Transport Area Working Group
Internet-draft
Expires: January 2001

S. Bailey
Sandburst
J. Pinkerton
Microsoft
C. Sapuntzakis
Cisco
M. Wakeley
Agilent
J. Wendt
HP
J. Williams
Emulex

# ULP Framing for TCP
draft-ietf-tsvwg-tcp-ulp-frame-00

## Status of this Memo



## Copyright Notice



## Abstract

The framing protocol accepts PDUs from a ULP (upper level protocol) and transports them over a TCP connection. This is done in such a

way that the PDUs can be recovered at the receiver even if preceding TCP segments have not yet been received. This is useful when the PDUs are self describing within the context of a protocol TCP connection. In this case, the framing protocol allows incoming packets to be parsed (but not processed) in the order received and their data to be placed directly in the ultimate destination memory instead of TCP reassembly buffers.

Table Of Contents

## 1. Introduction

Many upper layer protocols (ULP)s, particularly those which perform bulk data transfer, permit the final location of transferred data





(e.g. a ULP client buffer) to be known when the data is received. The information required to compute the final location of such data is contained in local protocol state and ULP protocol data unit (PDU) headers. In this case, ULP data can be placed directly at its final destination by a network interface with knowledge of the ULP. A direct placement network interface can offer extremely high performance since the host CPU does not copy the data at all, and the data only crosses system buses once.

Both specific application ULPs, such as iSCSI, and generic hardware acceleration ULPs, such as an RDMA protocol, offer the potential for direct data placement. The advantage of using a generic acceleration ULP for direct data placement is that the same direct placement network interface can be used to accelerate many different application protocols (e.g. iSCSI on RDMA).

PDU shall mean ULP PDU for the remainder of the document unless otherwise indicated.

TCP specifies that the ULP is notified of the delivery of octets in the order in which they are presented to the sender. Many ULPs rely on this sequencing guarantee. While notification from TCP is required to be in-order, this does not prohibit arbitrary placement of TCP data received in any order. Even if data for a ULP is placed out-of-order, the ULP may still only be notified of of such data in-order, in accordance with TCP semantics. In other words, direct data placement based upon ULP information is not at odds with TCP's stream-orientation, but rather is a natural application of TCP's philosophy that ULP PDU framing be performed at the layer above TCP. RFC 879 also points out in its discussion of layering and modularity that this type of behavior is completely in harmony with layered protocol design [RFC0879].

Packet delay, loss and reordering are expected, common occurrences in IP networks. Traditionally, data in TCP segments is placed in an intermediate reassembly buffer to restore the sending order

which may have been lost as a result of segment delay, loss or reordering. While it is possible for a direct placement network interface to implement a complete reassembly buffer, the cost of doing so is prohibitive. Such a reassembly buffer would need to have a size equal to the sum of the maximum window sizes of all active connections. On a fast network link (e.g. > 1 Gb/s), the window size for each connection can be very large, which would require a huge, very high speed reassembly buffer on the network interface.

A way to find PDUs when previous PDU headers are in delayed, lost or reordered segments will permit data in these subsequent PDUs to





be placed immediately by a direct placement network interface. This will reduce the buffer requirements for a direct placement network interface. Without such a mechanism, the data from subsequent PDUs must all be buffered in the adapter until all previous TCP segments are received. Initial discussion of this issue, and how it relates specifically to iSCSI can be found in an early iSCSI design team memo [Satran].

This document specifies a protocol with two modes for efficiently finding PDUs in the presence of lost, delayed or reordered TCP segments.

## 2. Theory Of Operation

One very efficient way to guarantee that subsequent PDUs can always be found when a previous PDU header has been lost is to ensure each TCP segment begins with a PDU and contains an integral number of PDUs. In this case, the data in each TCP segment may be placed independently of all other segments. No reassembly buffer is required at all. Guaranteeing a TCP segment begins with a PDU requires a modification to TCP's sending behavior. This document defines the behavior of a TCP with a modified sender behavior, called a `framing-aware TCP'. A framing-aware TCP allows a ULP implementation to ensure that each TCP segment begins with a PDU. A framing-aware TCP is fully compliant with all RFCs governing TCP and fully interoperable with existing, compliant, non-framing-aware TCP implementations. When the framing protocol can use a framing-

aware TCP, it operates in `PDU alignment mode'. The framing protocol in PDU alignment mode uses a combination of a framing-aware TCP and an encapsulation of PDUs to permit error free PDU location when TCP segments are lost.

