Network Programming

Lecture 03

Haitham A. El-Ghareeb

Faculty of Computers and Information Sciences
Mansoura University
Egypt
helghareeb@mans.edu.eg

February 26, 2017

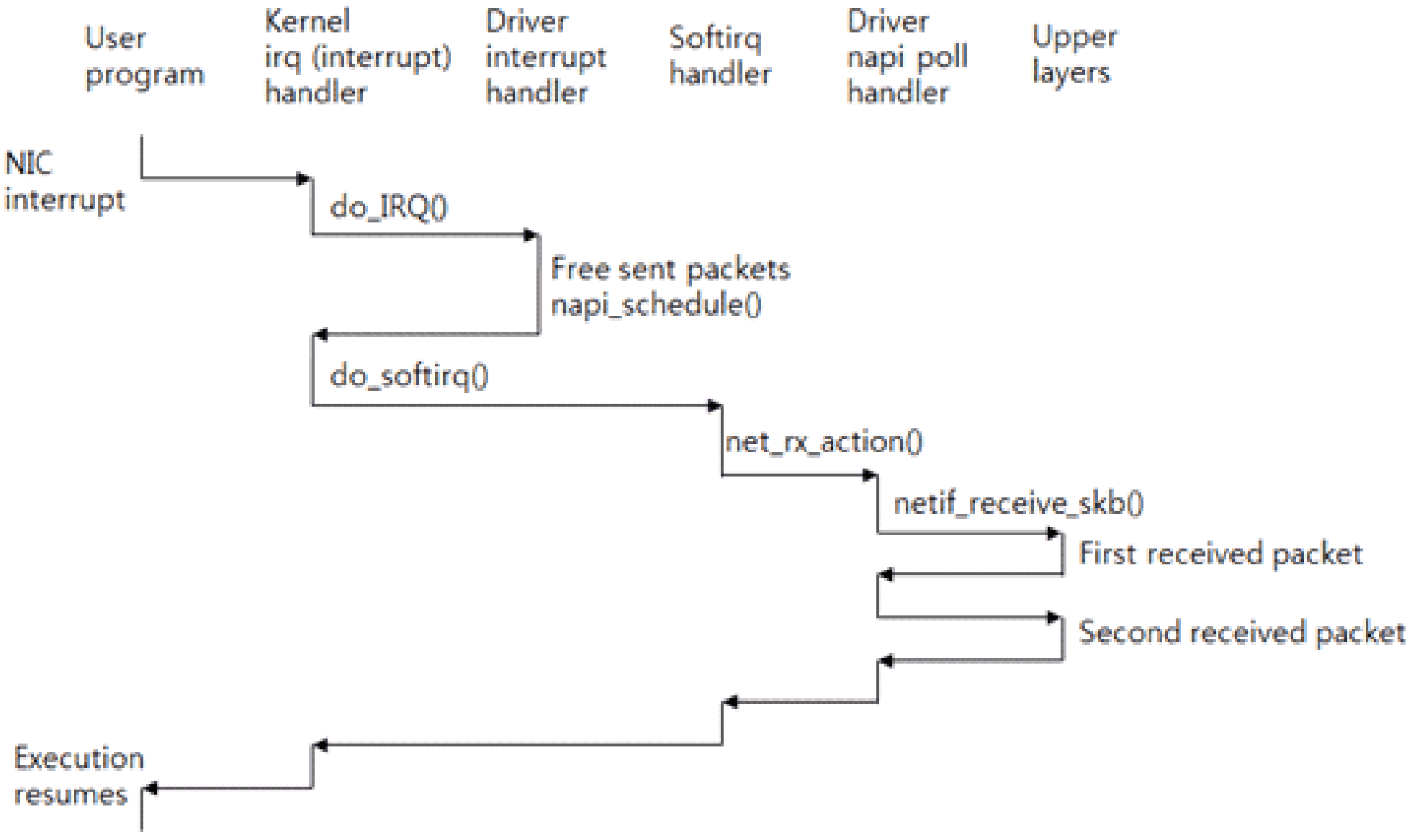
Data Structure

- sk_buff
