Transport Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 9, 2015

J. Saldana
University of Zaragoza

January 5, 2015

Simplemux. A generic multiplexing protocol
draft-saldana-tsvwg-simplemux-01

Abstract

There are some situations in which multiplexing a number of small packets into a bigger one is desirable. For example, a number of small packets can be sent together between a pair of machines if they share a common network path. Thus, the traffic profile can be shifted from small to larger packets, reducing the network overhead and the number of packets per second to be managed by intermediate routers.

This document describes Simplemux, a protocol able to encapsulate a number of packets belonging to different protocols into a single packet. It includes the "Protocol" field on each multiplexing header, thus allowing the inclusion of a number of packets belonging to different protocols on a packet of another protocol.

The size of the multiplexing headers is kept very low (it may be a single byte when multiplexing small packets) in order to reduce the overhead.

Status of This Memo

This Internet-Draft is submitted to IETF 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 http://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 July 9, 2015.

Table of Contents

1. Introduction

This document describes Simplemux, a protocol able to encapsulate a number of packets belonging to different protocols into a single packet. This can be useful e.g. for grouping small packets and thus reducing the number of packets per second in a network.

This proposal attempts to be general, meaning that it can be used for multiplexing packets belonging to a generic protocol on a single packet belonging to other (or the same) protocol). Thus, we will talk about the "multiplexed" protocol, and the "multiplexing" protocol. The "external header" will be the one of the "multiplexing" protocol (see the figure (Figure 1)).

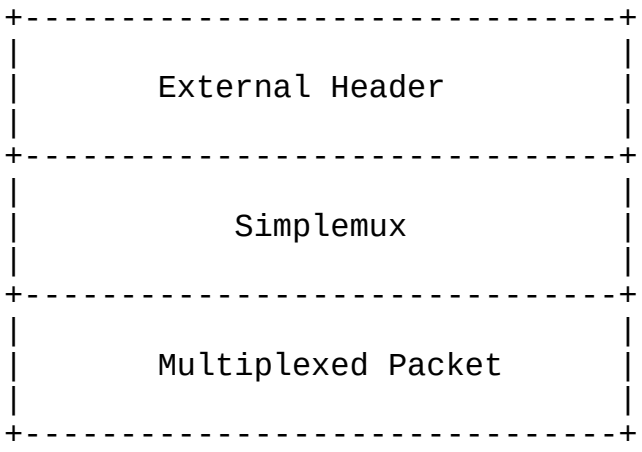


Figure 1

For example, if a number of IPv6 packets have to travel over an IPv4 network, they can be multiplexed into a single IPv4 packet. In this case, IPv4 is the "multiplexing" protocol and IPv6 is the "multiplexed" protocol. The IPv4 header is called in this case the "external header". The scheme of this packet would be:

|IPv4 hdr||Smux hdr|IPv6 packet||Smux hdr|IPv6 packet|| ...|

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Existing multiplexing protocols

Different multiplexing protocols have been approved by the IETF in the past:

o TMux [RFC1692]

TMux is able to combine multiple short transport segments, independent of application type, and send them between a server and host pair. As stated in the reference, "The TMux protocol is intended to optimize the transmission of large numbers of small data packets. In particular, communication load is not measured only in bits per seconds but also in packets per seconds, and in many situation the latter is the true performance limit, not the former. The proposed multiplexing is aimed at alleviating this situation."

A TMux message appears as:

|IP hdr||TMux hdr|Transport segment||TMux hdr|Transport segment||...|

So the Transport Segment is not an IP packet, since it does not include the IP header.

TMux works "between a server and host pair", so it multiplexes a number of segments between the same pair of machines. However, there are scenarios where a number of low-efficiency flows share a common path, but they are not sent between the same pair of machines.

o PPPMux [RFC3153]

PPPMux "sends multiple PPP encapsulated packets in a single PPP frame. As a result, the PPP overhead per packet is reduced." Thus, it is able to multiplex complete IP packets, using separators.

However, the use of PPPMux requires the use of PPP and L2TP in order to multiplex a number of packets together, as done in TCRTP [RFC4170]. However, this introduces more overhead and complexity.

