Trailer Encapsulations

Status of this Memo

   This RFC discusses the motivation for use of "trailer encapsulations"
   on local-area networks and describes the implementation of such an
   encapsulation on various media. This document is for information
   only. This is NOT an official protocol for the ARPA Internet
   community.

Introduction

   A trailer encapsulation is a link level packet format employed by
   4.2BSD UNIX (among others). A trailer encapsulation, or "trailer",
   may be generated by a system under certain conditions in an effort to
   minimize the number and size of memory-to-memory copy operations
   performed by a receiving host when processing a data packet.
   Trailers are strictly a link level packet format and are not visible
   (when properly implemented) in any higher level protocol processing.
   This note cites the motivation behind the trailer encapsulation and
   describes the trailer encapsulation packet formats currently in use
   on 3 Mb/s Experimental Ethernet, 10 Mb/s Ethernet, and 10 Mb/s V2LNI
   ring networks [1].

   The use of a trailer encapsulation was suggested by Greg Chesson, and
   the encapsulation described here was designed by Bill Joy.

Motivation

   Trailers are motivated by the overhead which may be incurred during
   protocol processing when one or more memory to memory copies must be
   performed. Copying can be required at many levels of processing,
   from moving data between the network medium and the host's memory, to
   passing data between the operating system and user address spaces.
   An optimal network implementation would expect to incur zero copy
   operations between delivery of a data packet into host memory and
   presentation of the appropriate data to the receiving process. While
   many packets may not be processed without some copying operations,
   when the host computer provides suitable memory management support it
   may often be possible to avoid copying simply by manipulating the
   appropriate virtual memory hardware.

   In a page mapped virtual memory environment, two prerequisites are
   usually required to achieve the goal of zero copy operations during
   packet processing. Data destined for a receiving agent must be

aligned on a page boundary and must have a size which is a multiple
of the hardware page size (or filled to a page boundary). The latter
restriction assumes virtual memory protection is maintained at the
page level; different architectures may alter these prerequisites.

Data to be transmitted across a network may easily be segmented in
the appropriate size, but unless the encapsulating protocol header
information is fixed in size, alignment to a page boundary is
virtually impossible. Protocol header information may vary in size
due to the use of multiple protocols (each with a different header),
or it may vary in size by agreement (for example, when optional
information is included in the header). To insure page alignment the
header information which prefixes data destined for the receiver must
be reduced to a fixed size; this is normally the case at the link
level of a network. By taking all (possibly) variable length header
information and moving it after the data segment a sending host may
"do its best" in allowing the receiving host the opportunity to
receive data on a page aligned boundary. This rearrangement of data
at the link level to force variable length header information to
"trail" the data is the substance of the trailer encapsulation.

There are several implicit assumptions in the above argument.

   1. The receiving host must be willing to accept trailers. As this
   is a link level encapsulation, unless a host to host negotiation
   is performed (preferably at the link level to avoid violating
   layering principles), only certain hosts will be able to converse,
   or their communication may be significantly impaired if trailer
   packets are mixed with non-trailer packets.

   2. The cost of receiving data on a page aligned boundary should be
   comparable to receiving data on a non-page aligned boundary. If
   the overhead of insuring proper alignment is too high, the savings
   in avoiding copy operations may not be cost effective.

   3. The size of the variable length header information should be
   significantly less than that of the data segment being
   transmitted. It is possible to move trailing information without
   physically copying it, but often implementation constraints and
   the characteristics of the underlying network hardware preclude
   merely remapping the header(s).

   4. The memory to memory copying overhead which is expected to be
   performed by the receiver must be significant enough to warrant
   the added complexity in the both the sending and receiving host
   software.

The first point is well known and the motivation for this note.

Thought has been given to negotiating the user of trailers on a per
host basis using a variant of the Address Resolution Protocol [2]
(actually augmenting the protocol), but at present all systems using
trailers require hosts sharing a network medium to uniformly accept
trailers or never transmit them. (The latter is easily carried out
at boot time in 4.2BSD without modifying the operating system source
code.)

The second point is (to our knowledge) insignificant. While a host
may not be able to take advantage of the alignment and size
properties of a trailer packet, it should nonetheless never hamper
it.

Regarding the third point, let us assume the trailing header
information is copied and not remapped, and consider the header
overhead in the TCP/IP protocols as a representative example [3]. If
we assume both the TCP and IP protocol headers are part of the
variable length header information, then the smallest trailer packet
(generated by a VAX) would have 512 bytes of data and 40+ bytes of
header information (plus the trailer header described later). While
the trailing header could have IP and/or TCP options included this
would normally be rare (one would expect most TCP options, for
example, to be included in the initial connection setup exchange) and
certainly much smaller than 512 bytes. If the data segment is
larger, the ratio decreases and the expected gain due to fewer copies
on the receiving end increases. Given the relative overheads of a
memory to memory copy operation and that of a page map manipulation
(including translation buffer invalidation), the advantage is
obvious.