Another way to locate PDUs in the presence of lost TCP segments is to insert markers at a known period in the TCP octet stream. Each marker points to the beginning of the next PDU. If the marker frequency is high relative to packet loss rate (e.g. once per TCP segment), the receiver can, with very high likelihood, learn the location of the next PDU from a marker even when a previous PDU header has been lost. The receiver must still buffer the octets between the lost TCP segment and the subsequent PDU, but this is likely to be a much smaller buffer than the maximum TCP window size. By limiting the maximum PDU size, the receiver buffering can be reasonably bounded. This document defines a periodic marker mechanism which can be used to bound receiver reassembly buffers.

Two framing protocol modes are defined because of the substantial





tradeoff between the modes. Both modes can bound reassembly buffer on a direct placement network interface, but the modes apply in disjoint circumstances.

Marker mode has the following advantage:

1. Implementable without TCP sender modification

The PDU alignment mode has the following advantages:

1. No reassembly buffering required at all

2. Placement information is always at the start of a TCP segment, substantially simplifying hardware processing

PDU alignment mode is more powerful, and is preferable when available. Marker mode still requires some high-speed reassembly memory, whose size is a linear function of the number of active TCP connections. Furthermore, marker mode only offers a probabilistic bound on the reassembly buffer size per active TCP connection. In cases where many TCP segments with PDU headers are lost, the buffer size required for direct placement could approach that of a

complete reassembly buffer.

It is expected that ultimately PDU alignment mode will dominate because of compelling cost and performance scalability advantages. However, until framing-aware TCPs are ubiquitous, marker mode offers an alternative for use with an unmodified TCP implementation. To make transition from marker mode to PDU alignment mode easy, the sockets API extension defined in Appendix A supports both modes relatively transparently. A ULP which implements the behavior required for PDU alignment mode can use marker mode without modification.

Framing protocol receivers MAY implement either PDU alignment mode, or marker mode, or both. Framing protocol senders, MUST implement marker mode, and MUST implement PDU alignment mode if the underlying TCP is framing-aware.

## 3. ULP Support For Framing

A ULP using the framing protocol will submit each complete PDU to the framing module in a single sending operation. This behavior is already common practice for most ULP implementations.

When the framing protocol is in PDU alignment mode, each PDU





submitted is limited to the smaller of 2^16-8 (65528) and the size that will fit entirely within a TCP segment. The framing protocol in PDU alignment mode MUST fail any attempt to submit a PDU that is larger than will fit with an 8-byte framing header in a TCP segment.

The TCP maximum segment size (MSS) is defined in RFC 793 [TCP] as the segment size exchanged on TCP connection establishment. In addition, there is the segment size presently used by TCP which is less than or equal to the exchanged MSS, adjusted by the current path MTU [PathMTU]. This document calls the MSS presently in use the `effective maximum segment size' (EMSS). The EMSS is of primary concern to the framing protocol in PDU alignment mode.

The TCP EMSS can shrink to 8 octets [PathMTU] which leaves no room

for a PDU in PDU alignment mode. If the EMSS goes below 512 octets, the ULP MAY instruct the framing protocol to enter an "emergency mode." In this mode, the framing module MUST accept PDUs up to 512 octets and MAY fragment a PDU across TCP segments.

The EMSS may change during the course of the connection. The framing module in PDU alignment mode MUST notify the ULP sender of changes in the EMSS. The framing module in PDU alignment mode MUST provide the current value of the path EMSS to the ULP on request.

When the framing protocol is in marker mode, each PDU submitted is limited to 2^16-8 minus the size of all interspersed markers. The framing protocol in marker mode MUST fail any attempt to submit a PDU larger than this limit. The framing module MAY impose a smaller, implementation specific size limit on PDUs. In order to effectively bound the receiver's reassembly buffer size, the ULP SHOULD submit PDUs limited in size by some appropriate function of the receiver's reassembly buffer resources, but no specific limit is imposed by the framing protocol.

## 4. Negotiating Use Of The Framing Protocol

Negotiating use of the framing protocol is the responsibility of the ULP. The use of the framing protocol MAY be negotiated separately for each direction on a particular connection. The negotiation procedure MUST ensure that when receive framing is enabled, the remote peer will not transmit the first TCP segment with framed data until it is certain that the local peer has actually enabled receive framing.

If a receiver requests PDU alignment mode, and the sender supports





PDU alignment mode, then the sender MUST enable PDU alignment mode. This ensures that PDU alignment mode, with its favorable hardware characteristics, is used when possible.

