TCP ULP Message Framing
iSCSI Framing

Randy Haagens
Allyn Romanow

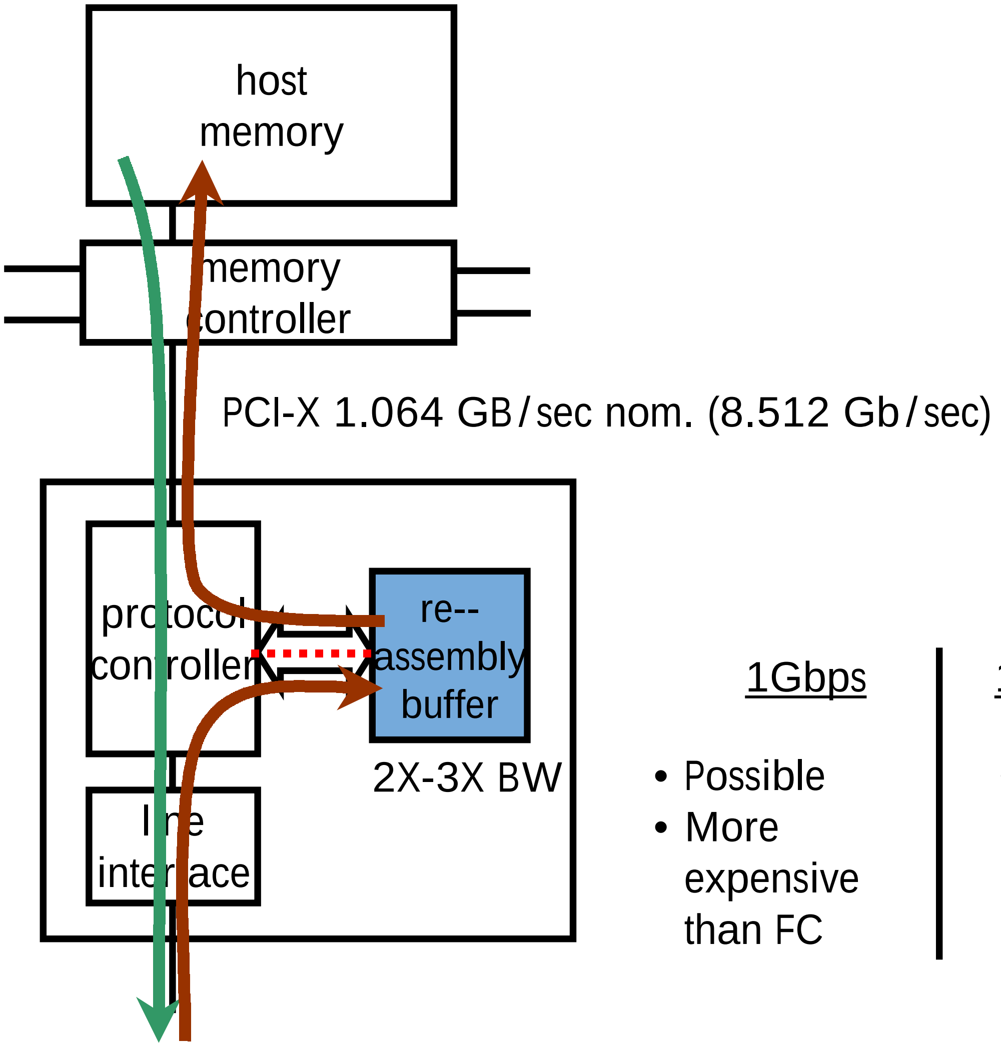
Outline

- The Problem -- Conserving host memory bandwidth
- How to solve it -- Direct data placement
- The framing issue
- Solutions to framing -- pros and cons
  - TCP unaware
  - TCP aware
- TCP message boundary option

The Problem: Cost, Feasibility of Re-assembly

- Limited host memory and bus bandwidth
  - PCI-X delivers approx. 8.5 Gbps
- Must deliver data directly to host memory buffers
  - One use of bus and memory
  - “Zero copy”
- Re-assembly buffer required on NIC to reorder TCP segments received out of order
  - 1 Gbps, possible, expensive (Fibrechannel)
  - 10 Gbps, it does not look feasible

