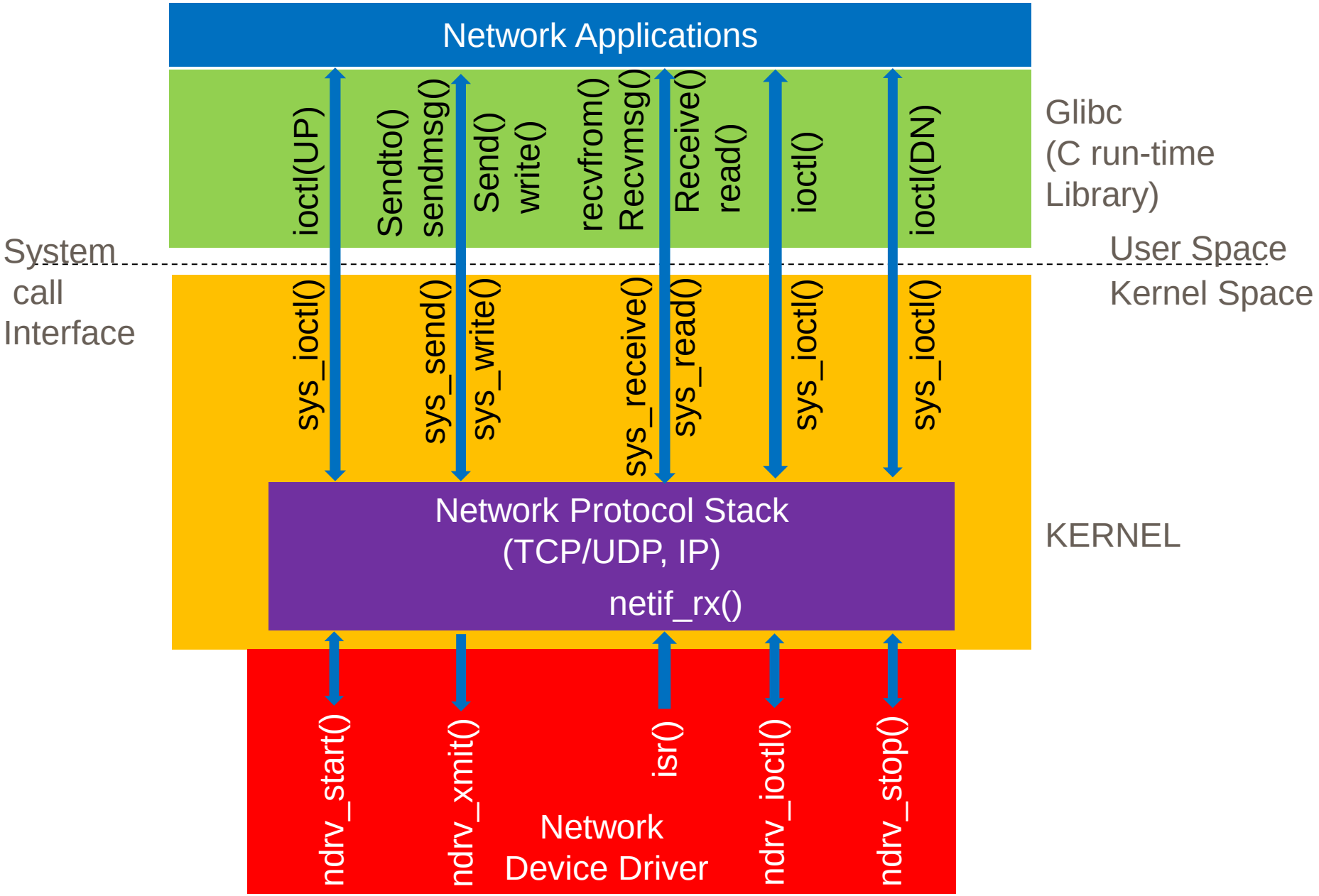
Linux Network Device Driver

Linux Network Device Drivers - 2

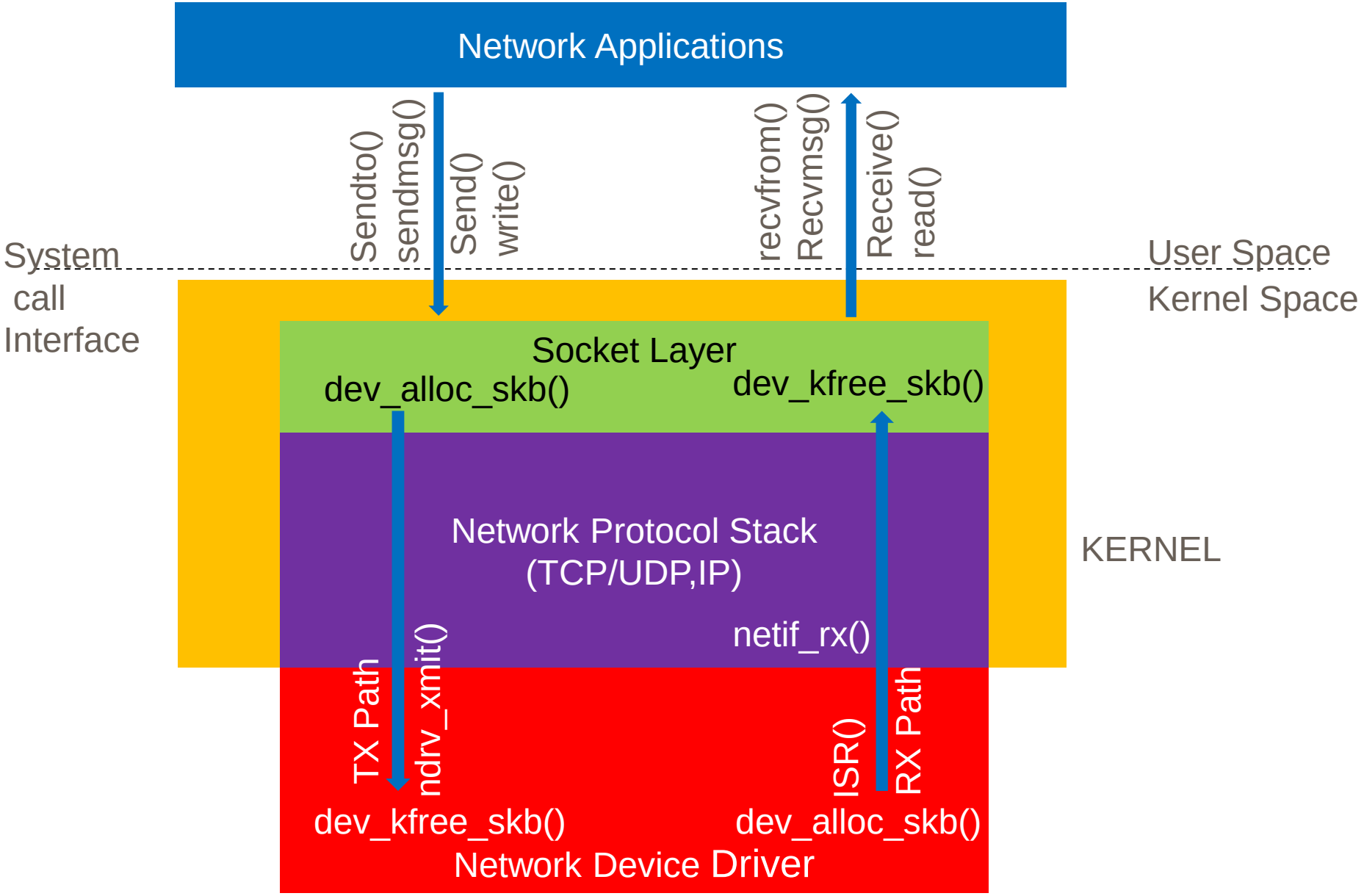
By

Jitesh Verma

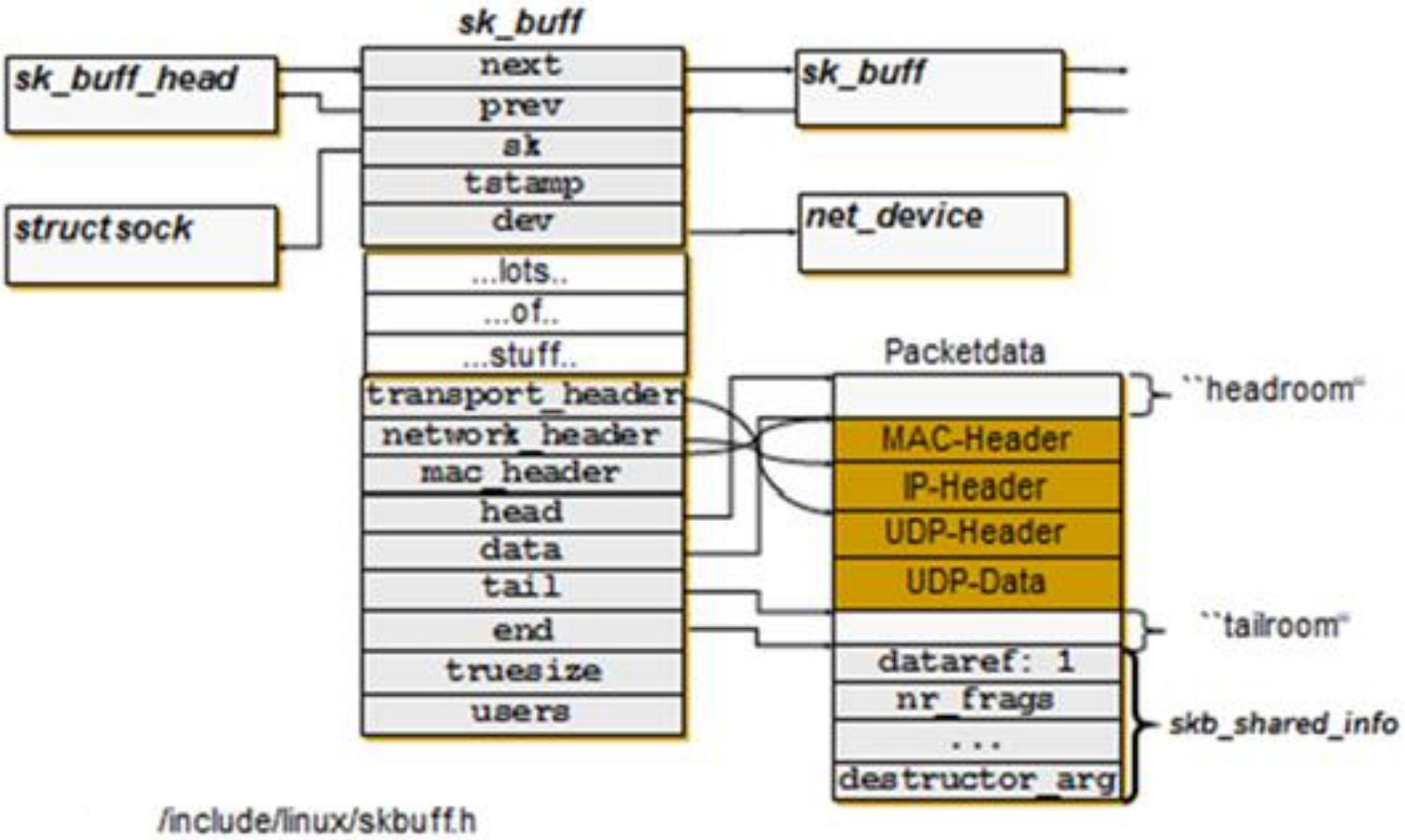
Application – Network Device Driver Interaction



Life-Cycle/Flow of Socket Buffer (SK_BUFF)

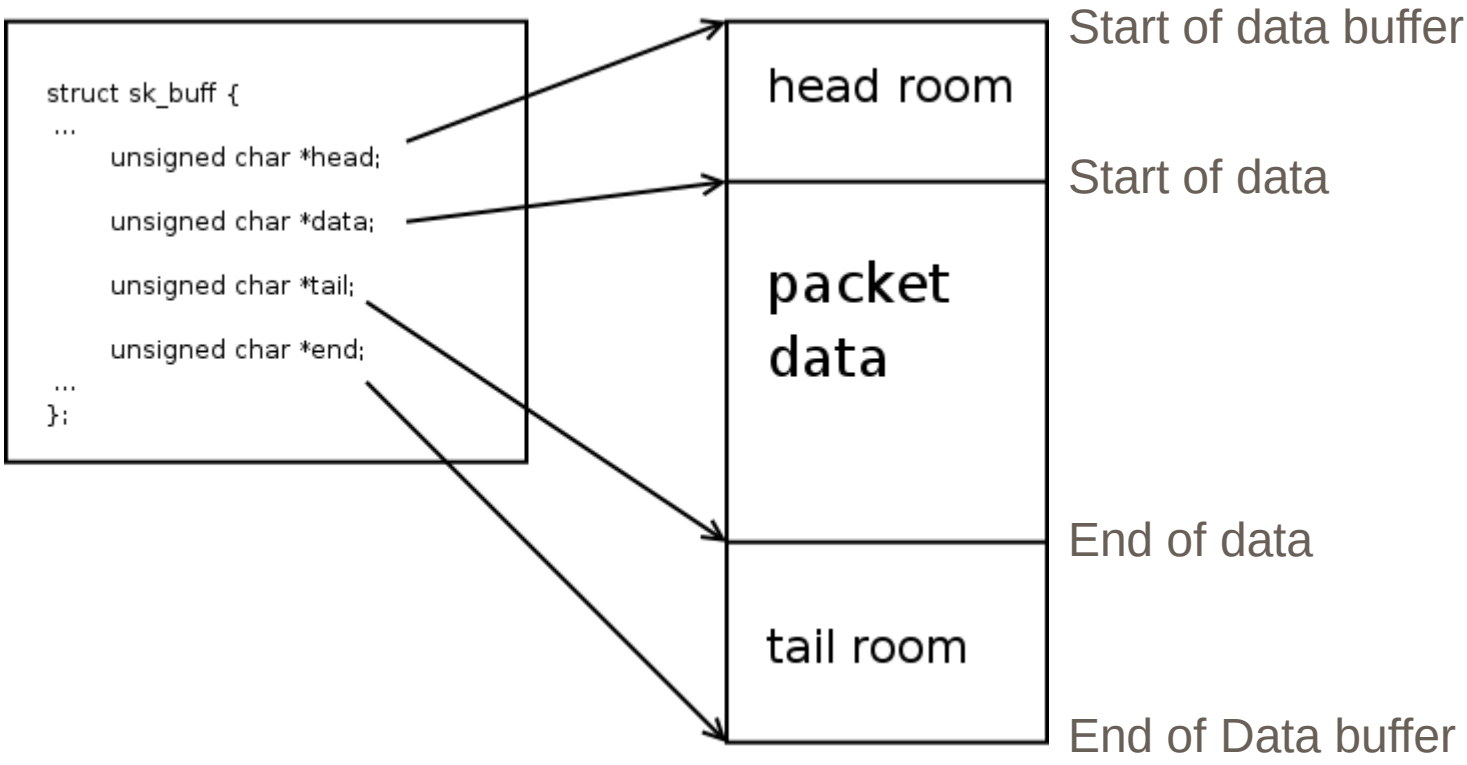