The specific negotiation mechanism for enabling the framing protocol and choosing the framing mode is outside the scope of this document. However, note that framing protocol behavior is requested by the receiver and offered by the sender. Negotiation

will probably include exchange of:

1. the receiver's desired mode(s)

2. the sender's framing key if PDU alignment mode is selected

2. ULP packing behavior if PDU alignment mode is selected

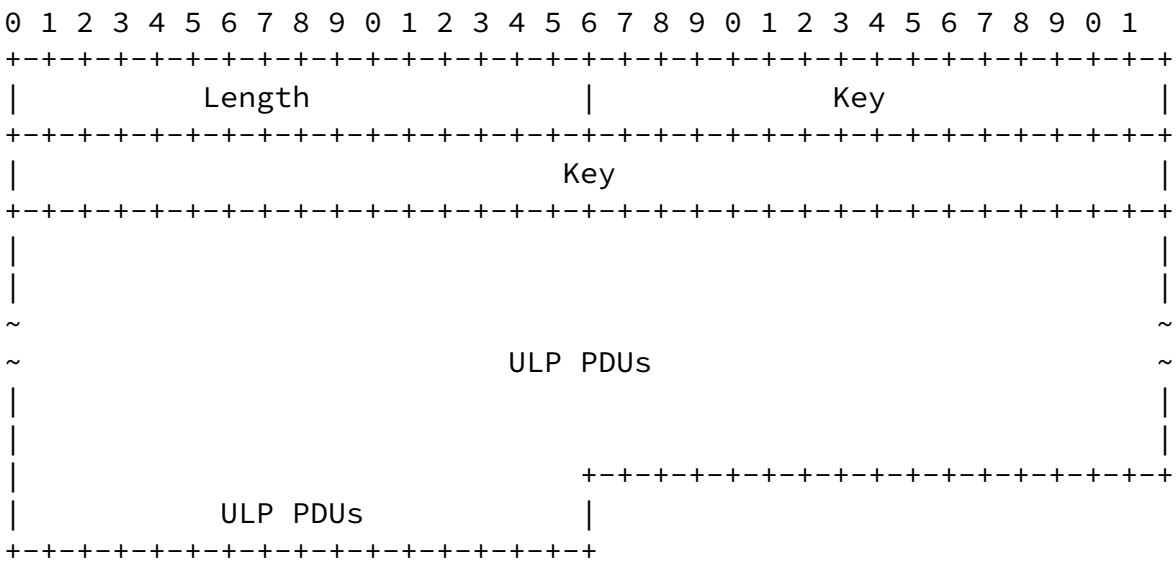
3. the receiver's desired marker period if marker mode is selected

4. the receiver's desired maximum PDU size if marker mode is selected

## 5. PDU Alignment Mode

The framing protocol in PDU alignment mode sends one or more complete ULP PDUs preceded by a framing header. This framing header and set of ULP PDUs is called a `framing PDU'. The framing protocol in PDU alignment mode is supported by a framing-aware TCP whose behavior is described in `Framing-Aware TCP', below.

The format of a framing PDU is as follows:




The "Length" field is 16 bits and contains the length in octets of the set of framed ULP PDUs, excluding the framing header.

The "Key" field is 48 bits and is selected at random by the sender, and signalled to the receiver in a ULP-specified way. All framing PDUs sent on the same connection in the same direction must use the same key value. A good quality random number generator MUST be used to generate the initial key. RFC 1750 discusses relevant characteristics and provides references for good quality random number generation [RFC1750].

The length of the framing PDU in octets will be 8 + L, where L is the length of the set of framed ULP PDUs.

Whether more than one ULP PDU may be packed into a single framing PDU is a controllable option of the framing module in PDU alignment mode. Some receivers may choose to expect exactly one ULP PDU per TCP segment when framing is behaving nominally. The sender MUST NOT pack more than one ULP PDU into a framing PDU if this behavior is desired by the receiver. ULP packing behavior may be negotiated or specified priori by the ULP.

## 5.1. Framing-aware TCP

A framing-aware TCP SHALL send one complete framing PDU per TCP segment whenever possible. Cases when it may not be possible to send a complete framing PDU in each TCP segment are described in `PDU Alignment Mode Exception Cases', below.

A framing-aware TCP MUST NOT send any TCP segment containing octets from more than one sending operation. In other words, the boundary between data of consecutive sending operations MUST occur between TCP segments. By following this rule, the sender guarantees that in the event an exception causes PDU alignment to be lost temporarily, it will be regained as soon as possible.