“Conventional” NIC Implementation



host memory

memory controller

PCI-X 1.064 GB/sec nom. (8.512 Gb/sec)

protocol controller

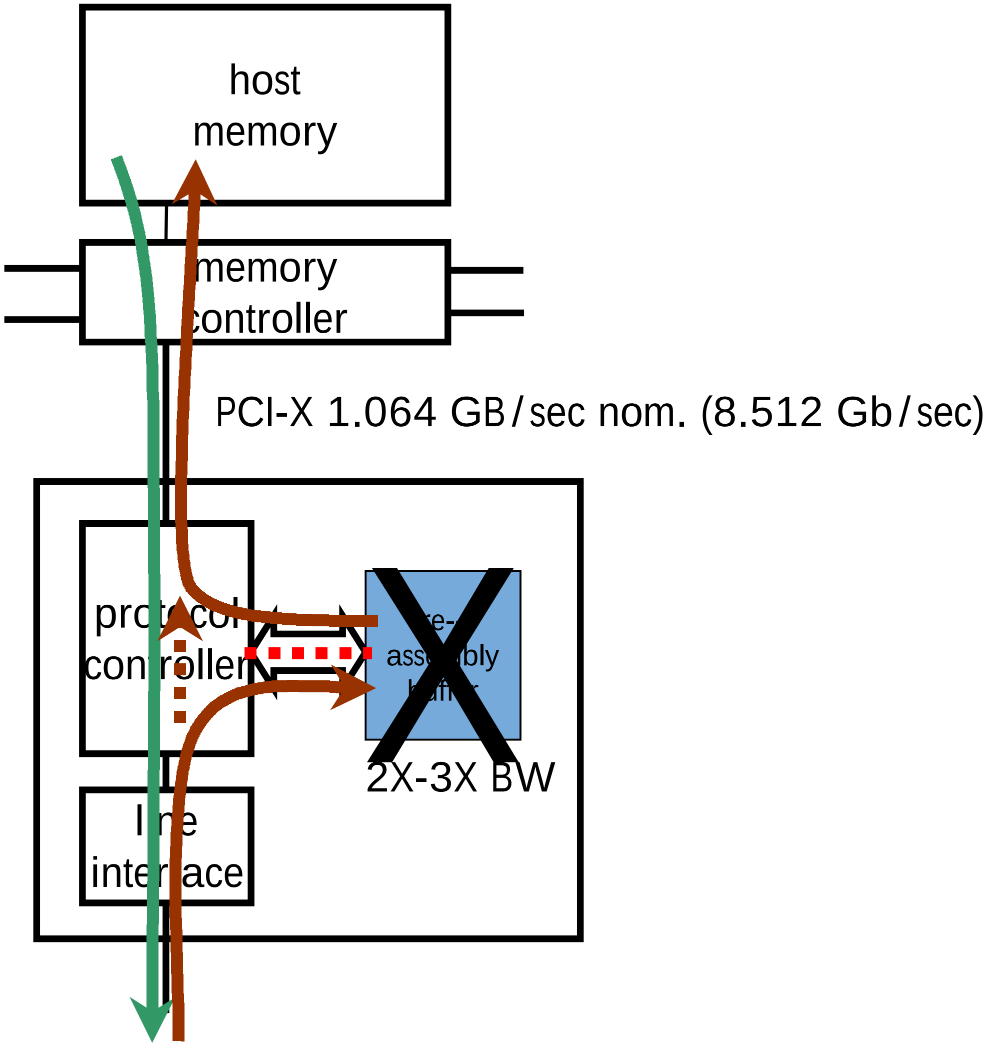
re--assembly buffer

2X-3X BW

line interface

1Gbps	10 Gbps
• Possible • More expensive than FC	• Questionable feasibility

Desired NIC Implementation



host memory

memory controller

PCI-X 1.064 GB/sec nom. (8.512 Gb/sec)

