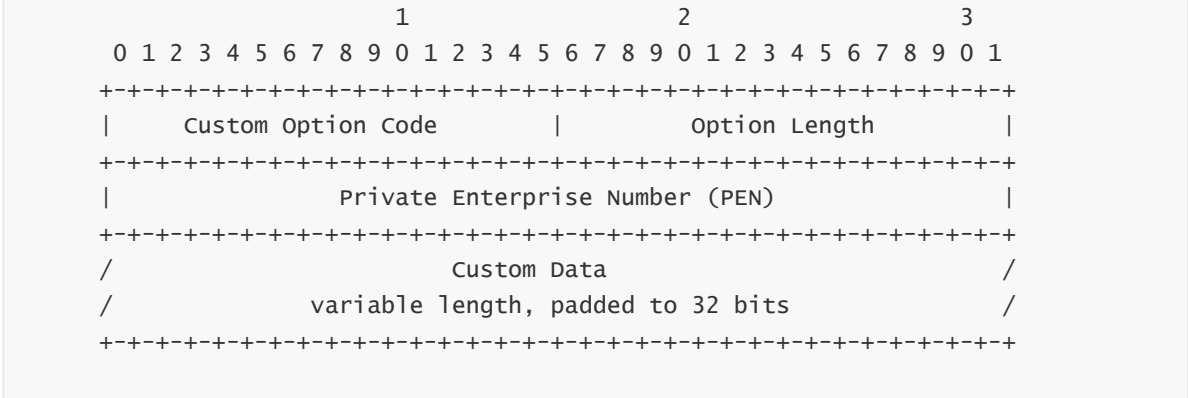


Figure 8: Custom Options Format

自定义选项具有以下字段：

- **Custom Option Code**： Custom Option的代码，可能是下列十进制数之一：

    - 2988：这个选项代码标识了Custom Data部分是UTF-8字符串的Custom Option。该字符串不以0为结尾。如果pcapng文件被一个应用程序操纵，这个Custom Option可以安全地复制到新文件；否则应该使用19372。详情参看6.2节。
    - 2989：这个选项代码标识了Custom Data部分是二进制字节组的Custom Option。如果pcapng文件被一个应用程序操纵，这个Custom Option可以安全地复制到新文件；否则应该使用19373。详情参看6.2节。
    - 19372：这个选项代码标识了Custom Data部分是UTF-8字符串的Custom Option。这个字符串不以0结尾。如果pcapng文件被一个应用程序操纵，这个Custom Option不能被复制到新文件。详情参看6.2节。
    - 19373：这个选项代码标识了Custom Data部分是二进制字节组的Custom Option。如果pcapng文件被一个应用程序操纵，这个Custom Option不能被复制到新文件。详情参看6.2节。
- **Option Length**：像3.1节描述的那样，它包含了选项值的长度，包括4字节的Private Enterprise Number和可变长度的Custom Data字段，不包括填充字段。
- **Private Enterprise Number**：由IANA分配的私营企业号，用于标识定义“Custom Option”的组织。
- **Custom Data**：自定义数据，填充到32位对齐。

3.6 数据形式

3.6.1 字节序

每节包含的数据总是根据捕获设备的特征来保存。这是指所有保存为数字，并且跨越超过2个或更多的字节组的字段。

具有每个部分的方法以生成主机的原生形式保存，因为它避免在自身主机读/写时的数据转换，这是生成/处理捕获文件的最常见情况。

请注意：字节序由Section Header Block表明（4.1节）。由于在一个pcapng文件中一个块可以出现多次，单个文件可以包含两个字节序变体。

3.6.2 对齐

这个说明的所有字段都对16位和32位值使用正确的对齐方式。 在使用如内存映射文件这类技术时，让读/写文件内容更快速简便。

对齐字节组（文章提到的”padded to 32 bits“）必须用0填充。

请注意：64位的值不和64位边界对齐。这是由于文件通常仅对齐32位边界。当读和写这样的数据时必须谨慎考虑。（同时注意，在写入文件的计算机的字节序中，某些64位值表示为64位整数；其他64位值代表了两个32位的值，一个包括高32位的值，另一个包括低32位的值，每个像32位机器的字节序一样写入文件。这两种形式都不保证64位对齐）