The use of oversize TCP segments sent by means of IP fragmentation is discouraged due to the limited size of the IP header Identification field and the potential for undetected errors due to wrapping of the Identification value. Framing-aware TCP implementations SHOULD resegment at the TCP layer according to the rule given in the previous paragraph when necessary to meet requirements of the current maximum segment size for a path. In this document, EMSS means the current TCP maximum segment size used for sending segments on a connection, which is initially negotiated during the connection handshake, and subsequently adjusted by path maximum transfer unit (PMTU) discovery behavior [PathMTU].

A framing-aware TCP must notify the framing module of changes in the EMSS. The framing module must be able to retrieve the EMSS from the framing-aware TCP.

If the framing-aware TCP chooses to probe for path MTU increase using TCP segment larger than the path MTU, the framing-aware TCP MUST report an appropriate EMSS increase. The candidate path MTU will only be probed when the framing protocol submits a framing PDU larger than the current EMSS. Immediately following the probing segment, the framing-aware TCP MUST reduce EMSS to its previous value until the candidate path MTU is confirmed.

Probing for path MTU increase is optional [PathMTU], and a framing-aware TCP might elect not to do so unless the EMSS becomes `inconveniently' small. By not probing for path MTU increase when the current EMSS provides adequate performance, the framing protocol will not send the potentially unaligned PDUs that would be used to probe path MTU.

Although framing-aware TCP is defined specifically to support the framing protocol in ULP alignment mode, it may be used by other clients, assuming framing validation is provided by some means. For example, as discussed below in `Security Considerations', a framing-aware TLS could use a framing-aware TCP directly without adding framing PDU headers, because TLS validation can serve the same purpose, and actually provides stronger framing validations guarantees than a framing PDU header.

## 5.2. PDU Alignment Mode Exception Cases

Although the framing-aware TCP sender should place exactly one framing PDU in each TCP segment there are exceptions when this is not possible. These exceptions include the following.

1. The connection is in emergency mode and EMSS is less than 512 octets.

2. The EMSS has been reduced. This will result in a window during which the ULP is not yet aware of the reduced EMSS. Since some framing PDUs may already have been sent and possibly lost prior to being received, the same framing PDUs must be resent, if necessary, but in smaller TCP segments which conform to the new EMSS.

3. The remote end is advertising a window smaller than the EMSS. If both ends manage their window as required in RFC-1122 [RFC1122], and a reasonable amount of receive buffering is available, this case should not occur, but the sender, for robustness, must tolerate this.

4. The sender is probing an advertised window of zero.

5. The sender is probing to determine if the path MTU can be increased.

In addition, there is another case in which the receiver will receive framing PDUs which are not aligned with TCP segments.

6. There is a middle-box in the connection which is resegmenting the TCP data stream.

If the framing protocol in PDU alignment mode must send an unaligned framing PDU, it SHALL take one of the following actions.

1. Send the framing PDU as a single TCP segment using IP fragmentation. While this behavior is discouraged, it is not prohibited by the framing protocol, or any other applicable RFCs.





2. Send the framing PDU as several TCP segments, with each segment guaranteed not to appear as a well-formed, complete framing PDU on its own, at the time the segment is sent. That is, the sender SHALL ensure that one of the following is true for every segment with a partial framing PDU:

   A. octets 0-1 do not equal the segment length minus 8

   B. octets 2-8 do not match the framing key value

   C. the total segment length is less than the framing PDU header of 8 octets

These mechanisms ensure that the receiver will not falsely misinterpret any piece of a framing PDU sent in several segments as a complete, valid framing PDU. However if the TCP data stream is subjected to resegmenting by a middle-box, the sender may no longer control segmentation of received data. In this case the framing protocol must rely on probability to ensure that segments of the resegmented data stream will not appear as valid, complete framing PDUs, if they are not.

In the case where the receiver detects a continuous stream of TCP segments which do not contain complete framing PDUs, the ULP SHOULD disable use of the framing protocol, or switch to marker mode if the ULP provides a means of doing this, and the end points so choose. Such a continuous stream of improperly framed TCP segments implies the presence of a resegmenting middle-box. Such a detection process SHOULD NOT mistake a temporary sequence of improperly framed TCP segments resulting from an EMSS change with the presence of a resegmenting middle-box

## 5.3. Validity Of Framing-aware TCP Segmentation

A framing-aware TCP normally sends exactly one framing PDU per TCP segment. This may therefore result in more segments being sent than would occur in a traditional TCP. However, the framing module is allowed to pack multiple ULP PDUs into a single framing PDU if ULP packing is enabled, which will give behavior approaching that of a traditional TCP. Even with ULP packing disabled, the behavior of a framing-aware TCP effectively corresponds to that of a traditional TCP sender with the Nagle algorithm disabled (i.e. TCP_NODELAY), and this is considered acceptable behavior.