struct sk_buff – Detailed View



struct sk_buff – Simplified View

HEADER FILE: linux/skbuff.h

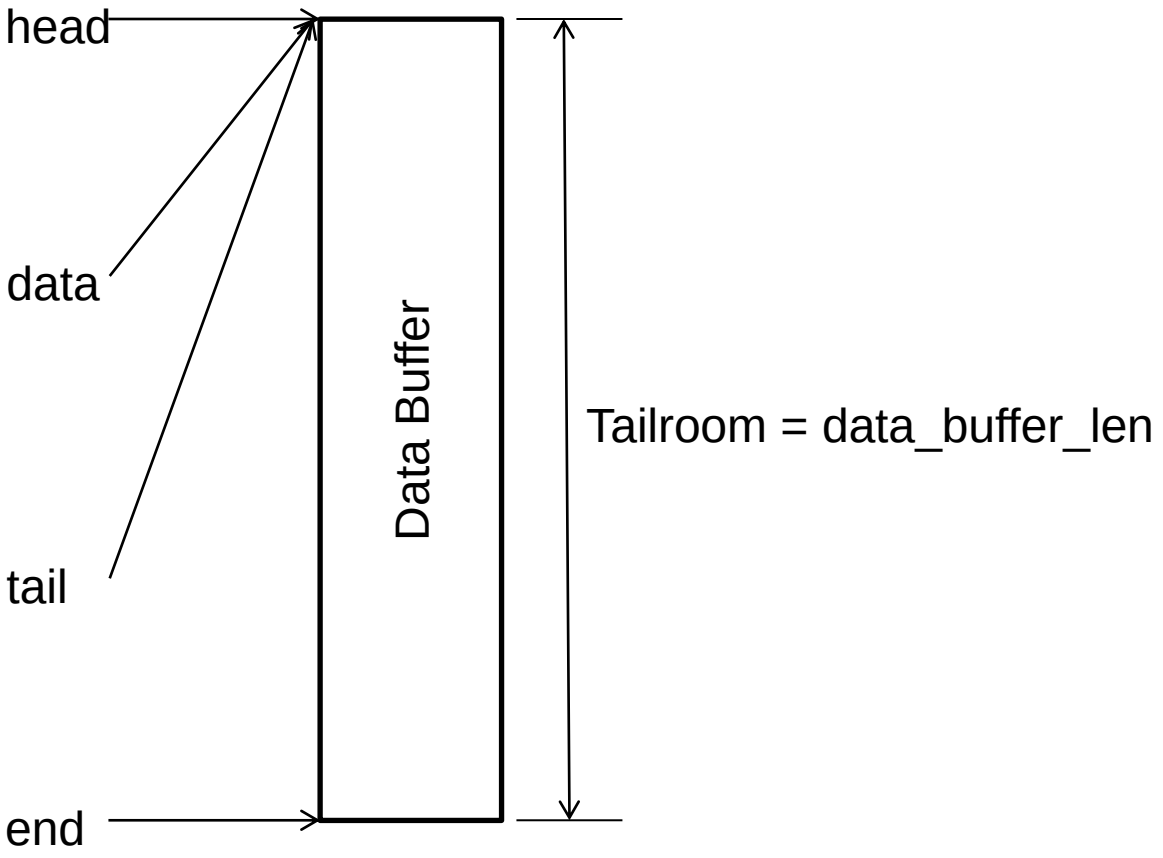


struct sk_buff & its Data Area (Buffer)

TX Processing: sk_buff view after SKB allocation

OPERATION:
struct sk_buff * alloc_skb (int data_buffer_len, GFP_KERNEL)