An IP packet including a number of them using PPPMux appears as:

|IP hdr|L2TP hdr|PPP hdr||PPPMux hdr|packet||PPPMux hdr|packet||...|

The scheme proposed by PPPMux is similar to the Compound-Frames of PPP LCP Extensions [RFC1570]. The key differences are that PPPMux is more efficient and that it allows concatenation of variable sized frames.

***

The definition of a protocol able to multiplex complete packets, avoiding the need of other protocols as PPP is seen as convenient. The multiplexed packets can be of any kind, since the "Protocol" field can be added for each of them. Not all the packets multiplexed in the same one have to belong to the same protocol. The general scheme of Simplemux is:

|external hdr||Simplemux hdr|packet||Simplemux hdr|packet||...|

The Simplemux header includes the "Protocol" field, so it permits the multiplexing of different kinds of packets in the same bundle.

We will also refer to the Simplemux header with the terms "separator", "Simplemux separator" or "mux separator".

When applied to IP packets, the scheme of a multiplexed packet becomes:

|IP hdr||Simplemux hdr|IP packet||Simplemux hdr|IP packet||...|

1.3. Benefits of multiplexing

The benefits of multiplexing are:

- Tunneling a number of packets together. If a number of packets have to be tunneled through a network segment, they can be multiplexed and then sent together using a single external header. This will avoid the need for adding a tunneling header to each of the packets, thus reducing the overhead.

- Reduction of the amount of packets per second in the network. It is desirable for two main reasons: first, network equipment has a limitation in terms of the number of packets per second it can manage, i.e. many devices are not able to send small packets back to back due to processing delay.

- Bandwidth reduction. The presence of high rates of tiny packets translate into an inefficient usage of network resources, so there is a need for mechanisms able to reduce the overhead introduced by low-efficiency flows. When combined with header compression, as done in TCRTP [RFC4170] multiplexing may produce significant bandwidth savings, which are interesting for network operators, since they may alleviate the traffic load in their networks.

- Energy savings: a lower amount of bandwidth packets per second will reduce energy consumption in network equipment since, according to [Bolla], internal packet processing engines and switching fabric require 60% and 18% of the power consumption of high-end routers respectively. Thus, reducing the number of packets to be managed and switched will reduce the overall energy consumption. The measurements deployed in [Chabarek]on commercial routers corroborate this: a study using different packet sizes was presented, and the tests with big packets showed that energy consumption gets reduced, since a non-negligible amount of energy is associated to header processing tasks, and not only to the sending of the packet itself.

2. Description of the scenario

Simplemux works between a pair of machines. It creates a tunnel between the ingress and the egress. They MAY be the endpoints of the communication, but they MAY also be middleboxes able to multiplex packets belonging to different flows. Different mechanisms MAY be used in order to classify flows according to some criteria (sharing a common path, kind of service, etc.) and to select the flows to be multiplexed and sent to the egress (see Figure 2).

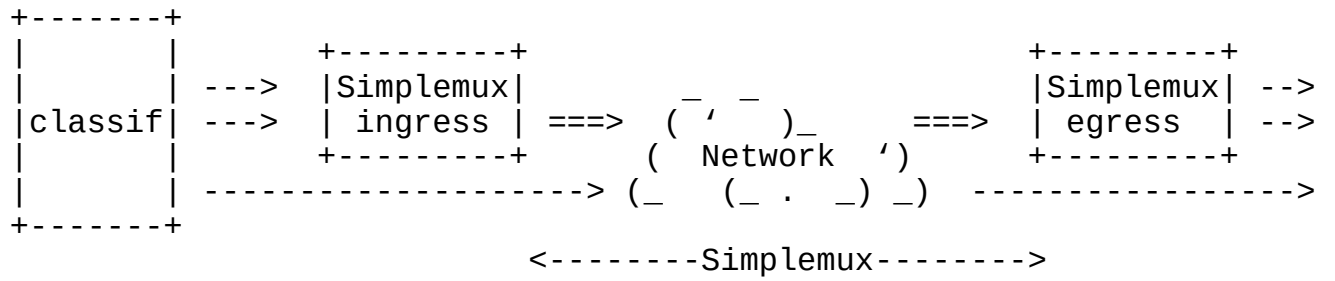


Figure 2

3. Protocol description

A Simplemux packet consists of:

- An external header which is used as the tunneling header for the whole packet.