- TCP Control Block
- TCP Connection Lookup Table

Next Week InchALLAH

Homework

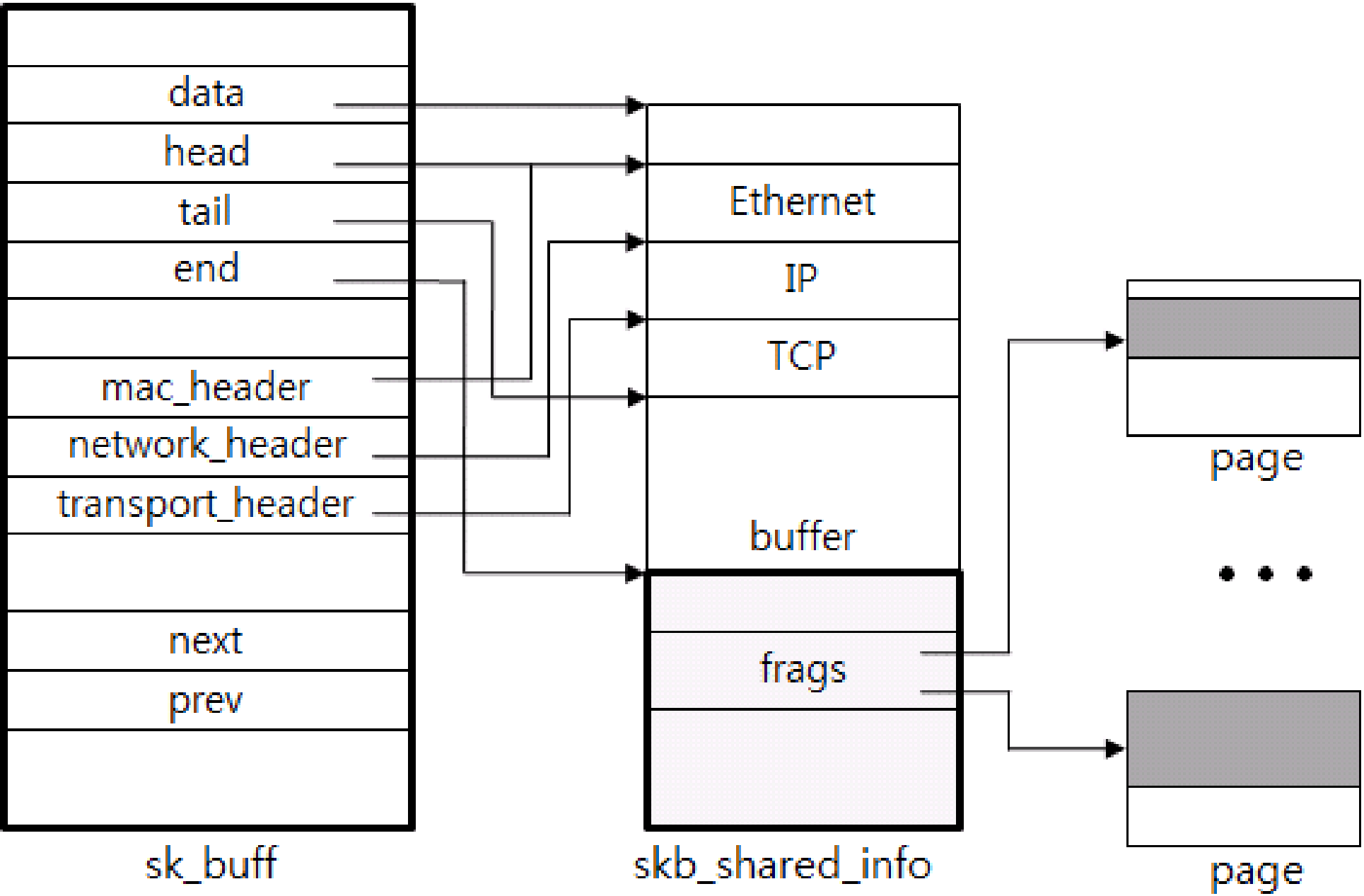
Following content is based on
http://www.cubrid.org/blog/dev-platform/understanding-tcp-ip-network-stack/