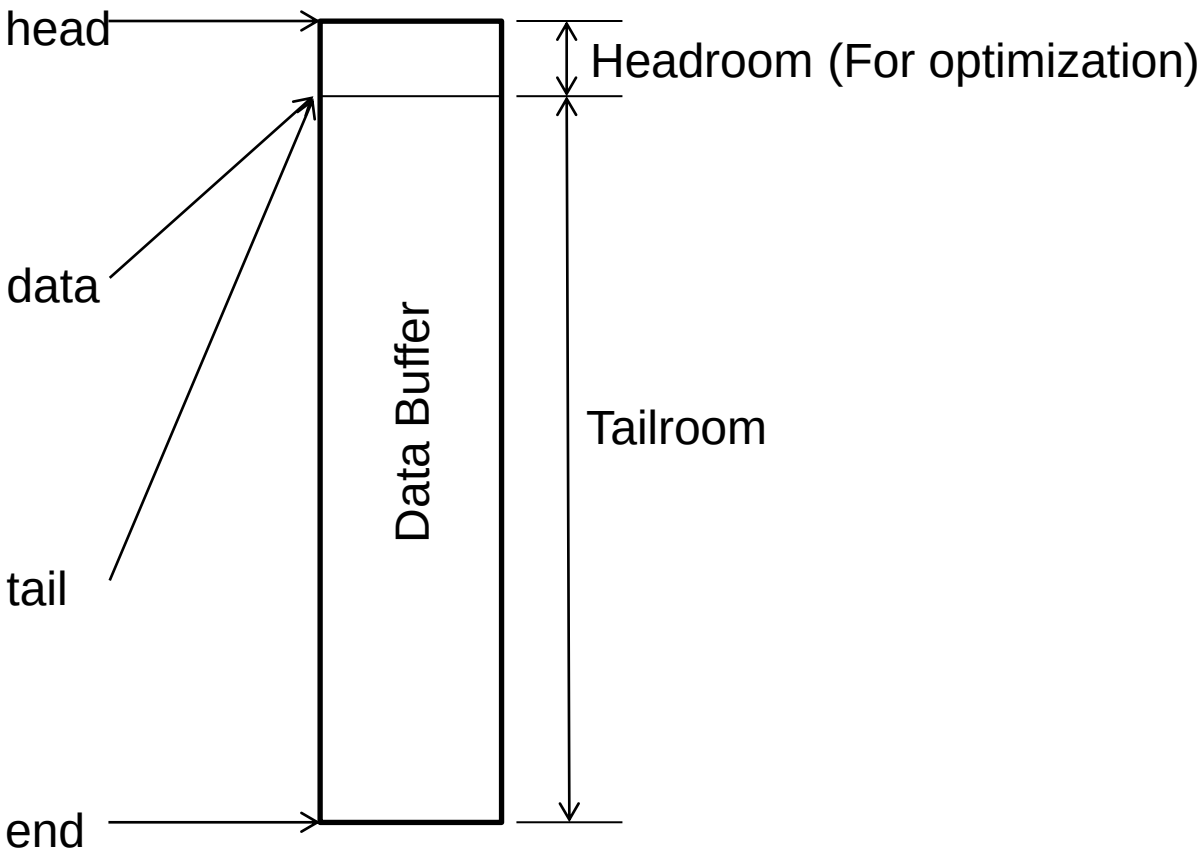
sk_buff->len = 0, Headroom = 0



TX Processing: sk_buff view after SKB allocation-2

OPERATION:
struct sk_buff * dev_alloc_skb (int data_buffer_len)

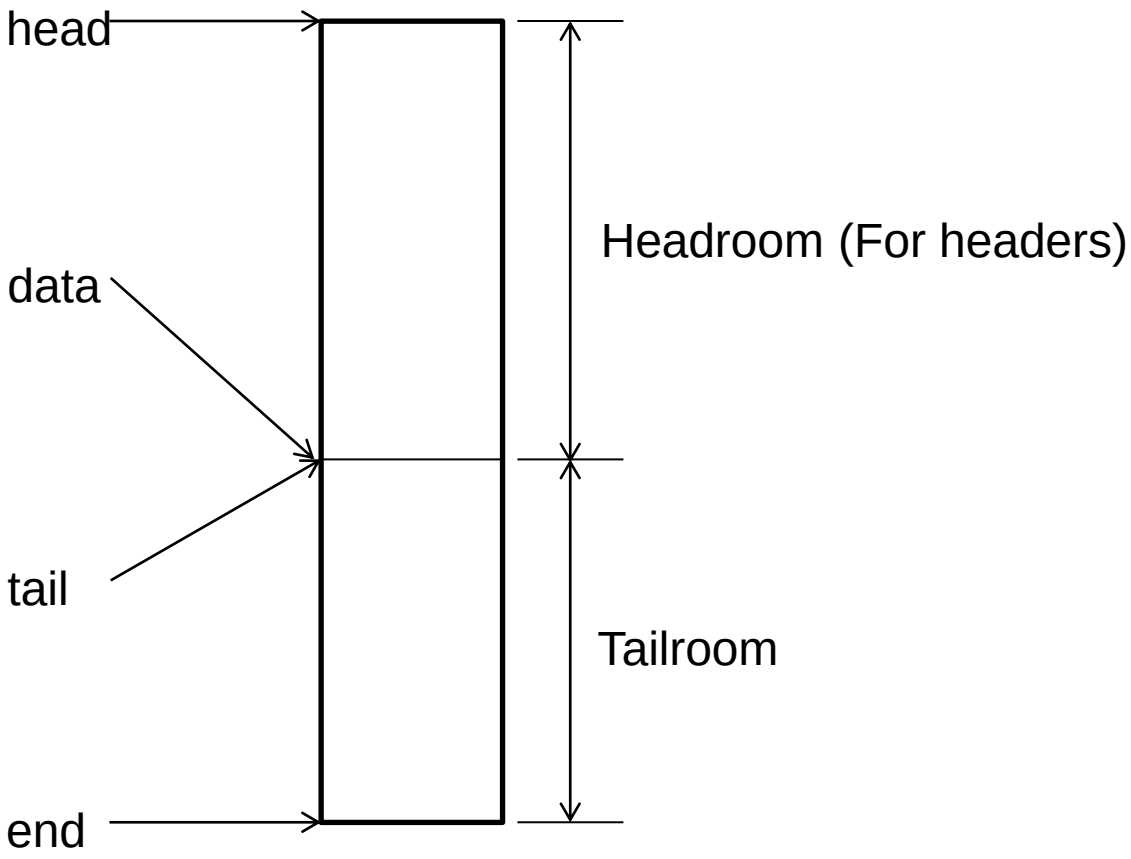
sk_buff->len = 0, data_buffer_len = Headroom + Tailroom



TX Processing: sk_buff view after creating Headroom

OPERATION:
void skb_reserve (struct sk_buff *skb, int Total_header_len)

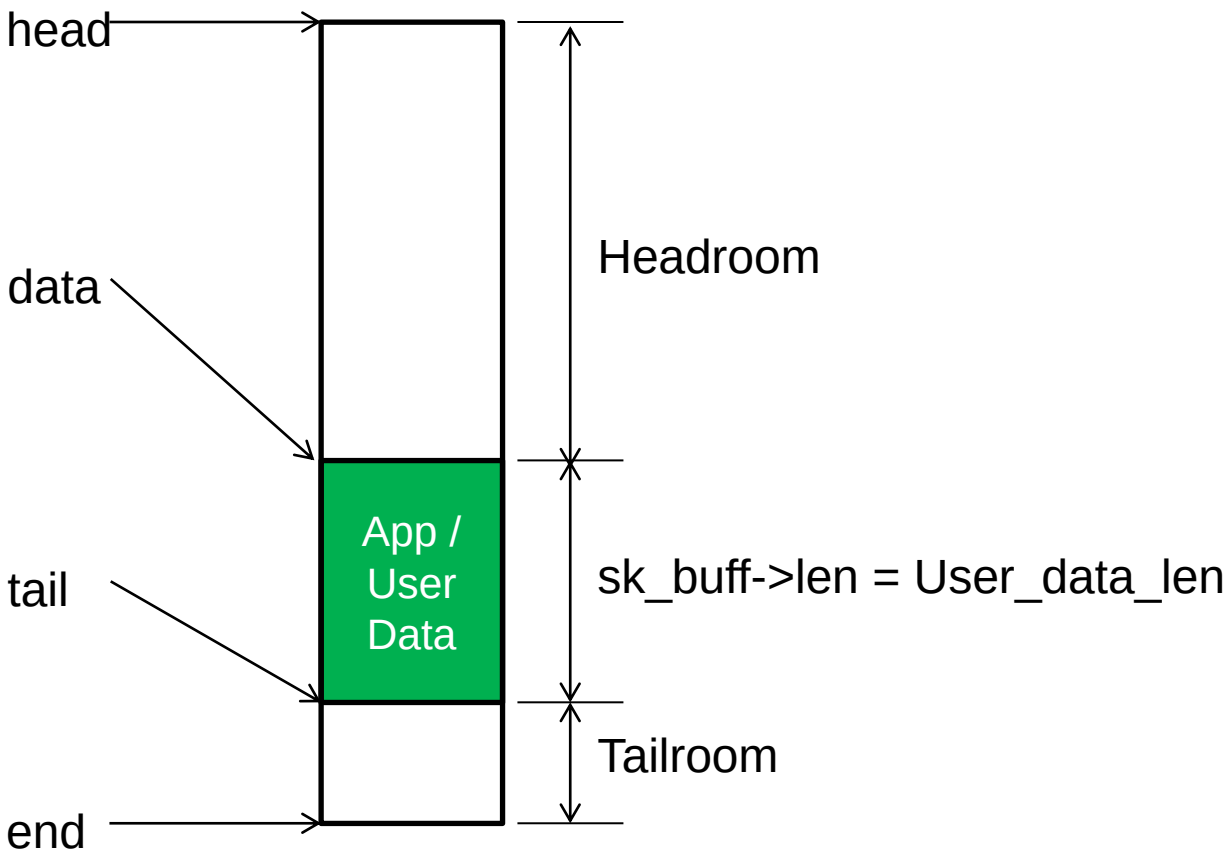
sk_buff->len = 0, Total_header_len = TCP/UDP hdr len + IP hdr len + MAC hdr len



