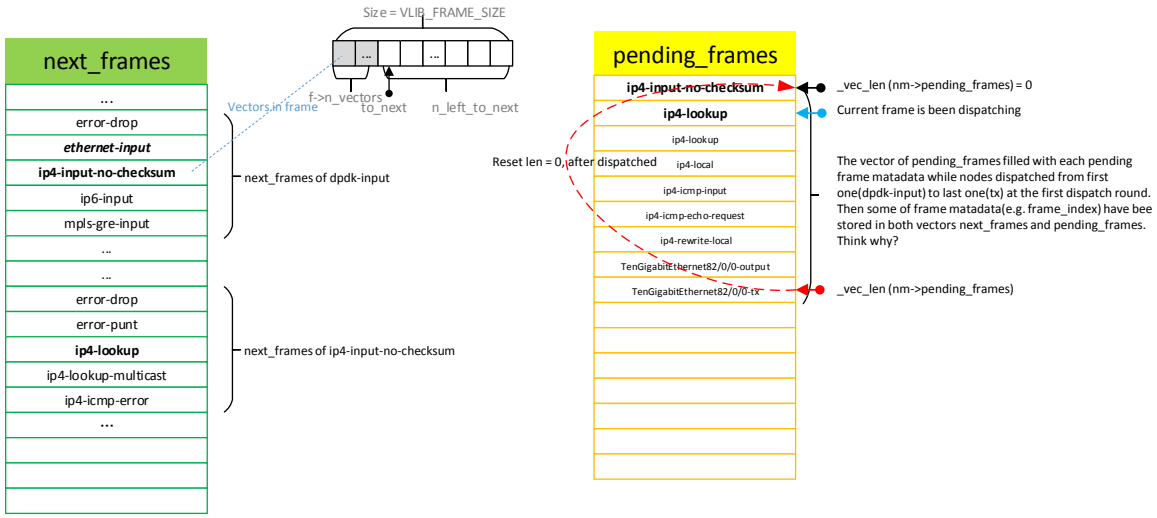
Worker Thread

Michael

2017

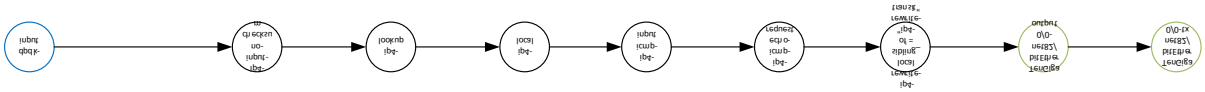
frame, next_frames and pending_frame



Dispatch Input Node

VPP16.09

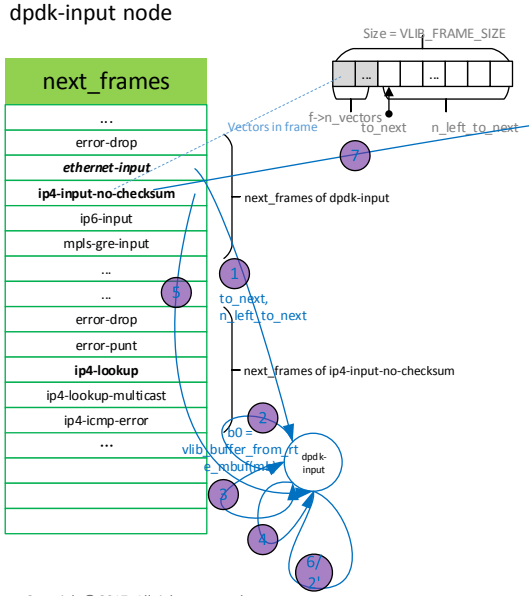
```
vlib_node_runtime_t *n;
vec_foreach (n, nm->nodes_by_type[VLIB_NODE_TYPE_INPUT])
{
   cpu_time_now = dispatch_node (vm, n, VLIB_NODE_TYPE_INPUT,
            VLIB_NODE_STATE_POLLING, /* frame */ 0,
            cpu_time_now);
}
```