Framing-aware TCPs still respect congestion control windows, which are maintained as a octet count not as a segment count.





On retransmission, a framing-aware TCP respects the original stream segmentation. This is allowed by RFC1122 [RFC1122], section 4.2.2.15.

## 5.4. Receiving In PDU Alignment Mode

Because each framing PDU contains sufficient information to determine its length, the beginning of the next framing PDU can be determined. Therefore each successive PDU can be recovered.

Conventional TCP implementations will pass received data to the ULP in order, so framing is easily recovered by the ULP.

Special receive implementations which exploit PDU alignment mode, typically found in direct placement network interfaces, may allow the ULP to do direct data placement on TCP segments received out of order. The receiving end can safely assume that a framing PDU is exactly contained within TCP segment payload if the following conditions are met.

1. Standard TCP processing indicates that this is a valid, in-window segment.

2. The payload of the TCP segment, parsed as a framing PDU, has a length field which equals the TCP segment length minus 8, and a key field which matches the expected key for the framing protocol connection.

The framing protocol passes the contained ULP PDUs to a ULP parser. The ULP parser performs direct placement for the PDUs. The ULP parser MUST NOT execute the ULP protocol (i.e. none of the ULP protocol state variables change), until all preceding octets in the TCP stream have also been received.

## 6. Marker Mode

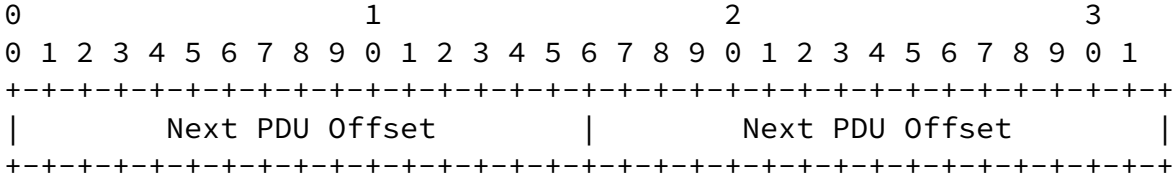
The framing protocol in marker mode inserts framing markers in the TCP octet stream at a period agreed upon by the framing protocol sender and receiver. Each framing marker points to the next PDU in the TCP octet stream. Marker insertion in the TCP octet stream is not synchronized in any way with the ULP. The ULP may use PDUs of





any size up to 2^16-8-(4 * # of markers inserted) (determined by marker interval). Markers will be inserted in the resulting octet stream, possibly interrupting PDUs, as necessary to maintain the interval. Although the placement of each marker is not a function of the ULP PDU boundaries, the contents of each marker are.

The format of a framing marker is as follows:




The "Next PDU Offset" contains the offset to the next PDU, in octets, from the end of the marker.

The "Next PDU Offset" occurs twice in the marker to guarantee that when a marker is split across TCP segments, a complete copy of Next PDU Offset occurs in at least one of the two TCP segments.

The framing protocol receiver must remove (or otherwise ignore) the periodic markers in the received TCP octet stream to reconstruct the PDUs from the sender.

The first marker SHALL be sent in the TCP octet stream preceding any framed PDUs. This first marker will, necessarily, have a Next PDU Pointer of 0. The first marker corresponds to the point in the TCP octet stream when the framing protocol is enabled.

# 7. Security Considerations

## 7.1. Security Protocol Interactions

The ULP framing protocol may be layered on top of IPSec, or TLS. A direct placement network interface which supports connections secured with IPSec or TLS must directly implement security protocol processing as well as framing and direct placement support.





## 7.2. Using IPSec With The Framing Protocol

Since IPSec is designed to secure arbitrary IP packet streams, including streams where packets are lost, the framing protocol could run cleanly on top of IPSec without any change.

Using IPSec end-to-end with the framing protocol in PDU alignment mode permits an optimization to the framing protocol. Because IPSec validation criteria guarantee that IP packets received are equivalent to the IP packets sent, it is not possible for an

intermediary to resegment the TCP stream. If IP fragmentation (rather than resegmenting) is used to send committed data when the EMSS changes, the framing PDU validation header is not needed. In this case, a ULP may run directly on top of a framing-aware TCP.