protocol controller

re-assembly buffer

2X-3X BW

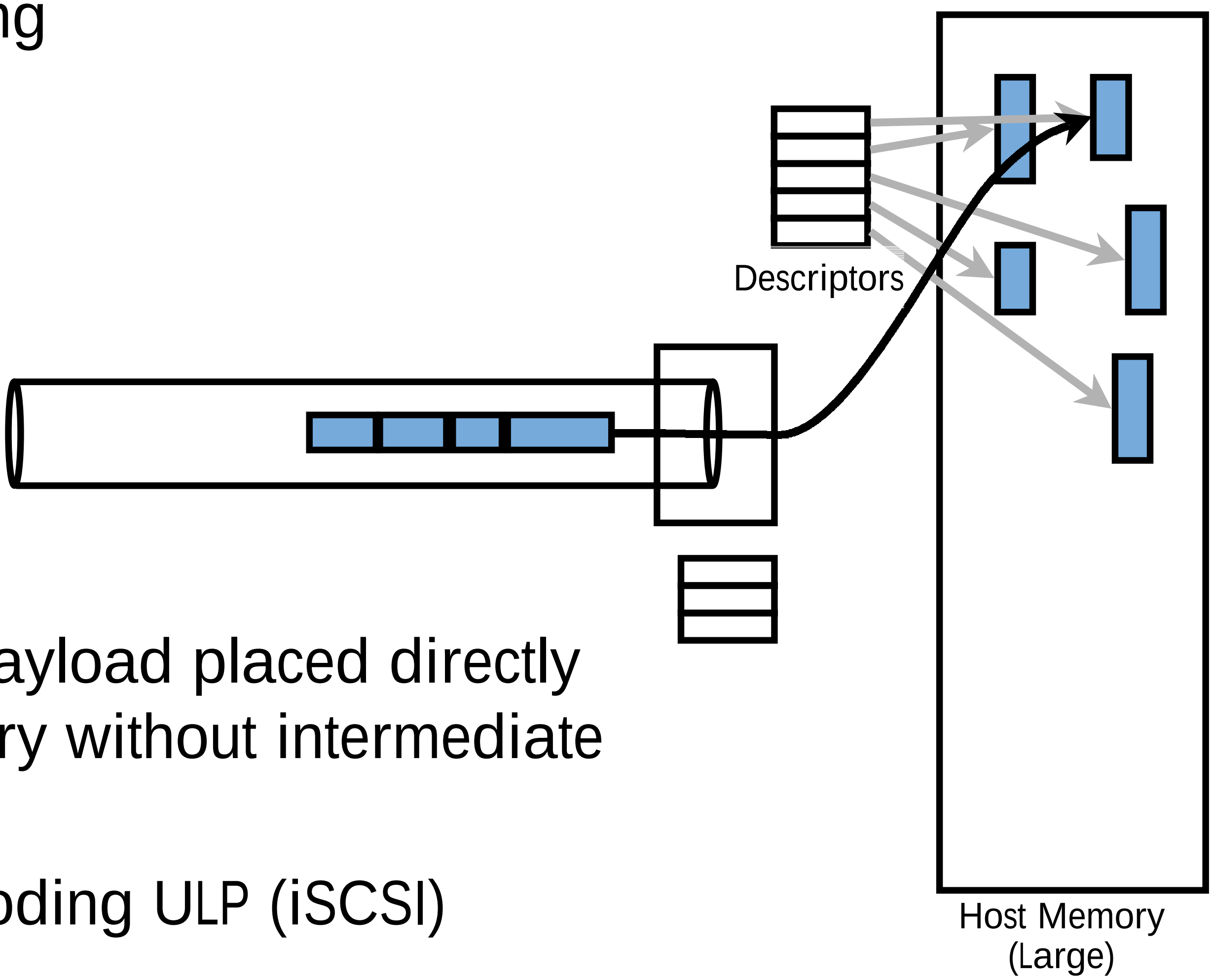
line interface

The Solution: Direct Data Placement

Payload steering

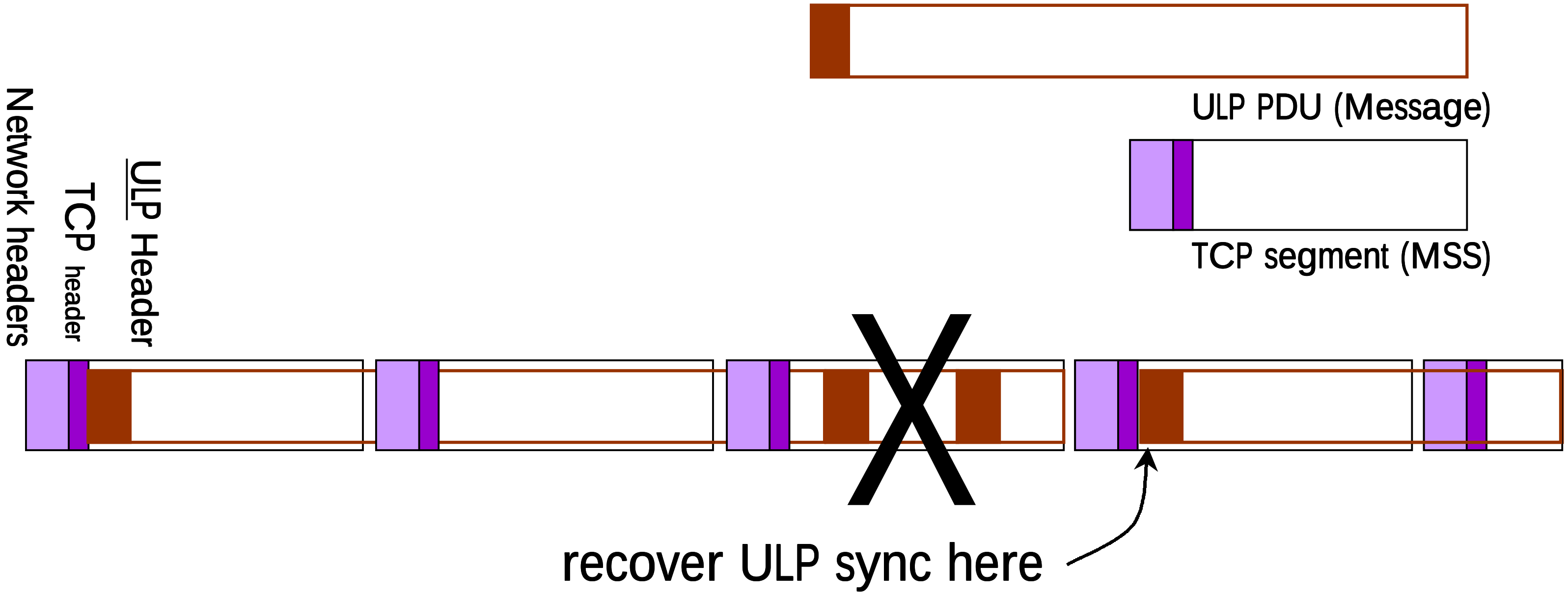
Data steering

RDMA



- ULP (iSCSI) payload placed directly in host memory without intermediate buffer
- Requires decoding ULP (iSCSI) headers

Loss of ULP Synchronization



- Segment containing a ULP header is dropped (or delayed), ULP sync is lost. Direct data placement cannot continue; data must be diverted to a re-assembly buffer