TX Processing: sk_buff view after adding App/User Data

OPERATION:
void * skb_put (struct sk_buff *skb, int User_data_len)

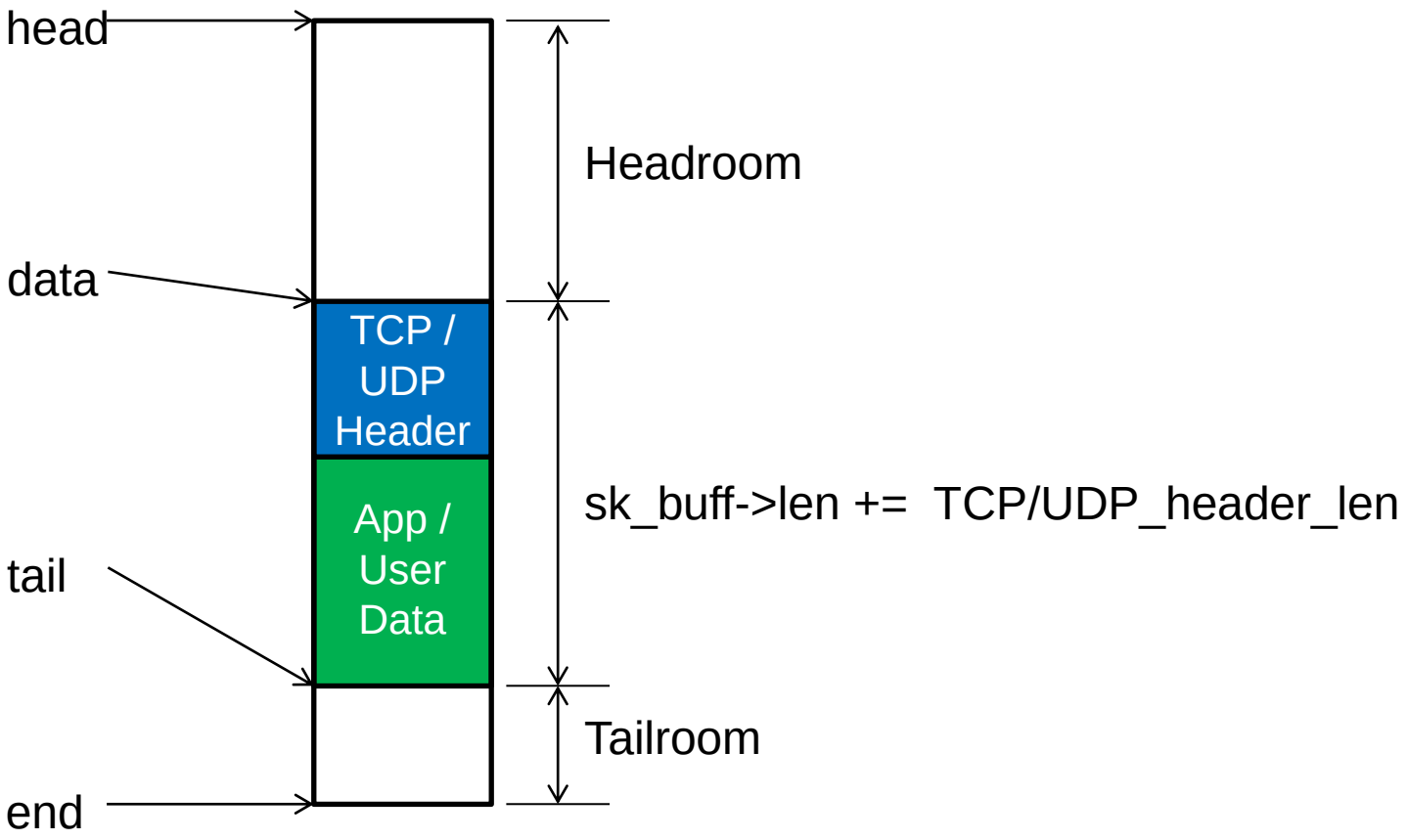
Returns pointer to user data.



TX Processing: sk_buff view after adding TCP/UDP Header

OPERATION:
void * skb_push (struct sk_buff *skb, int TCP/UDP_header_len)

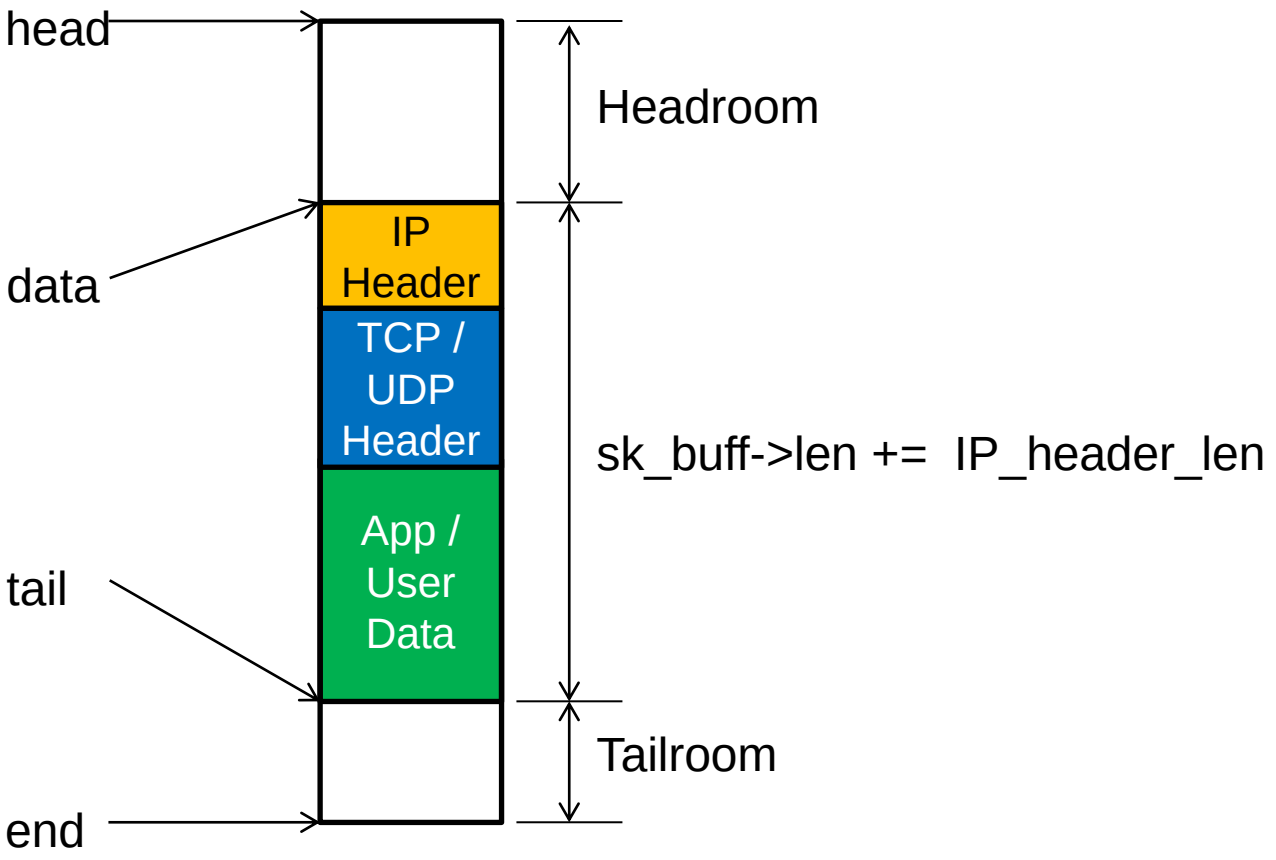
Returns pointer to TCP/UDP header (struct tcphdr * / struct udphdr *)



TX Processing: sk_buff view after adding IP Header

OPERATION:
void * skb_push (struct sk_buff *skb, int IP_header_len)