Dispatch Input Node(Cont')

VPP16.09

dpdk-input node



1) INPUT node dpdk-input has been dispatched, read all RX queues of NICs assigned to this thread/CPU core
2) ❶ if(n_buffers > 0), that means rte_mbuf received; then translate rte_mbuf to vlib_buffer/vector.
2.1) try to get given next_index of next_frame by vlib_get_next_frame()and return to_next, n_left_to_next
3) ❷ translate rte_mbuf to vlib_buffer and translate buf pointer to buf index

4) ❸ do put vector into frame
```
to_next[0] = bi0;
to_next++;
n_left_to_next--;
```

5) ❹ check related header filed to determine real next_ index of the frame which will be put this vlib_buf into by **dpdk_rx_next_and_error_from_mb_flags_x1( )**
6. vlib_put_next_frame( )
6.1 ❺ if next_index changed(diff as given one before), restore the given frame status, like f->n_vectors – 1
6.2 ❻ get the next_index of real next frame to put into vector(see also to_next, n_left_to_next above); then
6.2.1 cached next_index onto current node r->cached_next_index = next_index;
6.2.2 put next_frame to pending frames
6.2.3 ❼ change the frame flags and next_frame flags to PENDING(pre-check f->n_vectors > 0 && f->flags != PENDING)
nf->flags |= VLIB_FRAME_PENDING;
f->flags |= VLIB_FRAME_PENDING;
Note: if the frame has vectors, the flags should be PENDING already, think why? Pls refer to ❹

Dispatch Pending Frame

VPP16.09

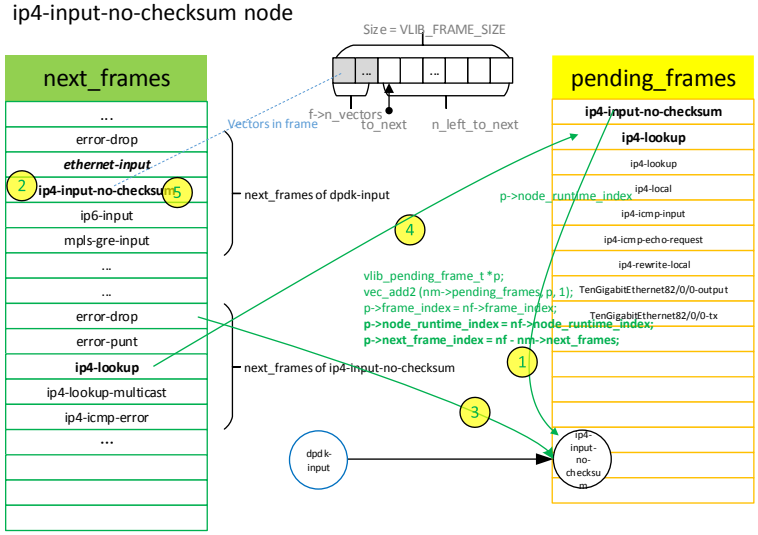
```
vlib_node_runtime_t *n;
vec_foreach (n, nm->nodes_by_type[VLIB_NODE_TYPE_INPUT])
{
   cpu_time_now = dispatch_node (vm, n, VLIB_NODE_TYPE_INPUT,
            VLIB_NODE_STATE_POLLING, /* frame */ 0,
            cpu_time_now);
}
```

pending_frames
ip4-input-no-checksum
ip4-lookup
ip4-lookup
ip4-local
ip4-icmp-input
ip4-icmp-echo-request
ip4-rewrite-local
TenGigabitEthernet82/0/0-output
TenGigabitEthernet82/0/0-tx

Dispatch Pending Frame(Cont')

