Trace

Michael

2017

Data Structure VPP16.09

```
typedef struct
{
  /* CPU time stamp trace was made. */
  u64 time;

  /* Node which generated this trace. */
  u32 node_index;

  /* Number of data words in this trace. */
  u32 n_data;

  /* Trace data follows. */
  u8 data[0];
} vlib_trace_header_t;
```

Flags

- **VLIB_NODE_FLAG_TRACE**
- **VLIB_BUFFER_IS_TRACED**

CLI VPP16.09

DBGvpp# trace filter include error-drop 1
cli_filter_trace()
DBGvpp# trace add dpdk-input 100
cli_add_trace_buffer()
```
typedef struct
{
  u32 count;
  u32 limit;
} vlib_trace_node_t;
```
DBGvpp# show trace
cli_show_trace_buffer()
- trace_apply_filter() <--- filter applied here
-- filter_accept()

Trace functions:
Vlib_trace_...()

Filter:
trace_apply_filter()
...

Add Trace VPP16.09

```
always_inline void *
vlib_add_trace (vlib_main_t * vm,
                vlib_node_runtime_t * r, vlib_buffer_t * b, u32 n_data_bytes)
{
  vlib_trace_main_t *tm = &vm->trace_main;
  vlib_trace_header_t *h;
  u32 n_data_words;

  vlib_validate_trace (tm, b);

  n_data_bytes = round_pow2 (n_data_bytes, sizeof (h[0]));
  n_data_words = n_data_bytes / sizeof (h[0]);
  vec_add2_aligned (tm->trace_buffer_pool[b->trace_index], h,
                    1 + n_data_words, sizeof (h[0]));

  h->time = vm->cpu_time_last_node_dispatch;
  h->n_data = n_data_words;
  h->node_index = r->node_index;

  return h->data;
}
```

→*tm->trace_buffer_pool[]*

```
vlib_trace_buffer(vlib_main_t * vm,
                  vlib_node_runtime_t * r,
                  u32 next_index, vlib_buffer_t * b, int follow_chain))
{
  ...
  pool_get (tm->trace_buffer_pool, h);

  do
   {
     b->flags |= VLIB_BUFFER_IS_TRACED;
     b->trace_index = h - tm->trace_buffer_pool;
   }
  while (follow_chain && (b = vlib_get_next_buffer (vm, b)));
}
```



```
t0 = vlib_add_trace (vm, node, b0, sizeof (t0[0]));
t0->adj_index = vnet_buffer (b0)->ip.adj_index[which_adj_index];
t0->flow_hash = vnet_buffer (b0)->ip.flow_hash;
t0->fib_index = (vnet_buffer(b0)->sw_if_index[VLIB_TX] != (u32)~0) ?
            vnet_buffer(b0)->sw_if_index[VLIB_TX] :
            vec_elt (im->fib_index_by_sw_if_index,
                     vnet_buffer(b0)->sw_if_index[VLIB_RX]);

clib_memcpy (t0->packet_data,
             vlib_buffer_get_current (b0),
             sizeof (t0->packet_data));
```

Example


```
typedef struct
{
  u32 buffer_index;
  u16 device_index;
  u16 queue_index;
  struct rte_mbuf mb;
  vlib_buffer_t buffer;
  u8 data[256];
} dpdk_rx_dma_trace_t;
```

```
typedef struct {
  u8 packet_data[64];
} ip4_input_trace_t;
```

```
typedef struct {
  u32 adj_index;
  u32 flow_hash;
  u32 fib_index;
  u8 packet_data[64 -
1*sizeof(u32)];
} ip4_forward_next_trace_t;
```

```
15:12:46:260666: dpdk-input
  TenGigabitEthernet82/0/0 rx queue 0
  buffer 0x1b390: current data 14, length 84, free-list 0, totlen-nifb
0, trace 0x1
  PKT MBUF: port 0, nb_segs 1, pkt_len 98
    buf_len 2176, data_len 98, ol_flags 0x0, data_off 128, phys_addr
0x7b587540
    packet_type 0x10
    Packet Types
      RTE_PTYPE_L3_IPV4 (0x0010) IPv4 packet without extension headers
  IP4: 90:e2:ba:84:1c:3a -> 90:e2:ba:84:45:b2
  ICMP: 100.1.0.2 -> 100.1.0.1
    tos 0x00, ttl 64, length 84, checksum 0x43d4
    fragment id 0x2ed0, flags DONT_FRAGMENT
  ICMP echo_request checksum 0x500c

15:12:46:260675: ip4-input-no-checksum
  ICMP: 100.1.0.2 -> 100.1.0.1
    tos 0x00, ttl 64, length 84, checksum 0x43d4
    fragment id 0x2ed0, flags DONT_FRAGMENT
  ICMP echo_request checksum 0x500c

15:12:46:260683: ip4-lookup
  fib 0 adj-idx 4 :  100.1.0.1/24 flow hash: 0x00000000
  ICMP: 100.1.0.2 -> 100.1.0.1
    tos 0x00, ttl 64, length 84, checksum 0x43d4
    fragment id 0x2ed0, flags DONT_FRAGMENT
  ICMP echo_request checksum 0x500c
```