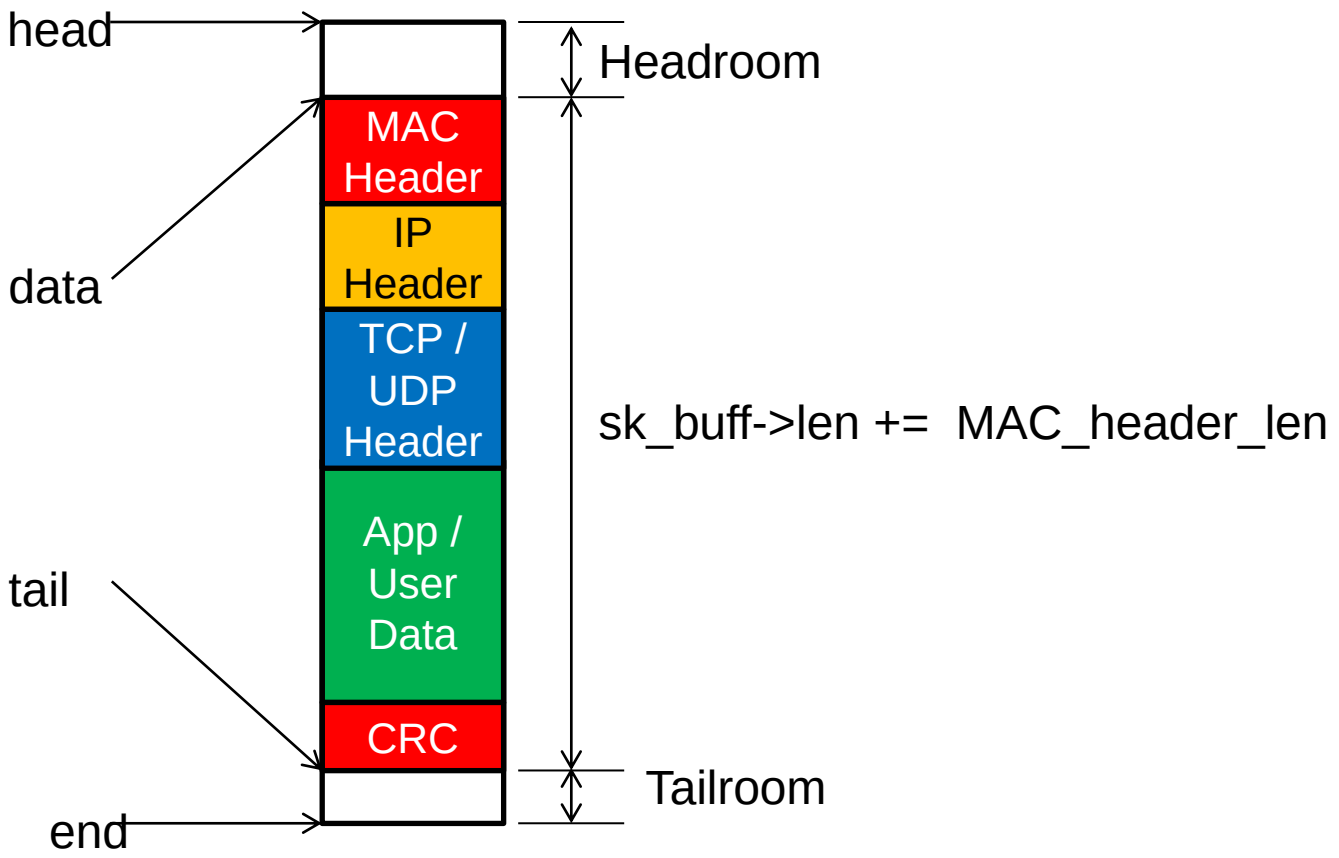
Returns pointer to IP header (struct iphdr *)



TX Processing: sk_buff view after adding MAC Header

OPERATION:
void * skb_push (struct sk_buff *skb, int MAC_header_len)

Returns pointer to MAC header (struct machdr *)

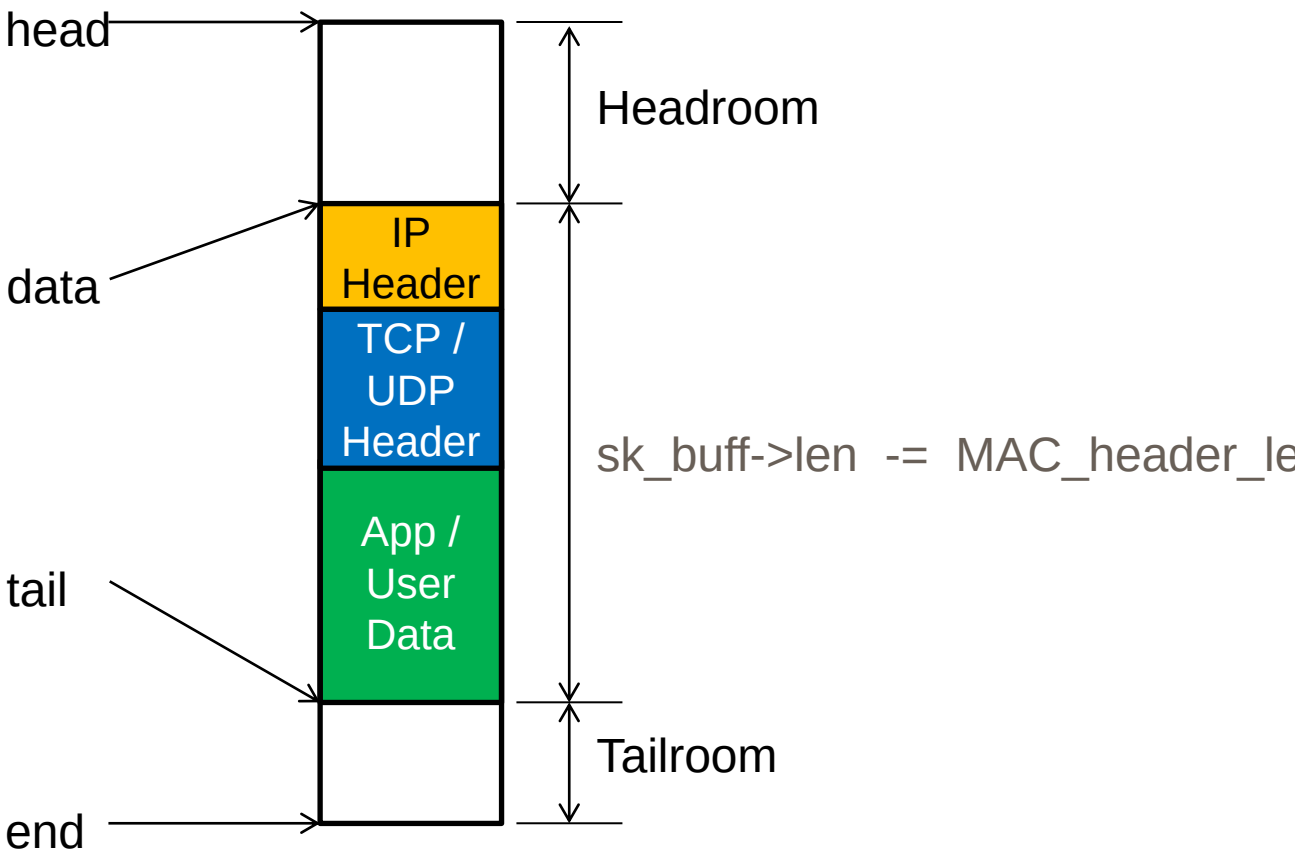


NOTE: Separate operation is required for adding CRC to Tailroom.

RX Processing: sk_buff view after removing MAC header

OPERATION:
void * skb_pull (struct sk_buff *skb, int MAC_header_len)

Returns pointer to IP header (struct iphdr *)

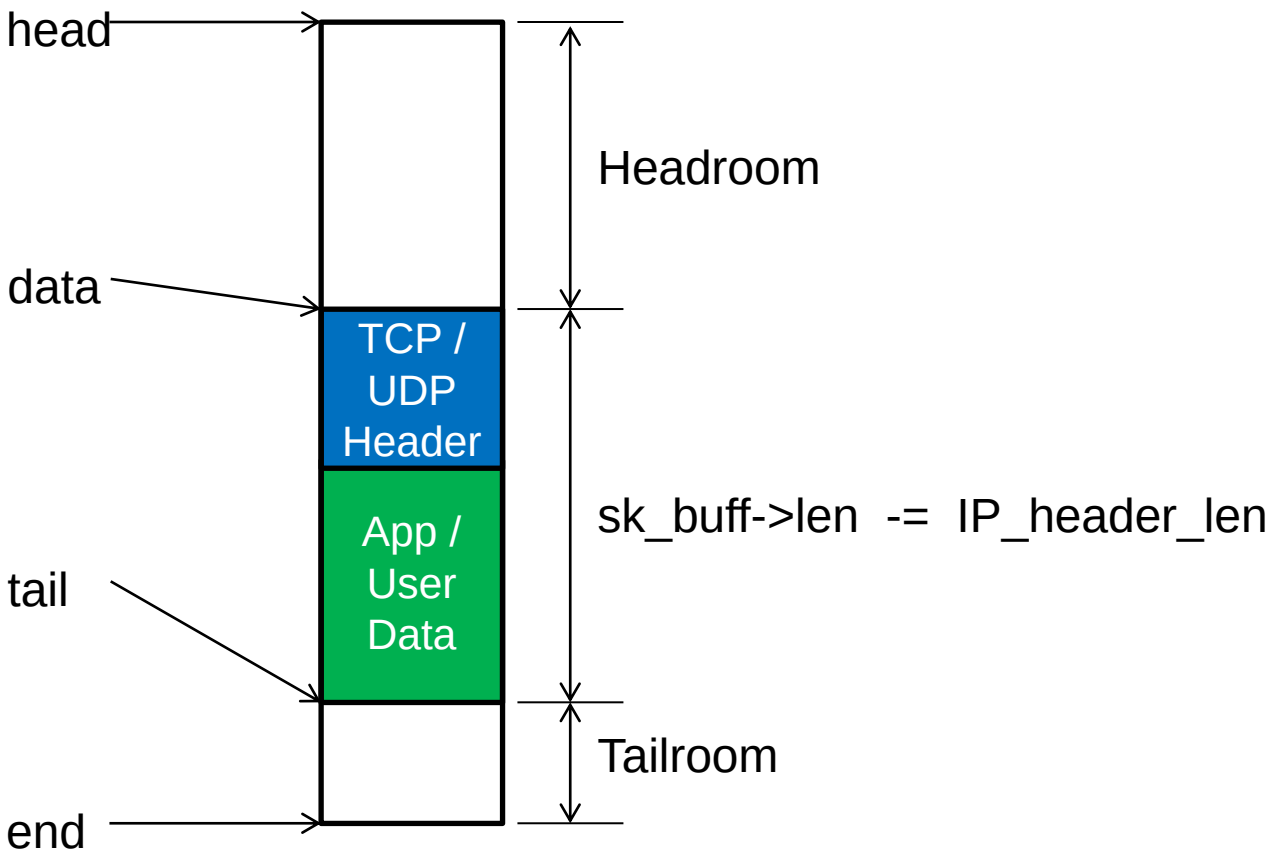


NOTE: Separate operation is required for removing CRC from Tailroom.