- A series of pairs "Simplemux header" + "packet".

This is the scheme of a Simplemux packet:

|external hdr||Smux hdr|packet||Smux hdr|packet||...|

The Simplemux header has two different forms: one for the First Simplemux header, and another one for the rest of the Simplemux headers (Non-first Simplemux headers).

o First Simplemux header (before the first multiplexed packet):

| Protocol (8 bits)|SPB(1 bit)| LXT(1 bit) | length (6 or 14 bits) |

- Protocol (8 bits) is the Protocol field of the multiplexed packet, according to IANA "Assigned Internet Protocol Numbers".

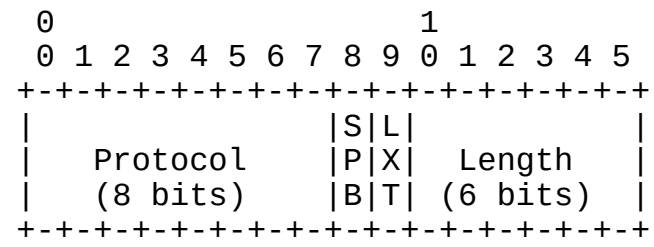
- Single Protocol Bit (SPB, one bit) only appears in the first Simplemux header. It is set to 1 if all the multiplexed packets belong to the same protocol (in this case, the "Protocol" field will only appear in the first Simplemux header). It is set to 0 when each packet MAY belong to a different protocol.

- Length Extension (LXT, one bit) is 0 if the length of the first packet can be expressed in 6 bits, and 1 in other case.

- Length (LEN, 6 or 14 bits): This is the length of the multiplexed packet in bytes not including the length field. If the length of the multiplexed packet is less than 64 bytes (less than or equal to 63

bytes), LXT is set to 0 and the 6 bits of the length field are the length of the multiplexed packet. If the length of the multiplexed packet is greater than 63 bytes, LXT is set to 1 and the 14 bits of the length field are the length of the multiplexed packet. The maximum length of a multiplexed packet is 16,383 bytes. Packets larger than 16,383 bytes will need to be sent in their native form. A Simplemux ingress is not required to multiplex all packets smaller than 16,383 bytes. It may choose to only multiplex packets smaller than a configurable size into a Simplemux multiplexed packet.

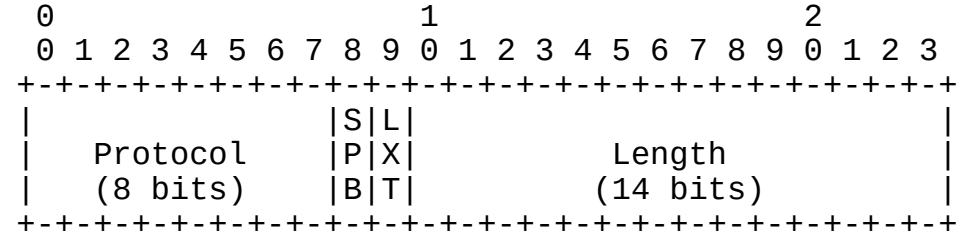
As an example, a First Simplemux header before a packet smaller than 64 bytes would be 16 bits long:



LXT = 0

Figure 3

And a First Simplemux header before a packet bigger than 63 bytes would be 24 bits long:



LXT = 1

Figure 4

o Subsequent (Non-first) Simplemux headers (before the other packets):

| Protocol(8 bits, optional) |LXT(1 bit) | length (7 or 15 bits) |

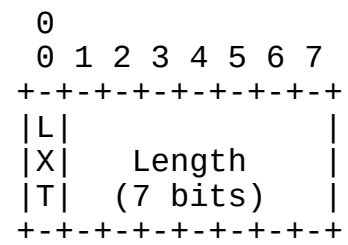
- Protocol (8 bits) is the Protocol field of the multiplexed packet, according to IANA "Assigned Internet Protocol Numbers". It only

appears in Non-first headers if the Single Protocol Bit (SPB) of the First Simplemux header is set to 1.