VPP16.09

ip4-input-no-checksum node

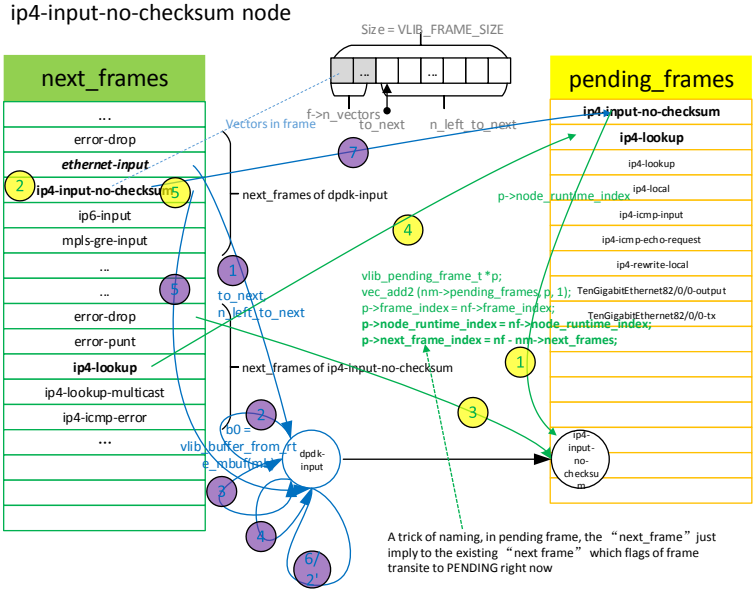


dispatch_pending_node()
1) ❶ get runtime node from pending_frame p->node_runtime_index
2) ❷ set next frame is unusable while current frame is under dispatching in vector next_frames according to pending_frame get from vector pending_frames via p->next_frame_index
```
 nf = vec_elt_at_index (nm->next_frames, p->next_frame_index);
 ...
 restore_frame_index = ~0;
 if (nf->frame_index == p->frame_index)
  {
   nf->frame_index = ~0;
   nf->flags &= ~VLIB_FRAME_IS_ALLOCATED;
   if (!(n->flags &
VLIB_NODE_FLAG_FRAME_NO_FREE_AFTER_DISPATCH))
            restore_frame_index = p->frame_index;
  }
```
3) ❸ dispatching node with frame calling dispatch_node()
```
 f = vlib_get_frame (vm, p->frame_index);
```
 3.1) get next_frame
 3.2) put vectors into next frame
 3.3) ❹ add next frame into pending_frames
4) reset frame
```
 f->flags &= ~VLIB_FRAME_PENDING;
```
❺ /* Frame is ready to be used again, so restore it. */
```
 if (restore_frame_index != ~0)
  {
   /* p->next_frame_index can change during node dispatch if node
      function decides to change graph hook up. */ WHY???!!!
   nf = vec_elt_at_index (nm->next_frames, p->next_frame_index);
   nf->frame_index = restore_frame_index;
   nf->flags |= VLIB_FRAME_IS_ALLOCATED;
  }
```

Dispatch Pending Frame(Cont')

VPP16.09

ip4-input-no-checksum node



⑤ Why?

Why p->next_frame_index(should be nf->frame_index) can change?
When the node running and ready to put vectors to next frame, but find the next frame is being dispatching via next_frame_index in next_frames; so a new frame is allocated by vlib_get_next_frame() -> vlib_get_next_frame_internel()

```
if (PREDICT_FALSE (!(nf->flags & VLIB_FRAME_IS_ALLOCATED)))
 {
  nf->frame_index = vlib_frame_alloc (vm, node, next_index);
  nf->flags |= VLIB_FRAME_IS_ALLOCATED;
 }
```

And nf->frame_index reassigned accordingly; after all vectors/mbufs processed, then this new nf matadata added to pending_frames.the nf->frame_index changed, not p->next_frame_index; so restore old nf->frame_index needed.

In VPP v16.09, it seems the restore is never happen in worker+main threads mode because the nodes dispatched one by one in sequence, but, unless dpdk read pkts > VLIB_FRAME_SIZE (256), not supportted yet; in io+worker threads nodes, io may dispatch more times/frames than worker.