RX Processing: sk_buff view after removing IP header

OPERATION:
void * skb_pull (struct sk_buff *skb, int IP_header_len)

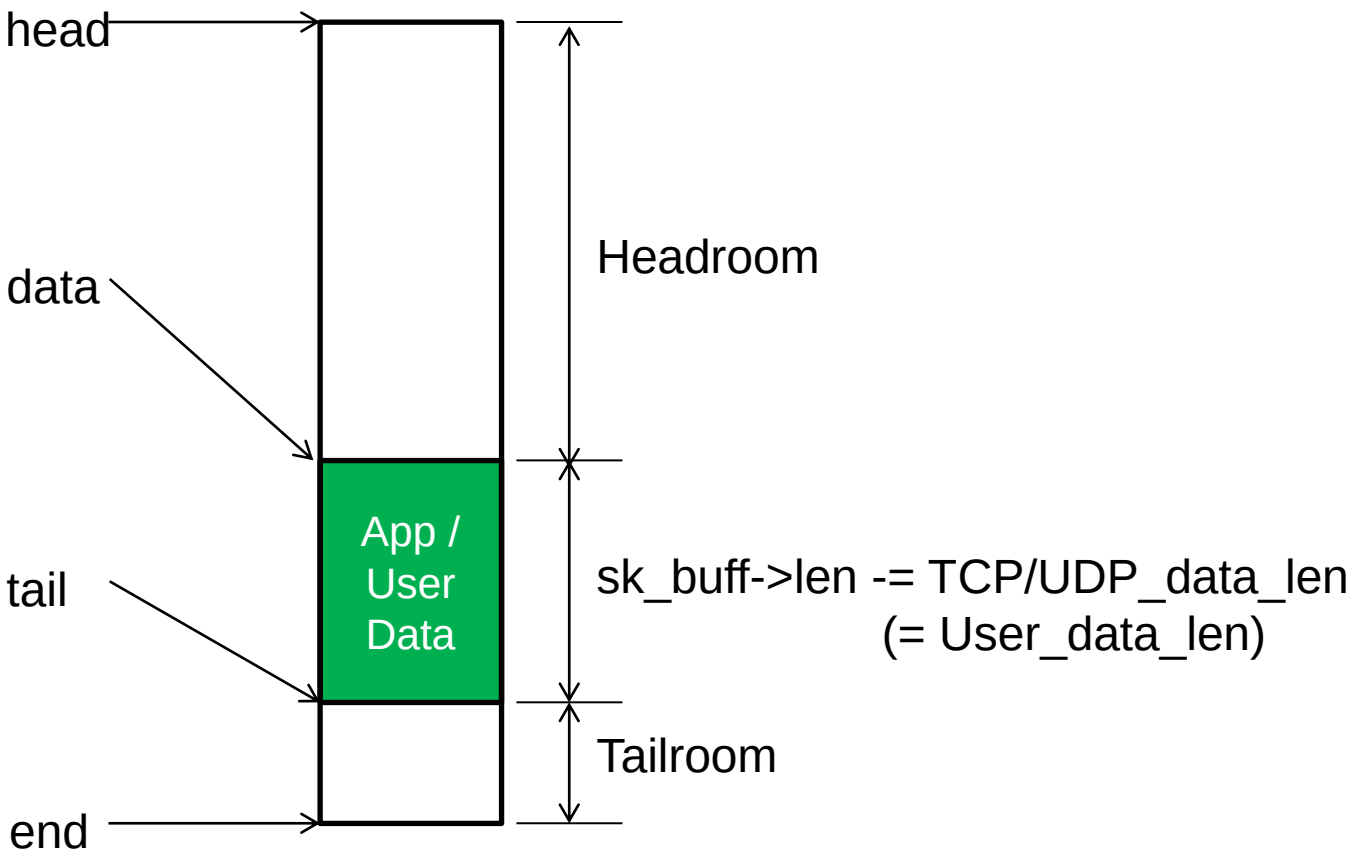
Returns pointer to TCP/UDP header (struct tcphdr * / struct udphdr *)



RX Processing: sk_buff view after removing TCP/UDP header

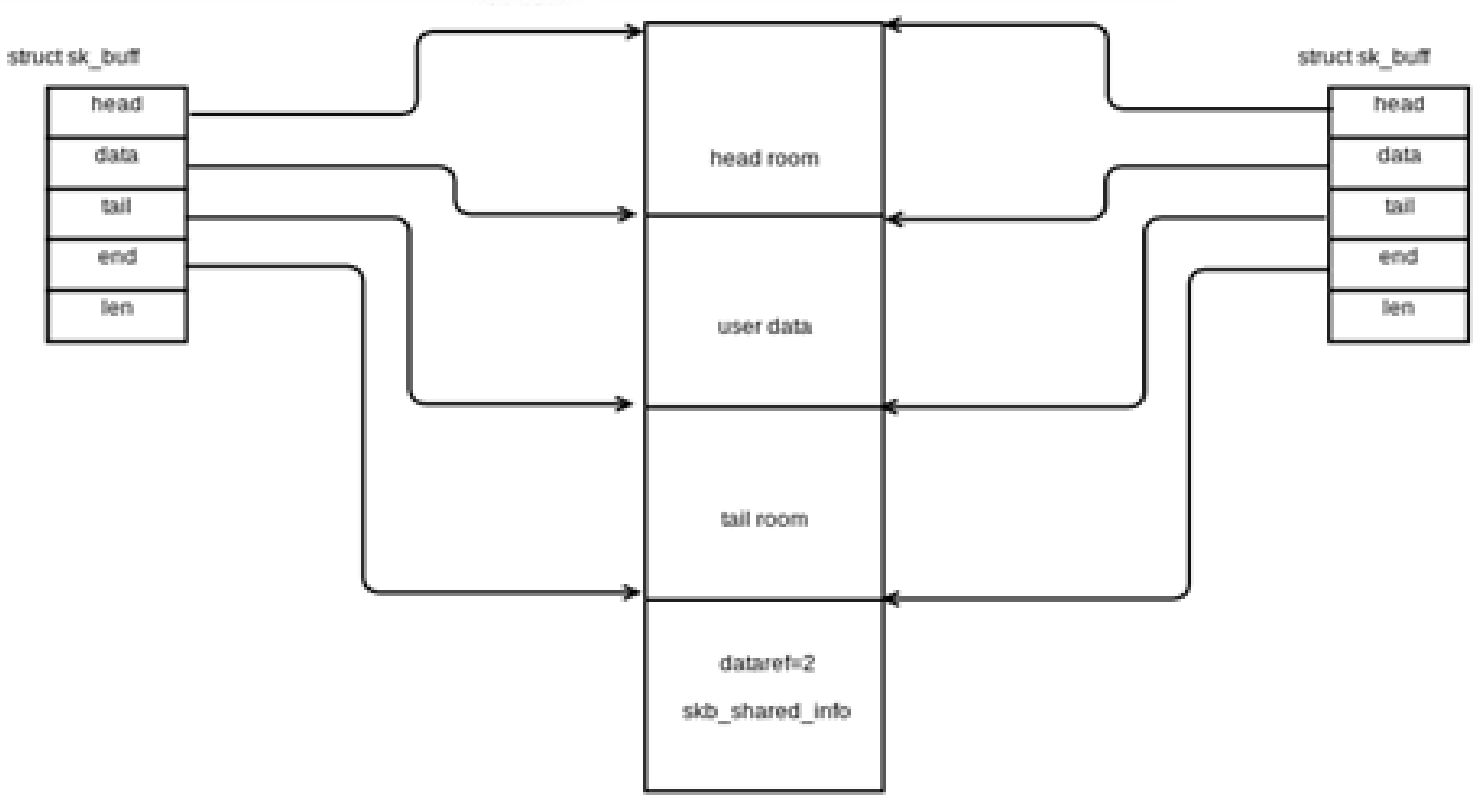
OPERATION:
void * skb_pull (struct sk_buff *skb, int TCP/UDP_header_len)

Returns pointer to user data.