- Length Extension (LXT, one bit) is 0 if the length of the first packet can be expressed in 7 bits, and 1 in other case.

- Length (LEN, 7 or 15 bits): This is the length of the multiplexed packet in bytes not including the length field. If the length of the multiplexed packet is less than 128 bytes (less than or equal to 127 bytes), LXT is set to 0 and the 7 bits of the length field represent the length of the multiplexed packet. If the length of the multiplexed packet is greater than 127 bytes, LXT is set to 1 and the 15 bits of the length field are the length of the multiplexed packet. The maximum length of a multiplexed packet is 32,768 bytes. Packets larger than 32,768 bytes will need to be sent in their native form. However, this will have to be reduced to 16,383 bytes taking into account that the maximum size of the First header is 14 bits. A Simplemux ingress is not required to multiplex all packets smaller than 32,768 bytes. It may choose to only multiplex packets smaller than a configurable size into a Simplemux multiplexed packet.

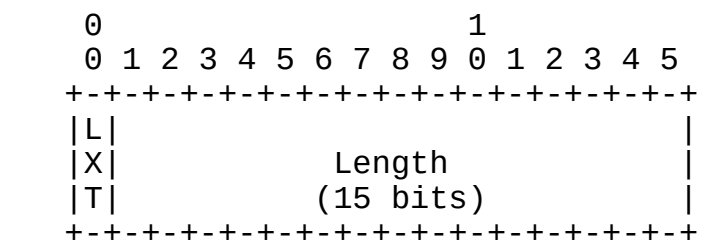
As an example, a Non-first Simplemux header before a packet smaller than 128 bytes, when the protocol bit has been set to 0 in the first header, would be 8 bits long:



LXT = 0
SPB = 0 in the first header

Figure 5

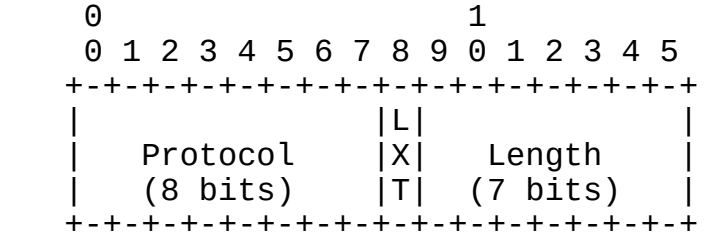
A Non-first Simplemux header before a packet bigger than 127 bytes, when the protocol bit has been set to 0 in the first header, would be 16 bits long:



LXT = 1
SPB = 0 in the first header

Figure 6

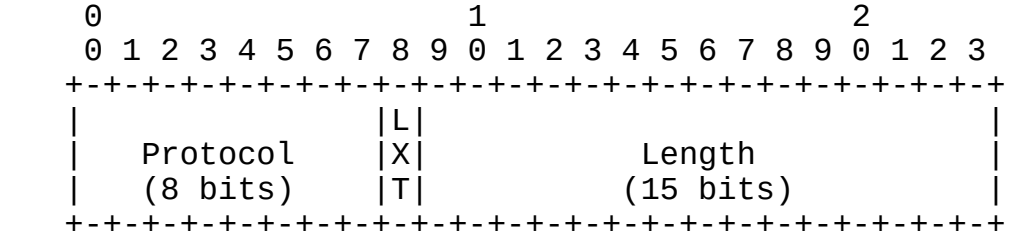
A Non-first Simplemux header before a packet smaller than 128 bytes, when the protocol bit has been set to 1 in the first header, would be 16 bits long:



LXT = 0
SPB = 1 in the first header

Figure 7

And a Non-first Simplemux header before a packet bigger than 127 bytes, when the protocol bit has been set to 1 in the first header, would be 24 bits long:



LXT = 1
SPB = 1 in the first header

Figure 8

These would be some examples of the whole bundles:

Case 1: All the packets belong to the same protocol: The first Simplemux header would be 2 or 3 bytes, and the other Simplemux headers would be 1 or 2 bytes. For small packets (< 128 bytes), the Simplemux header would only require one byte.

