MPA (Marker PDU Aligned Framing for TCP)

draft-culley-iwarp-mpa-01

Paul R. Culley

HP

11-18-2002

Marker (Protocol Data Unit) Aligned Framing, or MPA.

Motivation for MPA/DDP

- Enable "Direct Data Placement" (DDP) to Reduce Copy overhead for Network traffic
  - Each time data must be copied, a 10gigabit stream will incur at least an extra 2 Gigabytes (20 gigabits) of memory bandwidth someplace in the system.
- Reduce Host CPU overhead for Network traffic
  - Important overall, but not a serious factor in choice of MPA or other methods of frame recovery.
- Enable the above at volume price points
  - If you don't save money, it is not worth doing

11/18/2002 MPA

•We designed MPA to allow the DDP protocol (Direct data placement, draft-shah-iwarp-ddp-01) to work over TCP while maintaining the advantages of DDP. DDP allows the user's data to land exactly where he wants it without copies.

•Copies either steals performance (ultimately a cost overhead), or directly increase costs in our systems.

•The copy bandwidth includes CPU overhead if done by the CPU as in traditional sw stacks.

•If done by the NIC to host memory, it strains I/O and memory busses.

•If done on NIC, but to external RAM, that incurs cost.

•If the cost of doing this gets too high, it will not be adopted, because TCP with SW stack is the competition.

•DDP/MPA can save system overhead of 30-40%?? So we are talking about an extra server for every 3…

Why do MPA?

- DDP (a target ULP for MPA) headers must be identified to allow placement.
  - Segment lengths work for in order arrival.
  - Markers provides a deterministic method for out of order arrival.
- To Avoid Buffering, DDP must get the whole segment for placement at once.
  - So require alignment with TCP segment.
- Need better data integrity.
  - Added CRC

11/18/2002 MPA

•To avoid the copies, DDP needs its headers accurately identified. Not a problem for in order arrival, the segment lengths allow easily finding the next segment.

•For Out of order arrival, a Marker inserted at 512 byte intervals is used to locate the headers in a deterministic way. The receiver can always find a marker by examining the TCP sequence number.

•We decided on the use of Markers to identify the FPDU start because it was fully "deterministic" unlike other mechanisms that search the stream for (or just "catching") a pattern to identify the FPDU start (as in `draft-ietf-tsvwg-tcp-ulp-frame-01.txt). No chance of mis-identifying a user data pattern as the start of an FPDU.`

•If the whole DDP segment arrives at once, you don't have to store a partial segment or header while waiting for the rest, so we specify "Alignment" at the sender. In the usual case, the segments arrive intact.

•And TCP's data integrity is not always as good as we would like, so we upgraded it with MPA to be on a par with SCTP.

Comments on MPA

- "It changes TCP semantics"
  - By requiring TCP segments to be aligned with FPDUs at sender
    - To eliminate copies you have to get the whole FPDU at once
  - By not packing
    - To get the segments out quickly
- Basically we are saying: Turn off Nagle; use TCP_NODELAY

11/18/2002 MPA

•Issues from the reflector; The most vocal commenters claims that we are "Changing TCP semantics" in various ways.

•As mentioned before, In order to avoid copies, we have to identify and receive whole segments. This leads to alignment and no packing.

•This is fairly common practice already, most implementations have a way of disabling the Nagle algorithm, for example the Socket's TCP_NODELAY option. When Applications use records smaller than a TCP segment, this is basically what you get. And MPA allows its ULP, (DDP) to see the maximum segment size, so this works.

•Copy avoidance presumes that nothing interferes with alignment or full FPDU reception. So no "middle boxes" and full MPA/TCP implementations. This is very doable in the data center, and probably quite reasonable for certain applications outside the data center.

•If we don't get full, aligned, segments, then the recievers must still work, but NIC performance will often degrade.

Comments on MPA

- "It changes TCP semantics"
  - By allowing receiver to "do something" with stream data before it can be Acked
- MPA passes the segments to ULP as they arrive and are identified, and provides a mechanism to later establish order.
  - DDP uses this to "place" the data, and later "deliver" notification to the application.
  - When used together DDP and MPA provide for reliable delivery of data.

11/18/2002 MPA

•Another reflector comment is that we allow the receiver to " do something", in this case pass the data to the DDP layer, before TCP has reached the point where it can be Acknowledged. This applies to Out of order segments.

•For one thing, we are not Requiring this ability, we are only enabling it. An implementation is free to buffer and reorder if it wants to, and other implementations can use DDP to place the out of order data immediately.

•An MPA implementation can provide the complete, identified and validated segment to DDP regardless of order. The implementation also provides a mechanism to identify the correct order for the segments; this is usually based on the TCP sequence number, but can be anything.

•When DDP is the ULP, it can use the segments immediately for "placement" (storage into the final buffer), and later can "deliver" the notification to the application that entire messages have arrived and are in order.

•The end result is that MPA and DDP together have provided a reliable delivery of data to the application.

Comment: “MPA is not needed (just buffer the data until it can be delivered).”

Why not buffer until ordered: How much Memory?

- To act as a elasticity buffer: <100kbytes (cheap)
  - Can be reasonably implemented with a dedicated RAM on chip with today's technology
- To support reordering: delay * bandwidth (costly)
  - A single drop for a fully utilized 10 gigabit TCP connection with 1millisec of round trip latency takes 1 megabyte of storage.
  - Lots of connections *might* allow statistical allocation of less memory, but this is worst case
  - 1 ms might be ok in data center, but long latency (50 ms across the country) external connections makes the problem much worse.

11/18/2002 MPA

•Suggestion has been made to just use re-order buffer, “it is not that much more memory than an elasticity buffer”, so here is an example of why se don’t think this is cost effective…

•Really large elasticity buffers don’t help much, the NIC and system must keep up or use a scheme to back off the connection speed. This would typically be some heuristic for reduction of RWIN, or drop packets to invoke congestion control at sender. Small elasticity buffer can be implemented in dedicated on chip memory, so this need not count as a “real” copy for system level bandwidth reduction.

•Some argue that the ONLY way to deal with slower NIC/systems is to use buffers per connection, tied to RWIN, but this needs lots more memory. With MPA, vendors get the choice. The large memory needed needs to be in off chip memory, either host, or dedicated. Either way, you pay…

Comment: “It needs both Header and Data CRCs” (Instead of entire FPDU CRC)

Yes, if designing a “Flow through NIC”

- But Need for elasticity buffer anyway makes “flow through” an un-needed extra complexity.
- But NIC designers want to check L2 CRC, TCP, and IP checksums first anyway, requires whole packet.

- Yes, if “Placing” partial FPDUs (can place 1st part without whole FPDU)
  - As long as alignment is correct, this is not needed.
    - Expected to be the usual case in the data center, at least, and common elsewhere.
    - As an uncommon case, extra wire overhead and extra complexity of saving headers, while placing data doesn't seem worth the small buffer saving

11/18/2002 MPA

Maintaining alignment/framing/DDP is expected to be the common case, will be implemented as “fast path” in NIC. Once this breaks, NICs will probably go to “slow path” processing, using some host buffers for re-assembly. When this happens, performance will probably degrade.