4.块定义

这节详细介绍了块定义的格式。

4.1 头部块（Section Header Block）

Section Header Block（SHB）是强制的。它标识了捕获文件的起始部分。Section Header Block不包括数据，但它标识了一个逻辑相关的块列表（接口、数据包）。它的形式如图9。

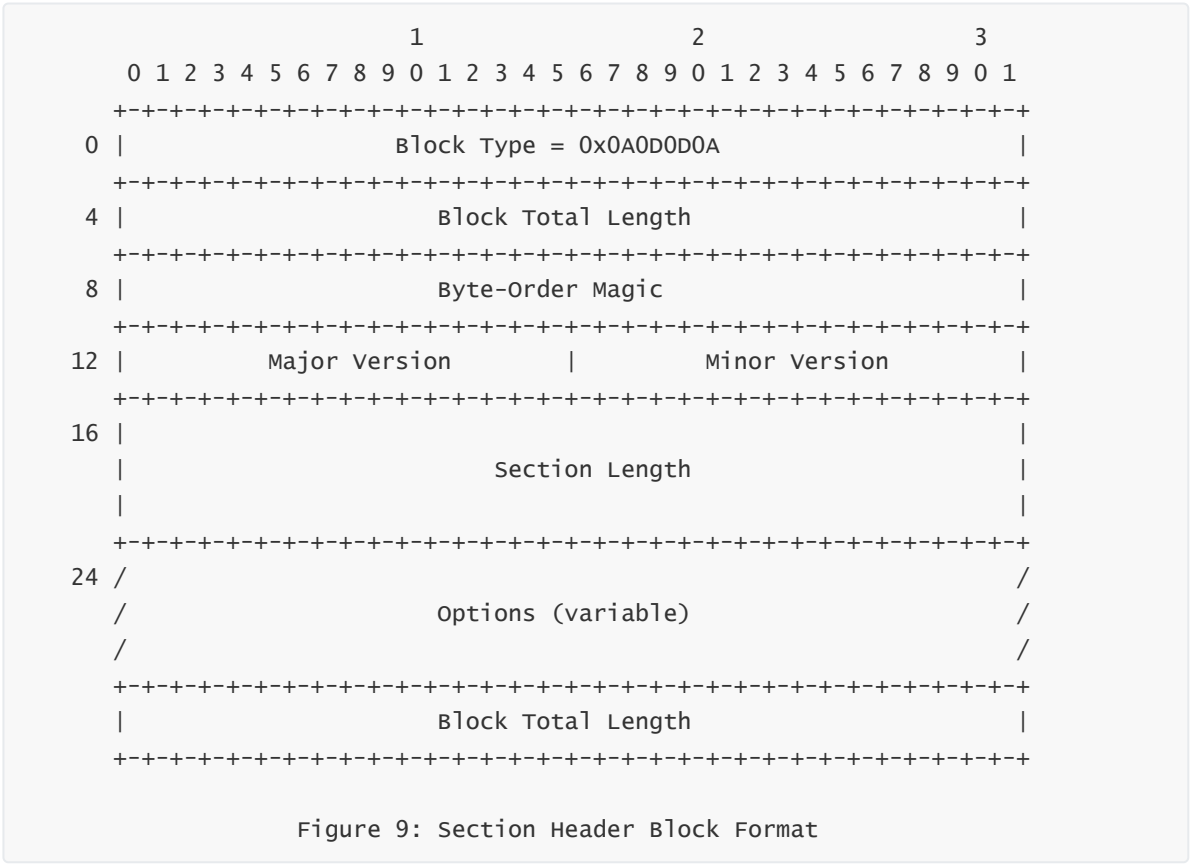


Figure 9: Section Header Block Format

这些字段的含义是：

Block Type：Section Header Block的块类型是对应于4-字符字符串"\n\r\r\n"的整数（`0x0A0D0D0A`）。使用这个特殊值有两个原因：