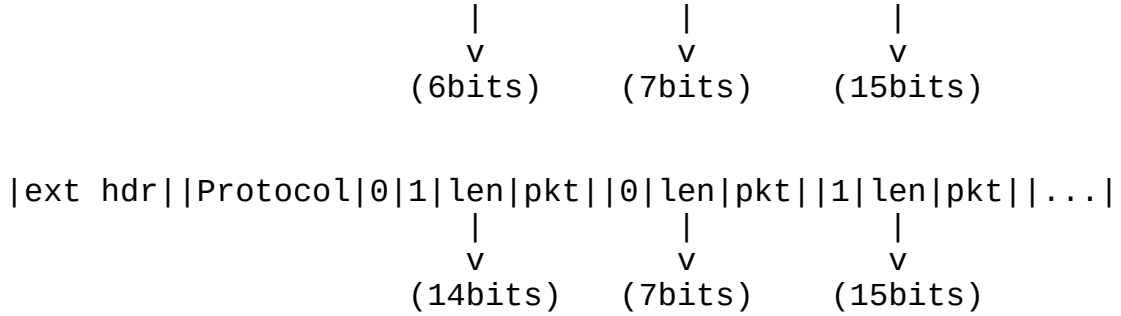


Figure 9

Case 2: Each packet may belong to a different protocol: All the Simplemux headers would be 2 or 3 bytes.

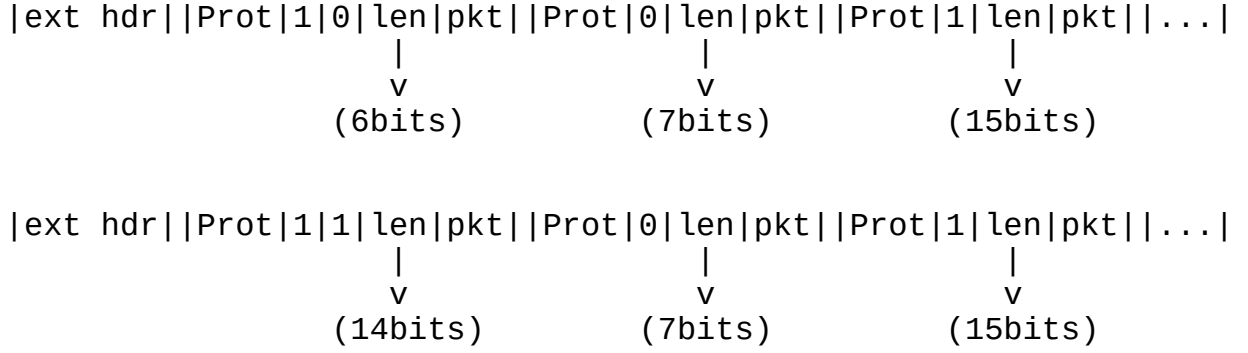


Figure 10

4. Acknowledgements

5. IANA Considerations

A protocol number should be requested to IANA for Simplemux.

As a provisional solution for IP networks, the ingress and the egress optimizers may agree on a UDP port, and use IP/UDP as the multiplexing protocol.

6. Security Considerations

7. References

7.1. Normative References

[RFC1570] Simpson, W., "PPP LCP Extensions", RFC 1570, January 1994.

[RFC1692] Cameron, P., Crocker, D., Cohen, D., and J. Postel, "Transport Multiplexing Protocol (TMux)", RFC 1692, August 1994.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3153] Pazhyannur, R., Ali, I., and C. Fox, "PPP Multiplexing", RFC 3153, August 2001.

[RFC4170] Thompson, B., Koren, T., and D. Wing, "Tunneling Multiplexed Compressed RTP (TCRTP)", BCP 110, RFC 4170, November 2005.

7.2. Informative References

[Bolla] Bolla, R., Bruschi, R., Davoli, F., and F. Cucchietti, "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures", IEEE Communications Surveys and Tutorials vol.13, no.2, pp.223,244, 2011.

[Chabarek] Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., and S. Wright, "Power Awareness in Network Design and Routing", INFOCOM 2008. The 27th Conference on Computer Communications. IEEE pp.457,465, 2008.

Author's Address

Jose Saldana
University of Zaragoza
Dpt. IEC Ada Byron Building
Zaragoza 50018
Spain

Phone: +34 976 762 698
Email: jsaldana@unizar.es