## 7.3. Using TLS With The Framing Protocol

Using TLS with the framing protocol is more complicated than using IPSec. The combination of TLS and the framing protocol must still provide a modest bound on reassembly buffer size to be useful.

TLS is a record-oriented protocol. TLS records are PDUs just like those used by ULPs that permit direct placement. As with other ULPs, the only way to avoid a complete reassembly buffer is to be able to find TLS PDUs in the presence of lost TCP segments. Therefore, to permit direct placement of ULPs secured with TLS, TLS should also be treated as a protocol which uses framing support.

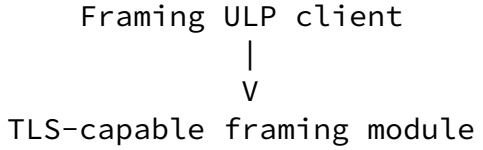
Using the framing protocol with TLS requires modification of a TLS implementation for the combination to perform effectively. Essentially, a TLS implementation must become a client of the framing protocol.

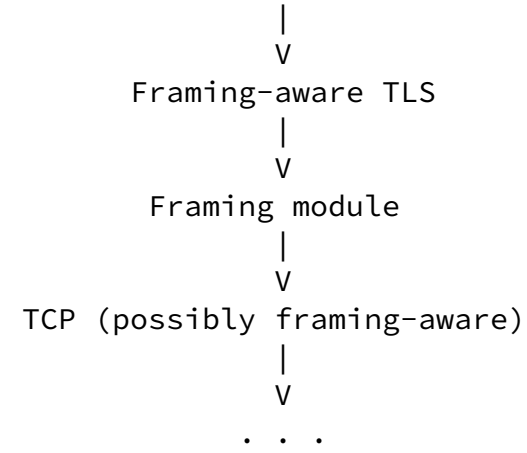
TLS provides a similar interface to TCP for sending protocol data. Protocol data submitted to the TLS send interface may be coalesced with other protocol data in a single TLS PDU, or it may be segmented arbitrarily across more than one TLS PDU. For the framing protocol in to properly support direct placement with TLS, a framing-aware TLS MUST provide a framing-aware interface to the ULP similar to the one described in Appendix A.

This layering looks like:










Although some framing information may be exposed in the clear when running TLS on the framing protocol, this information does not add to what is already available to an attacker. Framing only conveys the location of TLS PDUs, which are already available in the clear.

Unfortunately, ciphers defined for use with TLS do not offer the same independence of TLS PDUs that IPSec provides for IP datagrams. For one thing, TLS supports the use of stream ciphers, which IPSec does not. Stream ciphers typically have dependencies reaching far back in the data stream for deciphering at the current point. Therefore it is probably not appropriate to negotiate the use of a stream cipher when securing the framing protocol.

Block ciphers defined for use with TLS have similar properties to those defined for use with IPSec. Specifically, they all operate in Cipher Block Chaining (CBC) mode. However, while IPSec provides a CBC initialization vector for each IP datagram, TLS defines only a single CBC initialization vector for use in the first block. All subsequent blocks use the cipher-text of their predecessor. To decipher the current TLS PDU, the final cipher-text block from the previous TLS PDU must be available. Typically, block ciphers defined for use with TLS have an 8-octet block size. This implies that for ULP direct placement to be possible with TLS, data from a preceding TCP segment may be needed, where it is not when using the framing protocol without TLS. Note that if the preceding TCP segment is missing, all cipher blocks within the current TCP segment may still be processed except the first one (assuming the bounds of the TLS PDU is known).

### 7.3.1. Using TLS In PDU Alignment Mode

To run the framing protocol running on TLS in PDU alignment mode, an integral number of TLS PDUs may be sent in each TCP segment the same way ULP PDUs are sent in the absence of TLS. A framing-aware TLS would use the framing-aware TCP. In this case, the role of the framing PDU header in detecting unexpected modification of TCP segmentation is subsumed by the strong integrity checks performed on TLS PDUs. There is no need to encapsulate TLS PDUs in a framing PDU. In fact, the vulnerability of the framing key to active attack is eliminated by using TLS validation algorithms instead.

Use of a non-null TLS compression algorithm may interact badly with a framing-aware TLS implementation. A TLS compression algorithm is allowed to increase content length by up to 1024, which may result in the compressed TLS PDU no longer fitting within EMSS. Therefore, only TLS compression algorithms which are known not to increase content length, or increase content length by a small, manageable amount, should be selected.