1.在文件通过FTP或HTTP从一台机器到另一台机器时，用这个数检测是否是在ASCII转换不正确的情况下传输的。在这种情况下，这个字段会和标准（"\n\r\r\n"）不同，并且读者可以发现一个可能崩溃的文件。

2.这个值是回文的，所以读者可以无视这个部分的字节序而认出Section Header Block。字节序由Byte Order Magic识别，位于Block Type后八个字节。

Block Total Length：块的总长度，如3.1节中描述的一样。

Byte-Order Magic（32位）：一个无符号的魔数（魔数，一般用于表示文件格式），它的值是16进制的数`0x1A2B3C4D`。这个数可以被用来区分Section是存储在大端序还是小端序的主机上，并启发式地标识pcapng文件。

Major Version（16位）：一个无符号值，给定当前格式的主版本号。这个值是`1`。 如果格式发生这样的变化，读新格式的代码不能读旧格式和旧格式的代码不能读新格式（即，标识读取新旧两种形式的代码必须检查版本号并为两种形式使用不同的代码路径），这个值会发生变化。需要注意的是添加一个新的块类型或新选项不属于这样的变化。

Minor Version（16位）：一个无符号值，给定当前格式的次要版本号。这个值的当前版本格式是0。如果格式发生这样的改变：读新格式的代码在没有检查版本号但读取旧格式的代码不能读取新格式的全部文件时，这个值会发生变化。

Section Length（64位）：一个有符号数，指定了后续字节组的长度，不包括Section Header Block本身的长度。这个字段用来在大型文件内部跳过Section时的快速导航。如果Section Length是-1（0xFFFFFFFFFFFFFFFF），这意味着section的大小没有被指定，此时跳过这个Section的唯一方式是解析它包含的块。请注意如果这个字段是合法的（即，非负的），它的值始终是4的倍数，因为所有的块都对齐并填充到32位（4个字节）的边界。并且在访问这个字段时必须谨慎考虑，由于文件中所有块都是32位对齐的，这个字段不保证对齐64位边界。这在64位处理器上可能成为问题。

Options：可选的，一个选项的列表（格式请看3.5节中定义的规则）。

添加新的块类型或选项不必改变Major number或Minor number，因为不了解块类型或选项的代码会跳过它；只有跳过块或选项不生效时，minor number才应该发生变化。

除了3.5节中定义的选项外，下列选项在这个块中有效：

```
          +--------------+------+----------+-------------------+
          | Name         | Code | Length   | Multiple allowed? |
          +--------------+------+----------+-------------------+
          | shb_hardware | 2    | variable | no                |
          | shb_os       | 3    | variable | no                |
          | shb_userappl | 4    | variable | no                |
          +--------------+------+----------+-------------------+

                   Table 2: Section Header Block Options
```

shb_hardware：

- shb_hardware选项是创建这个section的硬件描述，是一个UTF-8字符串。这个字符串不是以0结尾的。
- 例如："x86 Persional Computer"，"Sun Sparc Workstation"。

shb_os：

- shb_os选项是用来创建这个section的操作系统，是一个UTF-8字符串。这个字符串不是以0结尾的。
- 例如："Windows XP SP2"，"openSUSE 10.2"

shb_userappl：

- sub_userappl选项是用来创建这个section的应用名称，是一个UTF-8字符串。这个字符串不是以0结尾的。
- 例如："dumpcap V0.99.7"

[未解决的问题：重写捕获文件的程序改变了原始硬件/系统/应用的信息了吗？]