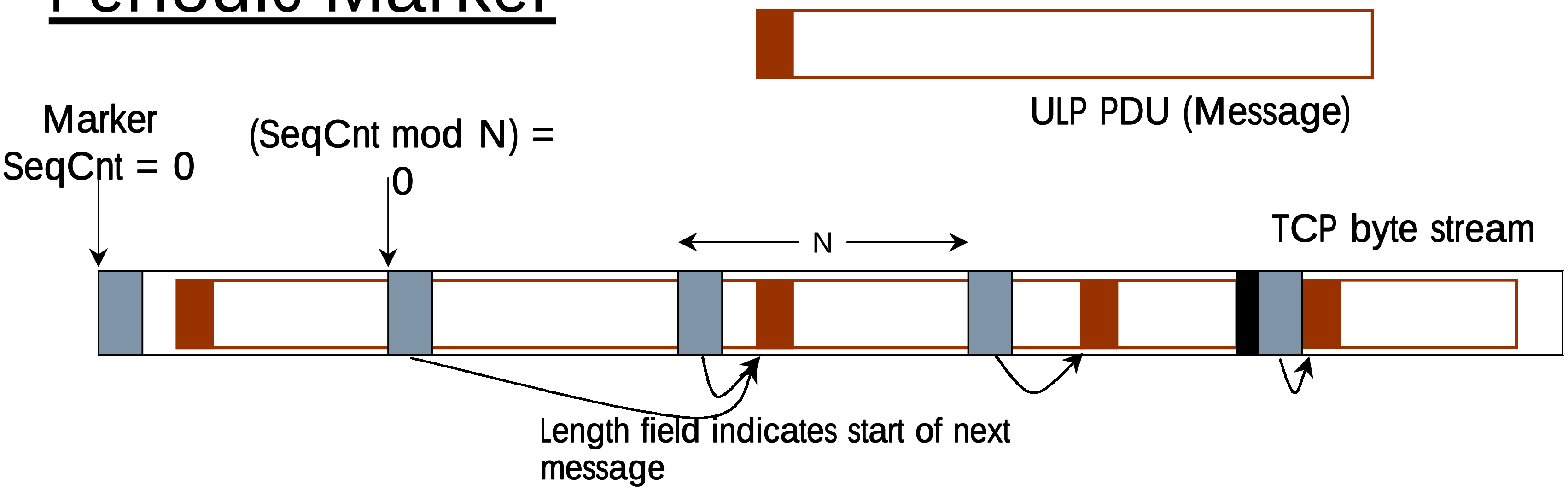
Recovery of ULP Framing Synchronization

- Goal is to recover ULP synchronization at the next ULP header
- Two categories of approaches
  - TCP unaware, transparent to TCP
  - TCP aware, not transparent to TCP

Approaches that are Transparent to TCP

- SCTP
  - Framing with chunks, but requires maturity
- Special characters: byte stuffing, recoding
  - Onerous for software, process stream byte by byte
- Fixed-length ULP messages
  - Inefficient for short ULP messages
- Periodic Marker - Best in class
  - Manageable software overhead to insert and remove markers; relatively easy to implement in hardware

Periodic Marker



- Marker 4B field--number of ULP bytes remaining in current PDU. Marker inserted and removed by framing protocol, e.g. iSCSI
- After loss of synch, locate next marker; use to locate the next ULP PDU
- Markers are transmitted twice in a row. Ensures markers can’t be split by stream segmentation

Approaches that are not Transparent to TCP

- URGent pointer
  - Not allowed -- IESG, not within spec
- PSH bit
  - Use of PSH as a record marker is not allowed (RFC 1122)
- TCP option for finding ULP message boundary

Message Boundary Recognition at Transport Layer

- Procedure
- TCP options - background
- TCP message boundary option for finding ULP framing

Message Boundary Support at Transport Layer

- General solution at transport layer vs. individualized solutions for different applications
- Procedure for standardizing a TCP option
  - TSV working group -- work item
    - Meeting
    - Proposal, mailing list
    - Time frame
- IPS -- follow the TSV WG progress

TCP Options - Overview

- Extend TCP
- Up to 40 bytes, before TCP payload
- Current TCP options
  - MSS Maximum Segment Size
  - Window Scale Factor
  - Timestamp
  - SACK Selective Acknowledgment
- Reference - Stevens

TCP Options - Overview - Issues

- Built-in mechanism for extension of TCP
- Limited space for options, 40 bytes, scarce resource
- Tension between TCP evolution and risk of changes, tend to minimize changes

TCP Message Boundary Option

- Two options
  - One for negotiating the option
  - The second for communicating the message boundary information