The need to receive the previous TCP segment before completing TLS processing of current TCP segment means that using the framing protocol in PDU alignment mode with TLS will require some high-speed receive packet buffer memory. This defeats one of the primary advantages of PDU alignment mode. Therefore, while it is possible to use TLS to secure the framing protocol in PDU alignment mode, IPSec would be a more appropriate choice for securing PDU alignment mode connections because it does not require any reassembly buffer memory.

### 7.3.2. Using TLS In Marker Mode

To use TLS on a framing protocol connection in marker mode, the TCP stream must actually contain two, independent sets of periodic markers. Clear-text markers in the TLS PDU stream will permit TLS PDUs to be found in the presence of lost TCP segments. Once a portion of the original, clear-text TCP stream is recovered by TLS processing, markers in the original octet stream are used to find ULP PDUs and perform direct placement.

## 7.4. Other Security Considerations

The modification of the sender's TCP segmentation algorithm in PDU alignment mode does not open any new attacks, since: 1) the segmentation algorithm is not based on input from the network, 2) the segmentation algorithm may pack small ULP PDUs into a single TCP segment so it does not open packet flooding attacks.

If an attacker can send an in-window TCP segment that is accepted, on an unsecured framing protocol connection the attacker can probably force the TCP receiver in to a framing protocol exception path, degrading service. However, such an attacker can also place arbitrary data into the stream, so merely forcing the receiver on to an exception path is not a compelling attack.

## 8. IANA Considerations

If framing is enabled a priori for a ULP by connecting to a well-known port, this well-known port would be registered for the framed ULP with IANA.

## 9. References

[ALF]
D. D. Clark and D. L. Tennenhouse, "Architectural considerations for a new generation of protocols," in SIGCOMM Symposium on Communications Architectures and Protocols , (Philadelphia, Pennsylvania), pp. 200--208, IEEE, Sept. 1990. Computer Communications Review, Vol. 20(4), Sept. 1990.

[SOCKS]
Leech, M., and others, "SOCKS Protocol Version 5," RFC 1928, April 1996

[RFC0879]
Postel, J., "TCP Maximum Segment Size And Related Topics", RFC

879, November 1983

[RFC1112]
Braden, R., ed., "Requirements for Internet Hosts -- Communications Layers", RFC 1122, October 1989





[PathMTU]
Mogul, J., and Deering, S., "Path MTU Discovery", RFC 1191, November 1990

[RFC1750]
Eastlake, D., Crocker, S., Schiller., J., "Randomness Recommendations for Security.", RFC 1750, December 1994

[RFC2581]
Allman, M. and others, "TCP Congestion Control," RFC 2581, April 1999

[Stevens]
Stevens, W. Richard, "Unix Network Programming Volume 1," Prentice Hall, 1998, ISBN 0-13-490012-X

[TCP]
Postel, J., "Transmission Control Protocol - DARPA Internet Program Protocol Specification", RFC 793, September 1981

[TLS]
Dierks, T. and others, "The TLS Protocol, Version 1.0", RFC 2246

[Satran]
Satran, J., "iSCSI - fragments, packets synchronization and RDMA", http://www.haifa.il.ibm.com/satran/ips/iSCSI-RDMA-

memo.txt, July 2000.

Authors' Addresses





Stephen Bailey
Sandburst Corporation
600 Federal Street
Andover, MA 01810
USA

Phone: +1 978 689 1614
Email: steph@sandburst.com

Jim Pinkerton
Microsoft, Inc.
1 Microsoft Way
Redmond, WA 98052
USA

EMail: jpink@microsoft.com

Constantine Sapuntzakis
Cisco Systems
170 W Tasman Drive
San Jose, CA 95134
USA

Phone: +1 408 525 5497

EMail: csapuntz@cisco.com

Matt Wakeley
Agilent Technologies
1101 Creekside Ridge Drive
Suite 100, M/S RH21
Roseville, CA 95661
USA

Phone: +1 916 788 5670
EMail: matt_wakeley@agilent.com





Jim Wendt
Hewlett Packard Corporation
8000 Foothills Boulevard MS 5668
Roseville, CA 95747-5668
USA

Phone: +1 916 785 5198
EMail: jim_wendt@hp.com

Jim Williams
Emulex Corporation
580 Main Street
Bolton, MA 01740
US

Phone: +1 978 779 7224
EMail: jim.williams@emulex.com

## Appendix A. Sockets Support For The Framing Protocol

The sockets support for the framing module takes the form of a set of socket options which may be set or requested to enable the appropriate behavior.