4.2 接口描述块（Interface Description Block）

接口描述块（IDB）是一个描述了数据包数据捕获接口的容器。

写/读捕获文件的工具，该工具将无符号递增数和每个Interface Description Block联系起来，称作相关接口的接口ID。这个数在每个Section中都是唯一的，并且标识了IDB相关的接口；它仅仅在当前section的内部唯一，因此，两个Section可能具有由相同Interface ID标识的不同接口。这个唯一的标识符被其他块引用，如Enhanced Packet Blocks和Interface Statistic Blocks，以表明和块相关的接口（例如用于

Enhanced Packet Block所包含的数据包的接口或Interface Statistic Block中的同进信息所引用的接口）。

每个接口必须有一个Interface Description Block，以便于其他块引用。如Enhanced Packet Block或Interface Statistic Block这样的块包含引用特定接口的Interface ID值，而Simple Packet Block的隐式引用Interface ID为0的接口。如果文件不包含任何使用Interface ID的块，则文件不需要Interface Description Block。

一个Interface Description Block仅在它属于的section中有效。Interface Description Block的结构如图10。

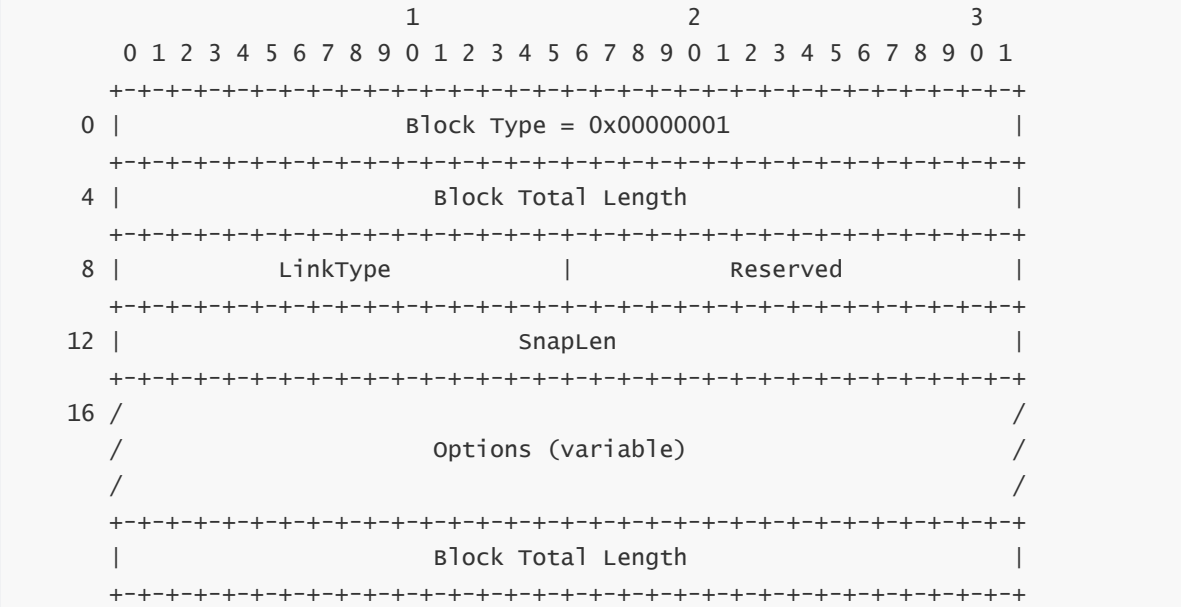


Figure 10: Interface Description Block Format

字段的含义如下：

Block Type：Interface Description Block，的块类型是`1`。

Block Total Length：这个块的总大小，像3.1节中描述的那样。

LinkType（16位）：一个无符号值，定义了这个接口的链路层类型。标准链路层类型代码的列表可以在[LINKTYPES]中获得。

Reserved（16位）：不使用 - 必须通过pcapng文件写入者用0填充，并且必须被pcapng文件读者忽略。

SnapLen（32位）：一个无符号值，表明每个捕获的数据包的最大字节数。每个数据包超过这个值的部分不会被存储在文件里。0值表示没有限制。

Options：可选的，一个选项的列表（格式的相关规则定义在3.5节中）。

除了定义在3.5节中的选项，下列选项在这个块中也是有效的：

Name	Code	Length	Multiple allowed?
if_name	2	variable	no
if_description	3	variable	no
if_IPv4addr	4	8	yes
if_IPv6addr	5	17	yes
if_MACaddr	6	6	no

```
    | if_EUIaddr     | 7    | 8                    | no                |
    | if_speed       | 8    | 8                    | no                |
    | if_tsresol     | 9    | 1                    | no                |
    | if_tzone       | 10   | 4                    | no                |
    | if_filter      | 11   | variable, minimum 1  | no                |
    | if_os          | 12   | variable             | no                |
    | if_fcslen      | 13   | 1                    | no                |
    | if_tsoffset    | 14   | 8                    | no                |