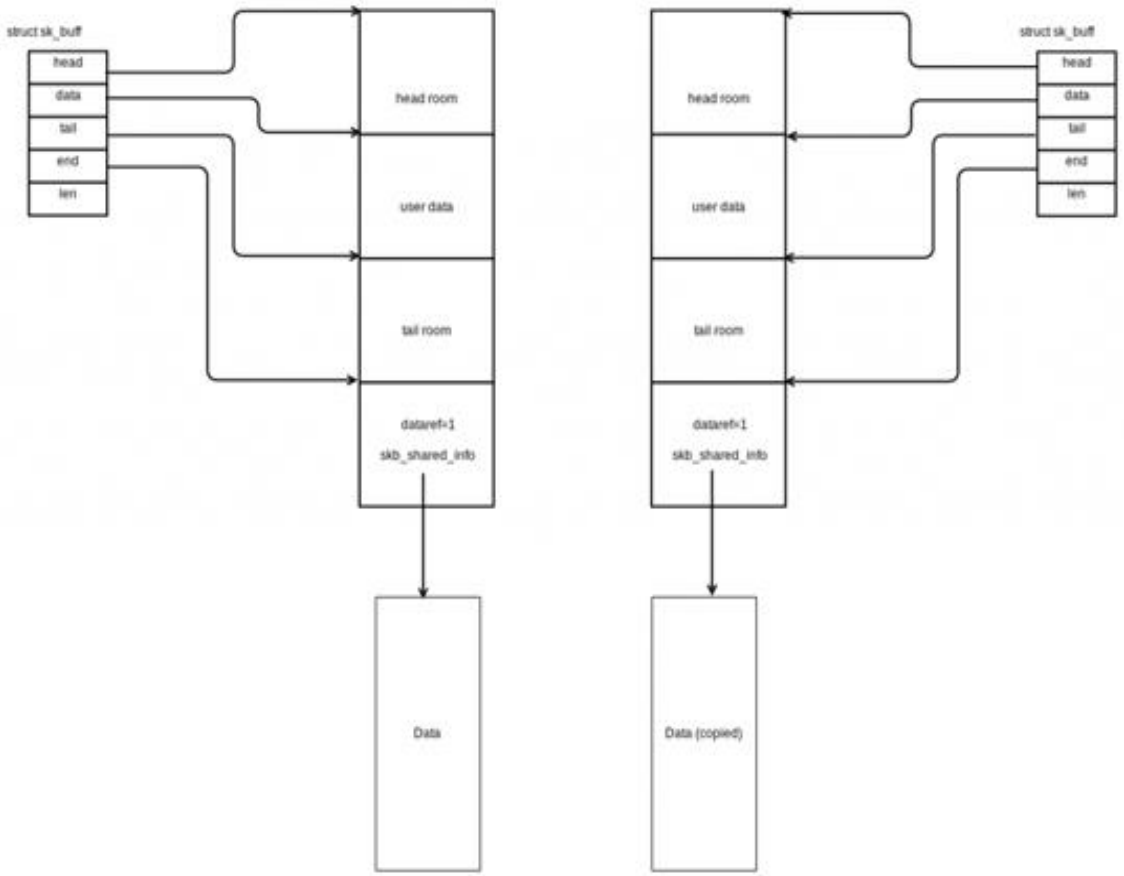
sk_buff - Cloning

OPERATION:
struct sk_buff * skb_clone (struct sk_buff *skb, int gfp_mask)



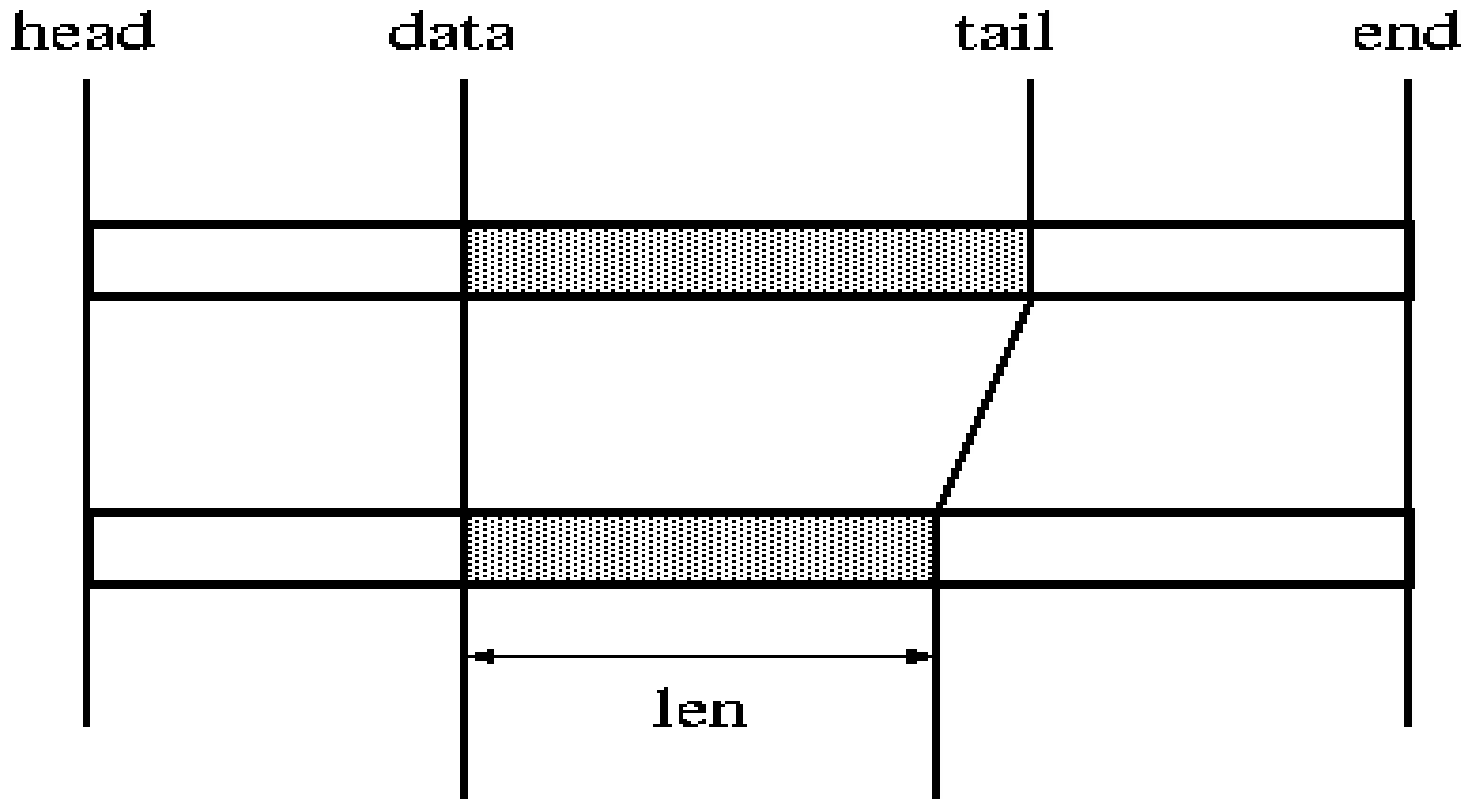
sk_buff - Copying

OPERATION:
struct sk_buff * skb_copy (struct sk_buff *skb, int gfp_mask)

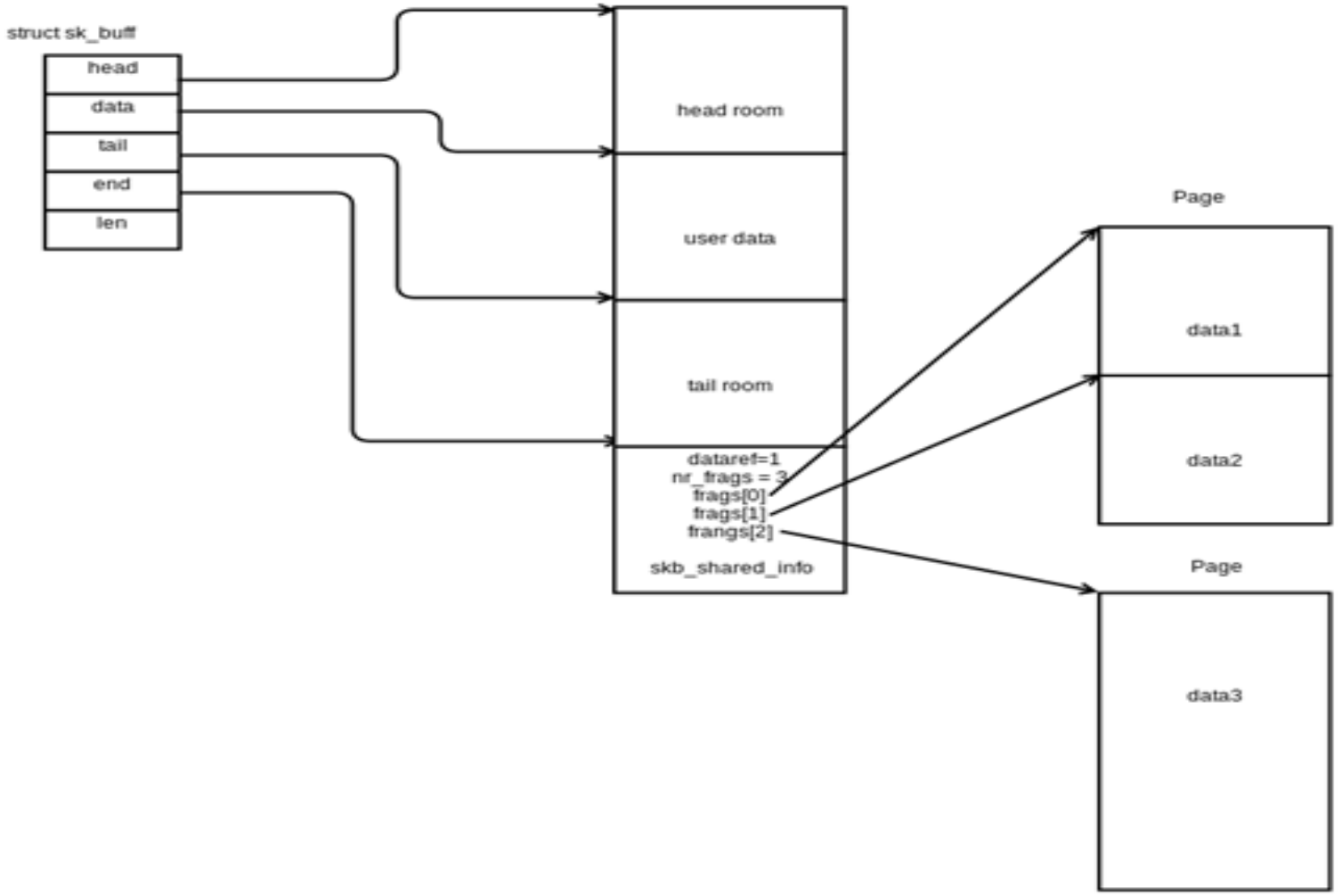


sk_buff – Trimming

OPERATION:
void skb_trim (struct sk_buff *skb, unsigned int len)



Non-Linear (Fragmented) sk_buff



sk_buff - Additional Support Functions

Free SKB in Kernel:

void kfree_skb (struct sk_buff *skb)

Free SKB in Device Driver:

void dev_kfree_skb (struct sk_buff *skb)

Return the headroom length:

unsigned int skb_headroom (struct sk_buff *skb)

Return the tailroom length:

int skb_tailroom (struct sk_buff *skb)

sk_buff - Additional Support Functions (contd.)