A socket may be in one of three modes in the send direction:

1. Framing-aware TCP mode. No data is added to the TCP octet stream (neither framing PDUs nor markers), but each data buffer presented in a sending operation is sent atomically as a single TCP segment. This mode provides direct access to a framing-aware TCP sender for purposes such as implementing a framing-aware TLS.

2. Framing protocol PDU alignment sender mode. A framing PDU header is added to data presented by an integral number of sending operations, and the resulting framing PDU is sent according to the rules of PDU alignment mode.

3. Framing protocol marker sender mode. Markers are inserted at fixed intervals which point to the octet past the current PDU submitted by a sending operation.

A socket may be in one of two modes in the receive direction:





1. Framing protocol PDU alignment receiver mode. Framing PDUs are expected in each TCP segment.

2. Framing protocol marker receiver mode. Markers are expected at a fixed interval in the TCP stream.

Received TCP segments are processed as defined above. If a socket receiving operation is used to retrieve received data (as opposed to direct placement), framing PDU headers or markers are removed before the data is returned.

## A.1 Enabling The Framing Protocol

```
/* Pick one sending mode and one receiving mode */
if (sendMode == ATOMIC)
  mode = TCP_FRAMING_SEND_ATOMIC
else if (sendMode == ALIGN)
  mode = TCP_FRAMING_SEND_ALIGN;
else /* sendMode == MARKERS */
  mode = TCP_FRAMING_SEND_MARKERS;

if (recvMode == ALIGN)
  mode |= TCP_FRAMING_RECV_ALIGN;
else /* recvMode == MARKERS */
  mode |= TCP_FRAMING_RECV_MARKERS;

setsockopt (s, SOL_TCP, TCP_FRAMING_MODE, &mode,
            sizeof(mode));
```

A framing module that does not support a requested mode MUST fail the setsockopt call. Framing may be enabled on a socket before or after it is connected, subject to the requirements of Section 2.

## A.2 Sending Data Atomically

The standard socket sending operations, including send(), sendto(), sendmsg(), writev(), and others are used to send framed data units (ULP PDU)s with the framing protocol. The EMSGSIZE error should be returned if the buffer passed to the sending operation does not satisfied the size requirements defined in the `ULP Support For Framing' section above.





When the path EMSS increases, the TCP MAY return EMSGSIZE once to inform the client of the change.

## A.3 Retrieving The Current EMSS

```
getsockopt (s, SOL_TCP, TCP_SEND_EMSS, &emss, sizeof(emss));
```

This call returns the maximum segment size that can be submitted in a sending operation without fragmentation. The number returned depends upon the current socket sending mode. If the socket is in framing protocol PDU alignment mode, the returned EMSS is appropriately adjusted by the size of the framing header. The number should not count any octets that go towards TCP options. A framing protocol implementation which does not support PDU alignment mode, because the underlying TCP sender is not framing-aware, is not required to implement this getsockopt call.

## A.4 Disabling ULP PDU Packing

```
flag = 0;
setsockopt (s, SOL_TCP, TCP_FRAMING_PACK_PDUS, &flag,
            sizeof(flag));
```

This call disables the framing protocol in PDU alignment mode from packing more than one ULP PDU into a framing PDU. By default, ULP PDU packing is enabled.

## A.5 Enabling Emergency Mode

```
flag = 1;
setsockopt (s, SOL_TCP, TCP_FRAMING_EMERGENCY, &flag,
            sizeof(flag));
```

This call enables emergency mode for PDU alignment mode. It may be called at any time on a socket, whether connected or not, and whether the current EMSS is smaller than 512 octets or not. By

default emergency mode is disabled.

### A.6 Setting The Sending Marker Interval

```
ivl = 2048;
setsockopt (s, SOL_TCP, TCP_FRAMING_SEND_INTERVAL, &ivl,
            sizeof(ivl));
```

This call sets the period at which markers will be introduced to the sent TCP octet stream. The sending marker interval may be set at any time, but it only has effect when sending markers is enabled for the socket.

### A.7 Setting The Receiving Marker Interval

```
ivl = 2048;
setsockopt (s, SOL_TCP, TCP_FRAMING_RECV_INTERVAL, &ivl
            sizeof(ivl));
```

This call sets the period at which markers are expected in the received TCP octet stream. The receiving marker interval may be set at any time, but it only has effect when receiving markers is enabled for the socket.

## Full Copyright Statement