    | if_hardware    | 15   | variable             | no                |
    | if_txspeed     | 16   | 8                    | no                |
    | if_rxspeed     | 17   | 8                    | no                |
    +----------------+------+----------------------+-------------------+

                Table 3: Interface Description Block Options
```

if_name:

一个UTF-8字符串，表示用来捕获数据的设备名称。这个字符串不以0结尾。

例如："eth0"，"\Device\NPF_{AD1CE675-96D0-47C5-ADD0-2504B9126B68}"

if_description:

一个UTF-8字符串，捕获数据的设备的描述。这个字符串不以0结尾。

例如："Wi-Fi"，"Local Area Connection"，"Wireless Network Conne"，"First Ethernet Interface"

if_IPv4addr:

接口的IPv4网络地址和对应的网络掩码。前四个字节表示IP地址，接下来四个字节表示掩码。当为接口分配了多个IPv4地址时，这个选项可以在相同的Interface Description Block中重复出现多次。需要注意的是，IP地址和掩码都被视为4个字节，地址或掩码的每个八位组；他们不是32位数，因此SHB的字节序不影响这个字段的值。

例如：'192 168 1 1 255 255 255 0'

if_IPv6addr:

接口的IPv6网络地址和相应的前缀长度。前16个字节是IP地址，接下来的一个字节是前缀长度。当为接口分配了多个IPv6地址时，这个选项可以在相同的Interface Description Block中重复出现多次。

例如： 2001:0db8:85a3:08d3:1319:8a2e:0370:7344/64被写作'20 01 0d b8 85 a3 08 d3 13 19 8a 2e 03 70 73 44 40'

if_MACaddr:

接口硬件MAC地址（48位），如果有。

例如：'00 01 02 03 04 05'

if_EUIaddr:

接口硬件EUI地址（64位），如果有。

例如：'02 34 56 FF FE 78 9A BC'

if_speed:

一个64位的无符号值，表明接口速度，单位bit/s。

例如：64位的十进制数，100Mbps为100000000。

if_tsresol：

标识时间戳的分辨率。如果最大有效位等于0，则剩余位表示时间戳的分辨率是10的负多少次幂。（例如，6表示微秒的分辨率，即10...^-6，时间戳是自1970-01-01 00:00:00 UTC）如果最大有效位是1，剩余位表示分辨率是2的负多少次幂（例如，10，表示1/1024秒）。如果这个选项不存在，则默认分辨率为10^-6（即，此时时间戳和标准libpcap时间戳的分辨率相同）。

例如：'6'

if_tzone：

GMT支持的时区（未完成：更好的定义）

例如：未完成：给出好的例子

if_filter：

用来捕获流量的过滤器（例如，"capture only TCP traffic"）。Option Data的首个字节保持了过滤器使用的代码（例如，如果这是一个libpcap字符串或BPF字节码，或其他）。更多关于这个格式的细节会在附录XXX（未完成）中提出。（未完成：更好的为不同的字段使用不同的选项。例如，if_filter_pcap，if_filter_bpf，...）

例如：'00'"tcp port 23 and host 192.0.2.5"。

if_os：

是一个UTF-8字符串，表示安装这个接口的机器的操作系统的名称。这个可能和具有相同信息的Section Header Block（4.1节）的内容不同，因为捕获操作可能是在远程机器上完成的。这个字符串不以0结尾。