Initialize SKB queue:

void skb_queue_head _init (struct sk_buff_head *list)

Add SKB at the head of the SKB queue:

void skb_queue_head (struct sk_buff_head *list, struct sk_buff *skb)

Add SKB at the tail of the SKB queue:

void skb_queue_tail (struct sk_buff_head *list, struct sk_buff *skb)

Remove SKB from the head of the SKB queue:

struct sk_buff * skb_dequeue (struct sk_buff_head *list)

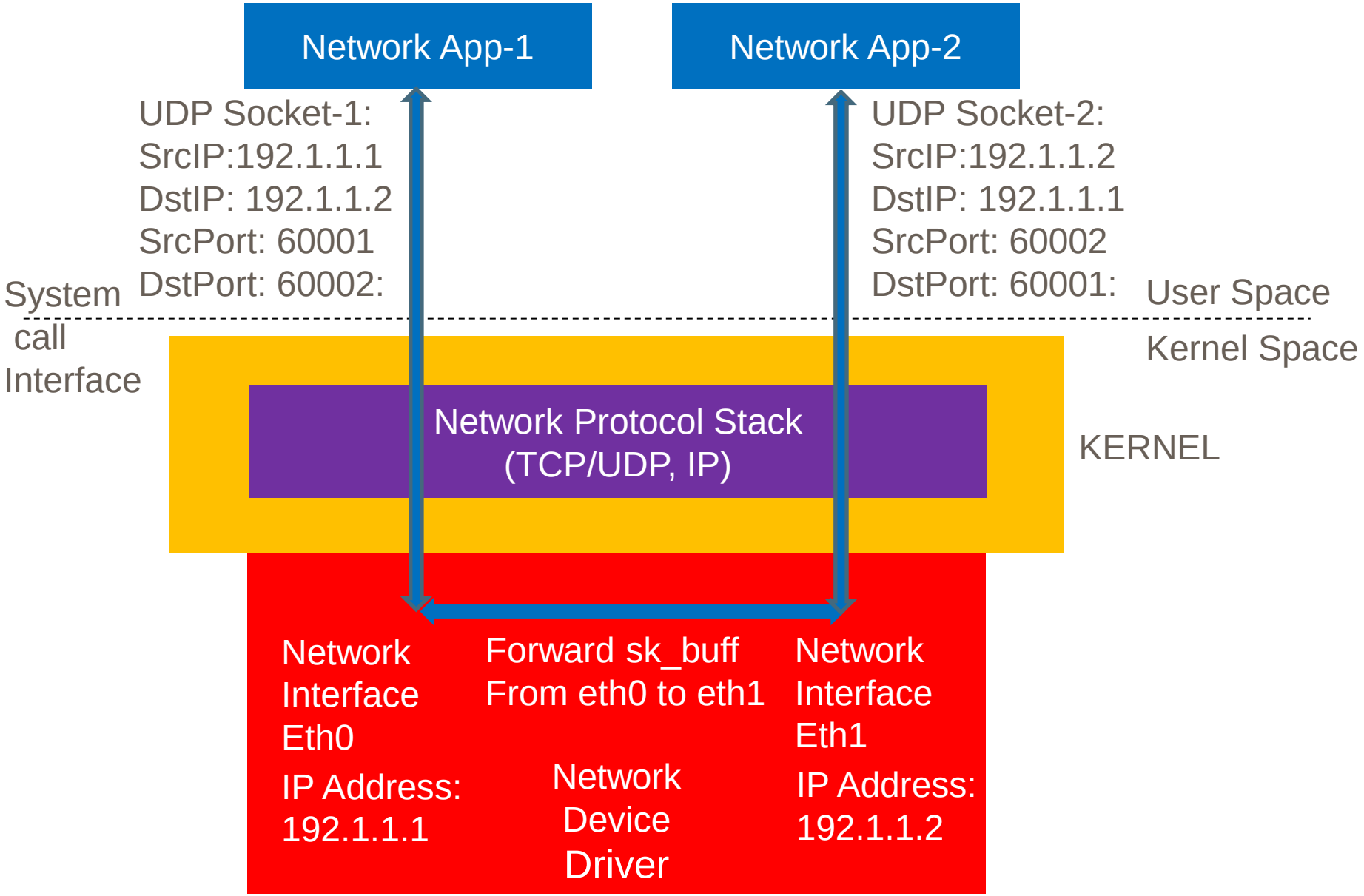
Remove SKB from the tail of the SKB queue:

struct sk_buff * skb_dequeue_tail (struct sk_buff_head *list)

Return the length of the SKB queue:

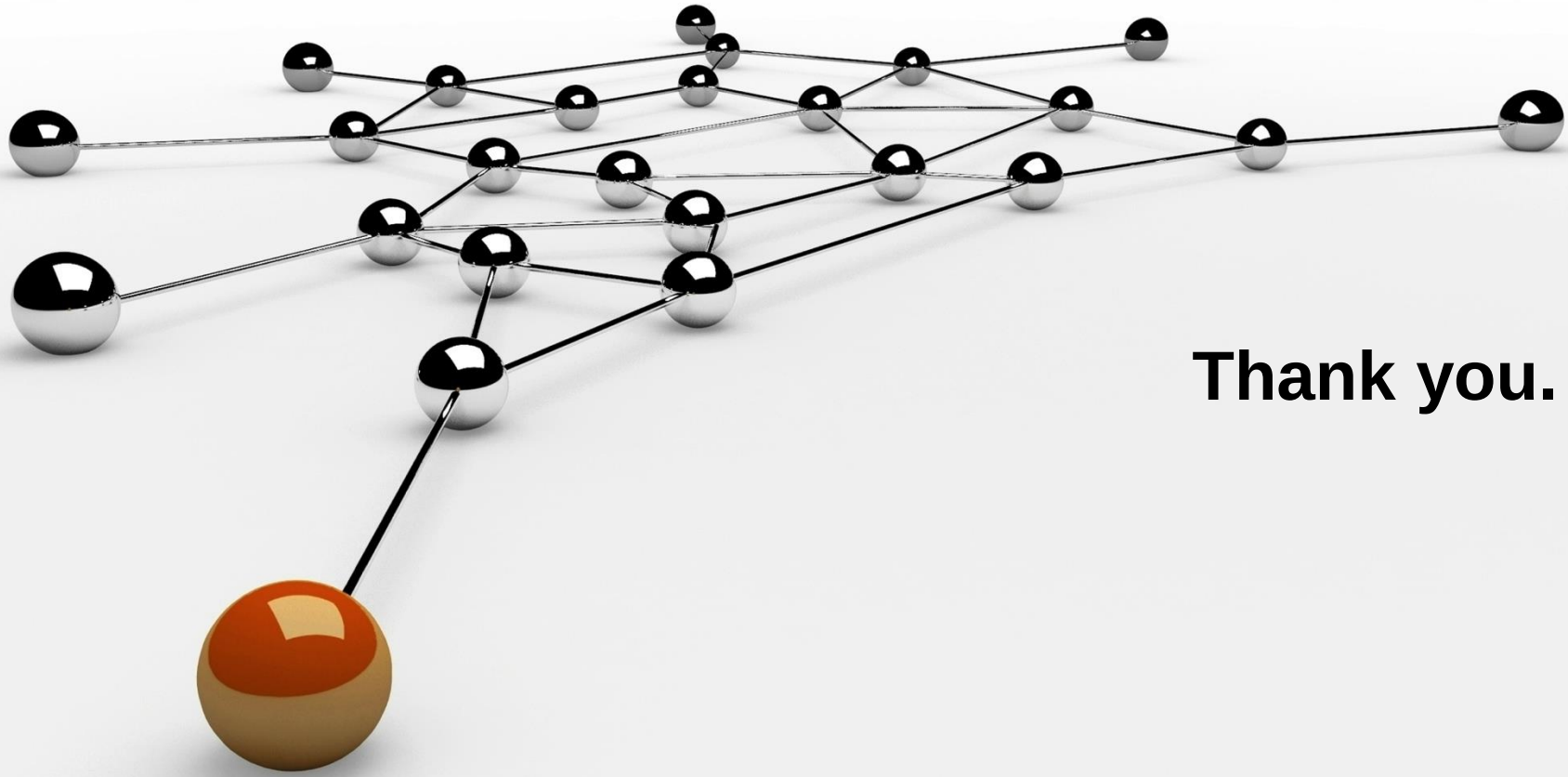
unsigned int skb_queue_len (struct sk_buff_head *list)

Virtual(Loopback) Network Device Driver - Architecture



Aricent

Engineering excellence. Sourced.



Thank you.