The fourth issue, we believe, is actually a non-issue. In our
implementation the additional code required to support the trailer
encapsulation amounts to about a dozen lines of code in each link
level "network interface driver". The resulting performance
improvement more than warrants this minor investment in software.
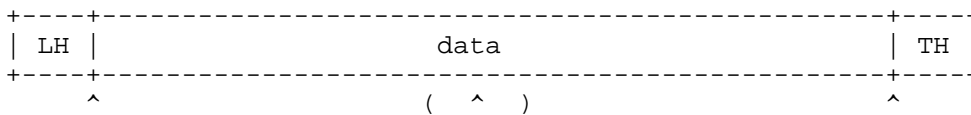
It should be recognized that modifying the network (and normal link)
level format of a packet in the manner described forces the receiving
host to buffer the entire packet before processing. Clever
implementations may parse protocol headers as the packet arrives to
find out the actual size (or network level packet type) of an
incoming message. This allows these implementations to avoid
preallocating maximum sized buffers to incoming packets which it can
recognize as unacceptable. Implementations which parses the network
level format on the fly are violating layering principles which have
been extolled in design for some time (but often violated in
implementation). The problem of postponing link level type

recognition is a valid criticism. In the case of network hardware
which supports DMA, however, the entire packet is always received
before processing begins.

Trailer Encapsulation Packet Formats

   In this section we describe the link level packet formats used on the
   3 Mb/s Experimental Ethernet, and 10 Mb/s Ethernet networks as well
   as the 10 Mb/s V2LNI ring network. The formats used in each case
   differ only in the format and type field values used in each of the
   local area network headers.

   The format of a trailer packet is shown in the following diagram.

```
     +----+--------------------------------------------------+----+
     | LH |                      data                        | TH |
     +----+--------------------------------------------------+----+
       ^                         (  ^  )                        ^
```

       LH:

           The fixed-size local network header. For 10 a Mb/s Ethernet,
           the 16-byte Ethernet header. The type field in the header
           indicates that both the packet type (trailer) and the length of
           the data segment.

           For the 10 Mb/s Ethernet, the types are between 1001 and 1010
           hexadecimal (4096 and 4112 decimal). The type is calculated as
           1000 (hex) plus the number of 512-byte pages of data. A
           maximum of 16 pages of data may be transmitted in a single
           trailer packet (8192 bytes).

       data:

           The "data" portion of the packet. This is normally only data
           to be delivered to the receiving processes (i.e. it contains no
           TCP or IP header information). Data size is always a multiple
           of 512 bytes.
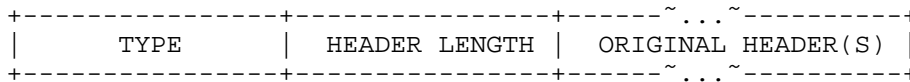
       TH:

           The "trailer". This is actually a composition of the original
           protocol headers and a fixed size trailer prefix which defines
           the type and size
           of the trailing data. The format of a trailer is shown below.

   The carats (^) indicate the page boundaries on which the receiving
   host would place its input buffer for optimal alignment when

receiving a trailer packet. The link level receiving routine is able
to locate the trailer using the size indicated in the link level
header's type field. The receiving routine is expected to discard
the link level header and trailer prefix, and remap the trailing data
segment to the front of the packet to regenerate the original network
level packet format.

Trailer Format

```
+----------------+----------------+------~...~----------+
|      TYPE      | HEADER LENGTH  | ORIGINAL HEADER(S)  |
+----------------+----------------+------~...~----------+
```

Type: 16 bits

   The type field encodes the original link level type of the
   transmitted packet. This is the value which would normally be
   placed in the link level header if a trailer were not generated.

Header length: 16 bits

   The header length field of the trailer data segment. This
   specifies the length in bytes of the following header data.

Original headers: <variable length>

   The header information which logically belongs before the data
   segment. This is normally the network and transport level
   protocol headers.

Summary

   A link level encapsulation which promotes alignment properties
   necessary for the efficient use of virtual memory hardware facilities
   has been described. This encapsulation format is in use on many
   systems and is a standard facility in 4.2BSD UNIX. The encapsulation
   provides an efficient mechanism by which cooperating hosts on a local
   network may obtain significant performance improvements. The use of
   this encapsulation technique currently requires uniform cooperation
   from all hosts on a network; hopefully a per host negotiation
   mechanism may be added to allow consenting hosts to utilize the
   encapsulation in a non-uniform environment.

References

    [1] "The Ethernet - A Local Area Network", Version 1.0, Digital
    Equipment Corporation, Intel Corporation, Xerox Corporation,
    September 1980.

    [2] Plummer, David C., "An Ethernet Address Resolution Protocol",
    RFC-826, Symbolics Cambridge Research Center, November 1982.

    [3] Postel, J., "Internet Protocol", RFC-791, USC/Information
    Sciences Institute, September 1981.