例如："Windows XP SP2"，"openSUSE 10.2"

if_fcslen：

一个无符号的8位整数，指明这个接口的Frame Check Sequence（用bits）的长度。对于FCS长度可能随时间变化的链路层，且可以在Enhanced Packet Block中使用Enhanced Packet Block的epb_flags选项（可以看4.3.1节）。

例如：'4'

if_tsoffset：

一个64位的有符号整数，定义了一个偏移，必须添加到每个数据包的时间戳中，才能获得一个数据包的绝对时间戳。如果这个选项丢失，在数据包存储的时间戳必须被视为绝对时间戳。时区的偏移可以用if_tzone指定。未完成：对于部分秒偏移的if_tsoffset_low对于高度同步的捕获系统会有用吗?

例如：'1234'。

if_hardware：

是一个UTF-8字符串，硬件接口的描述。这个字符串不以0结尾。

例如："Broadcom NetXtreme"，"Intel(R) PRO/1000 MT Network Connection"，"NETGEAR WNA1000Mv2 N150 Wireless USB Micro Adapter"。

if_txspeed：

64位无符号值，表明接口传输速度，单位bit/s。

例如：64位10进制数102400表示1024Kbps。

if_rxspeed：

64位无符号值，表明接口接收速度，单位bit/s。

例如：64位十进制数819200表示8192Kbps。

如果接口传输速度和接收速度相同，必须使用if_speed选项，不可以使用if_txspeed和if_rxspeed选项。如果传输速度未知，不可以使用if_speed和if_txspeed选项；如果接收速度未知，if_speed和if_rxspeed不可以使用。