Kind = ?	Len = 2

- Message Boundary Permitted Option
  - Sent with the TCP SYN packets

Message Boundary Option

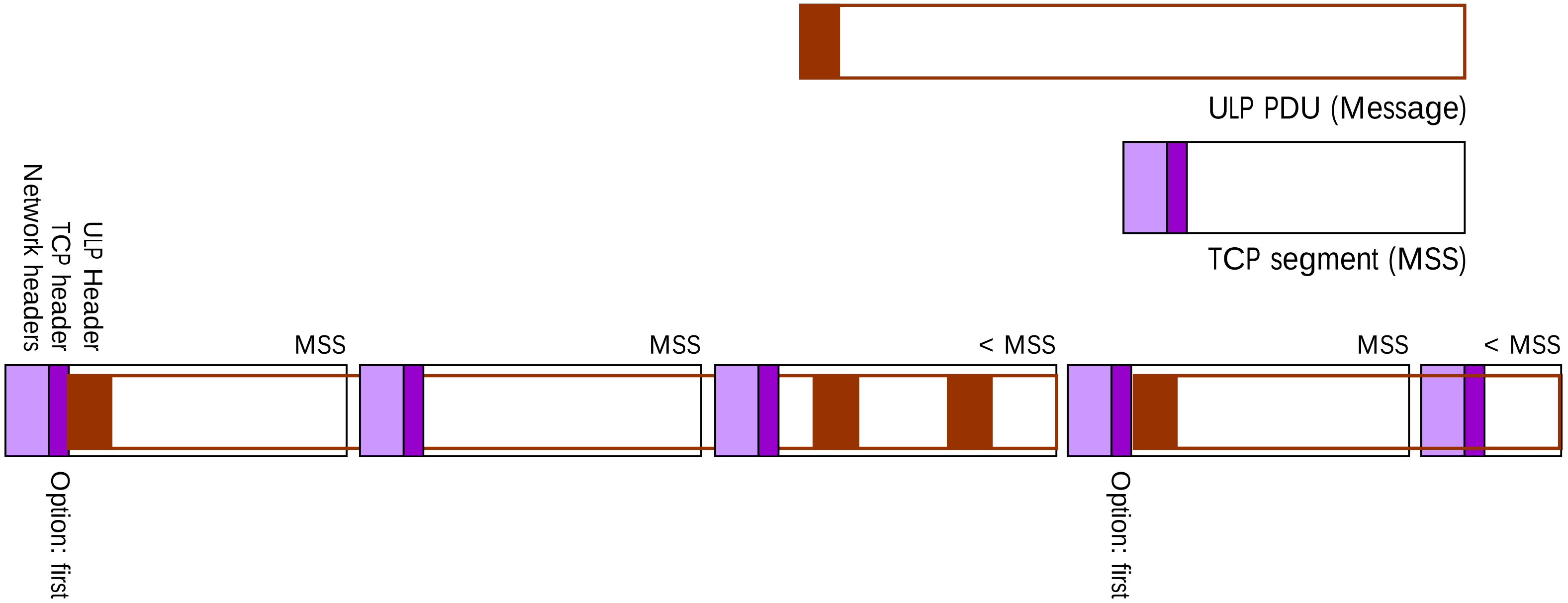
- Two approaches so far -- flag, offset

Flag Approach

Kind = ?	Len = 2

- Costa has written up a description of this alternative
- ULP header is aligned with first byte of TCP segment payload
- May cause segments smaller than an MSS

TCP Framing Option (a) Flag

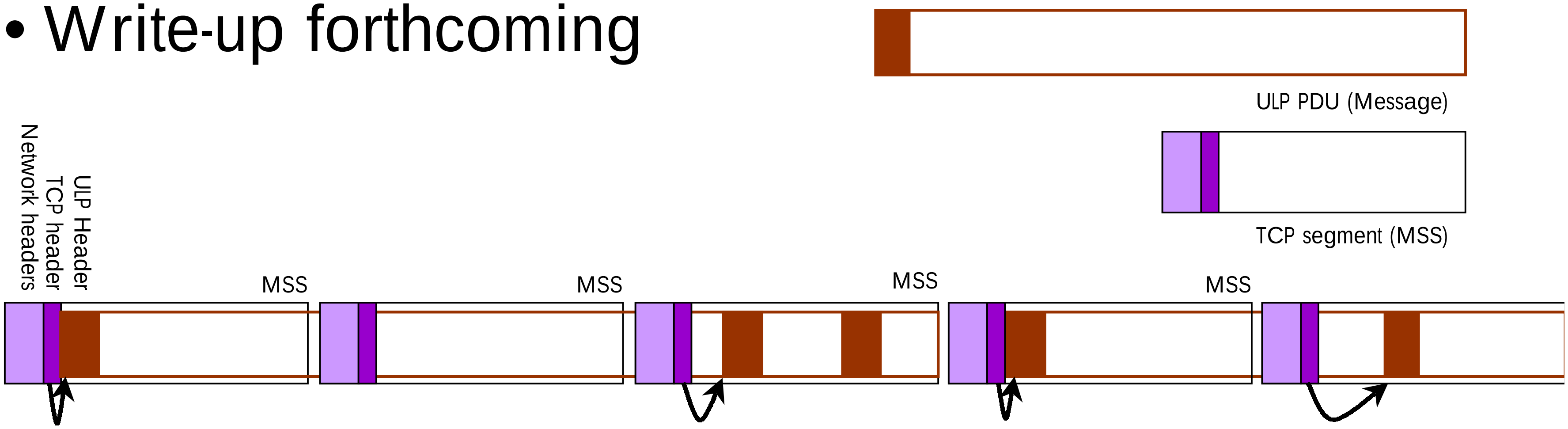


Message Boundary Option

Offset approach

Kind = ?	Len = 4	offset	offset

- 2-byte field indicates offset into TCP payload of first ULP header in the segment
- Write-up forthcoming



Conclusion

- The right way to solve the problem is at the TCP layer
- We’re going to try it
- Let’s see how it plays

- DISCUSSION