Output&Tx Node

VPP16.09

Output node: <interface name>_output
vnet_interface_output_node_no_flatten()
- vnet_interface_output_node_no_flatten_inline()
-- vlib_get_new_next_frame(), call this if the frame of tx node has some vectors not dispatched, so allocate new frame and reassign nf->frame_index. If IO thread available, then above scenario could be, because IO thread may run slowly; otherwise if only worker threads, the scenario never happen.
If vlib buffer
if ((b->flags & BUFFER_OUTPUT_FEAT_DONE) == 0)
While dispatching vector, check
count_trailing_zeros (next1, vnet_buffer (b1)->output_features.bitmap);
To determine that whether output node need to dispatch the vector to the frame whcih "next1" pointed to. **HOW ABOUT OUTPUT FEATURE WORKS?**
Otherwise output node dispatch vectors to TX frame with next_index = VNET_INTERFACE_OUTPUT_NEXT_TX

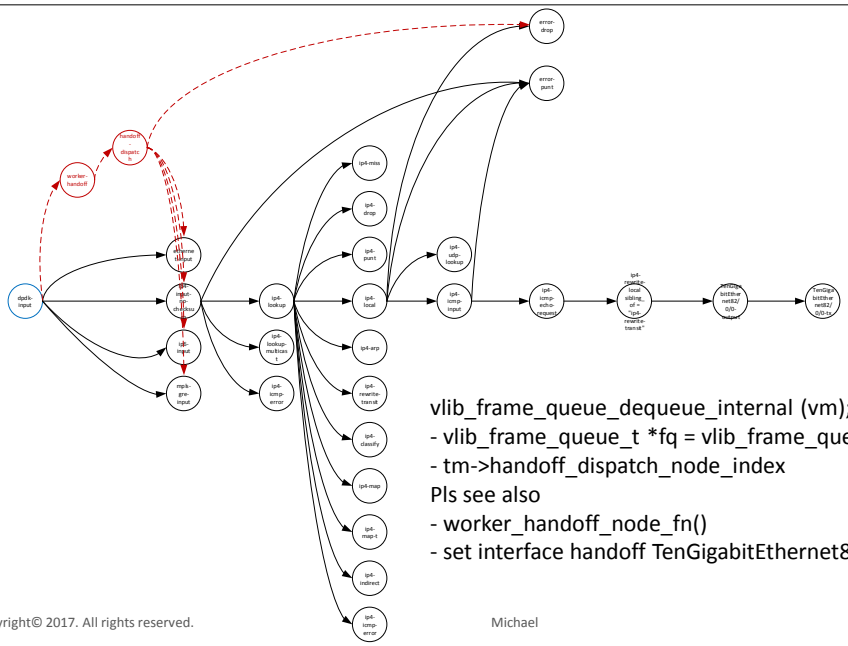
TX node:
dpdk_interface_tx()
- get tx_vector/tx_ring
- translate vlib_buffer to rte_mbuf
- ...
- Transit

Actually each TX queue has its own a pair of <ifname>_output and <ifname>_tx nodes in the same worker threads.
So the previous node of <ifname>_output node should determine output interface in advance. E.g. in node ip4-rewrite-transit
next0 = (error0 == IP4_ERROR_NONE) ? adj0[0].rewrite_header.next_index : next0;

Think why? See also dispatch_process() <- … <- dpdk_process() <- dpdk_lib_init() <- ethernet_register_interface() <- vnet_register_interface()

Worker Handoff Node

VPP16.09



vlib_frame_queue_dequeue_internal (vm);
- vlib_frame_queue_t *fq = vlib_frame_queues[thread_id];
- tm->handoff_dispatch_node_index

Pls see also
- worker_handoff_node_fn()
- set interface handoff TenGigabitEthernet82/0/0 workers 1

Worker Handoff Node(Cont')

VPP16.09