4.3 加强数据包块（Enhanced Packet Block）

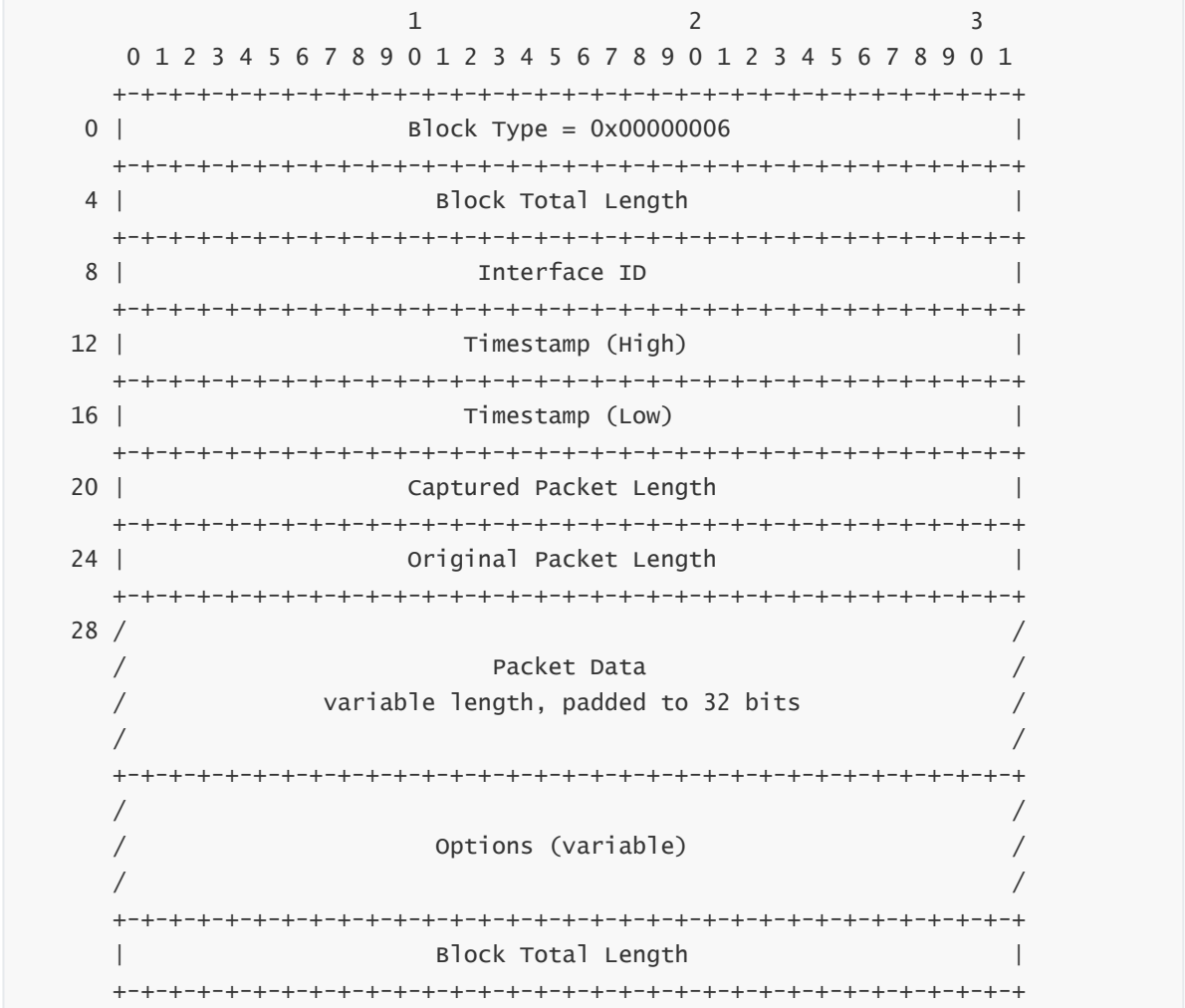
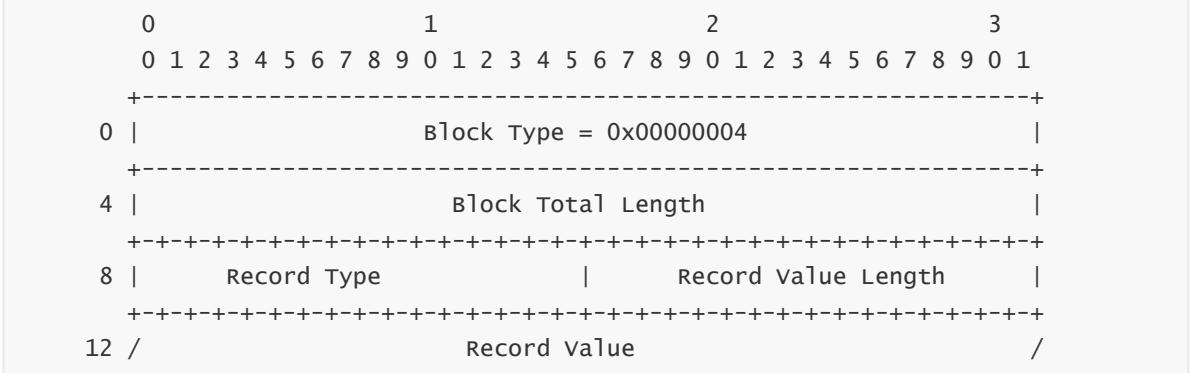


Figure 11: Enhanced Packet Block Format

4.5 名称解析块（Name Resolution Block）



```
/              variable length, padded to 32 bits               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                                                               .
.                  . . . other records . . .                    .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Record Type = nrb_record_end |   Record Value Length = 0     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                      Options (variable)                       /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Block Total Length                       |
+---------------------------------------------------------------+
```

Figure 13: Name Resolution Block Format

参考资料：

1.python-pcapng开发文档：https://python-pcapng.readthedocs.io/en/latest/

2.英文文档：https://github.com/pcapng/pcapng/

3.中文文档：https://wenku.baidu.com/view/7b0b7a7900f69e3143323968011ca300a6c3f644.html#