User program
Kernel irq (interrupt) handler
Driver interrupt handler
Softirq handler
Driver napi poll handler
Upper layers
NIC interrupt
do_IRQ()
Free sent packets
napi_schedule()
do_softirq()
net_rx_action()
netif_receive_skb()
First received packet
Second received packet
Execution resumes

Some Data Structures

The followings are some key data structures:

- sk_buff structure
- TCP Control Block
- net_device structure (NIC - skipped)
- TCP Connection Lookup Table


data
head
tail
end
mac_header
network_header
transport_header
next
prev
sk_buff
Ethernet
IP
TCP
buffer
frags
skb_shared_info
page
page

sk_buff

sk_buff

- **skb** structure that means a **packet**.

sk_buff

- **skb** structure that means a **packet**.
- Basic functions are:
  - Including Packet Data and *meta data*
  - How to Add or Delete a Header
  - How to Combine and Divide Packet
  - Quick Allocation and Free

skb - Including Packet Data and *meta data*

skb - Including Packet Data and *meta data*

- Structure directly includes packet data or refers to it by using a pointer.

skb - Including Packet Data and *meta data*

- Structure directly includes packet data or refers to it by using a pointer.
- In Figure, some of the packets refer to using the data pointer and the additional data (frags) refer to the actual page.

skb - Including Packet Data and *meta data*

- Structure directly includes packet data or refers to it by using a pointer.
- In Figure, some of the packets refer to using the data pointer and the additional data (frags) refer to the actual page.
- Necessary information, such as header and payload length is saved in the *meta data* area.

skb - Including Packet Data and *meta data*

- Structure directly includes packet data or refers to it by using a pointer.
- In Figure, some of the packets refer to using the data pointer and the additional data (frags) refer to the actual page.
- Necessary information, such as header and payload length is saved in the *meta data* area.
- In Figure, mac_header, network_header, transport_header have corresponding pointer data.

skb - Including Packet Data and *meta data*

- Structure directly includes packet data or refers to it by using a pointer.
- In Figure, some of the packets refer to using the data pointer and the additional data (frags) refer to the actual page.
- Necessary information, such as header and payload length is saved in the *meta data* area.
- In Figure, mac_header, network_header, transport_header have corresponding pointer data.
- Pointers point to the starting position of Ethernet header, IP header, and TCP header respectively.

skb - Including Packet Data and *meta data*

- Structure directly includes packet data or refers to it by using a pointer.
- In Figure, some of the packets refer to using the data pointer and the additional data (frags) refer to the actual page.
- Necessary information, such as header and payload length is saved in the *meta data* area.
- In Figure, mac_header, network_header, transport_header have corresponding pointer data.
- Pointers point to the starting position of Ethernet header, IP header, and TCP header respectively.
- This way makes TCP protocol processing easy.

How to Add or Delete a Header

How to Add or Delete a Header

- Header is added or deleted as up and down each layer of the network stack.

How to Add or Delete a Header

- Header is added or deleted as up and down each layer of the network stack.
- Pointers are used for more efficient processing.

How to Add or Delete a Header

- Header is added or deleted as up and down each layer of the network stack.
- Pointers are used for more efficient processing.
- For example, to remove the Ethernet header, just increase the head pointer.

How to Combine and Divide Packet

How to Combine and Divide Packet

- Linked-List is used for efficient execution of tasks such as adding or deleting packet payload data to the socket buffer, or packet chain.

How to Combine and Divide Packet

- Linked-List is used for efficient execution of tasks such as adding or deleting packet payload data to the socket buffer, or packet chain.
- The next pointer and pre pointer are used for this purpose.

Quick Allocation and Free

Quick Allocation and Free

- As a structure is allocated whenever creating a packet, the quick allocator is used.

Quick Allocation and Free

- As a structure is allocated whenever creating a packet, the quick allocator is used.
- For example, if data is transmitted at the speed of 10-Gigabit Ethernet, more than one million packets per second must be created and deleted.

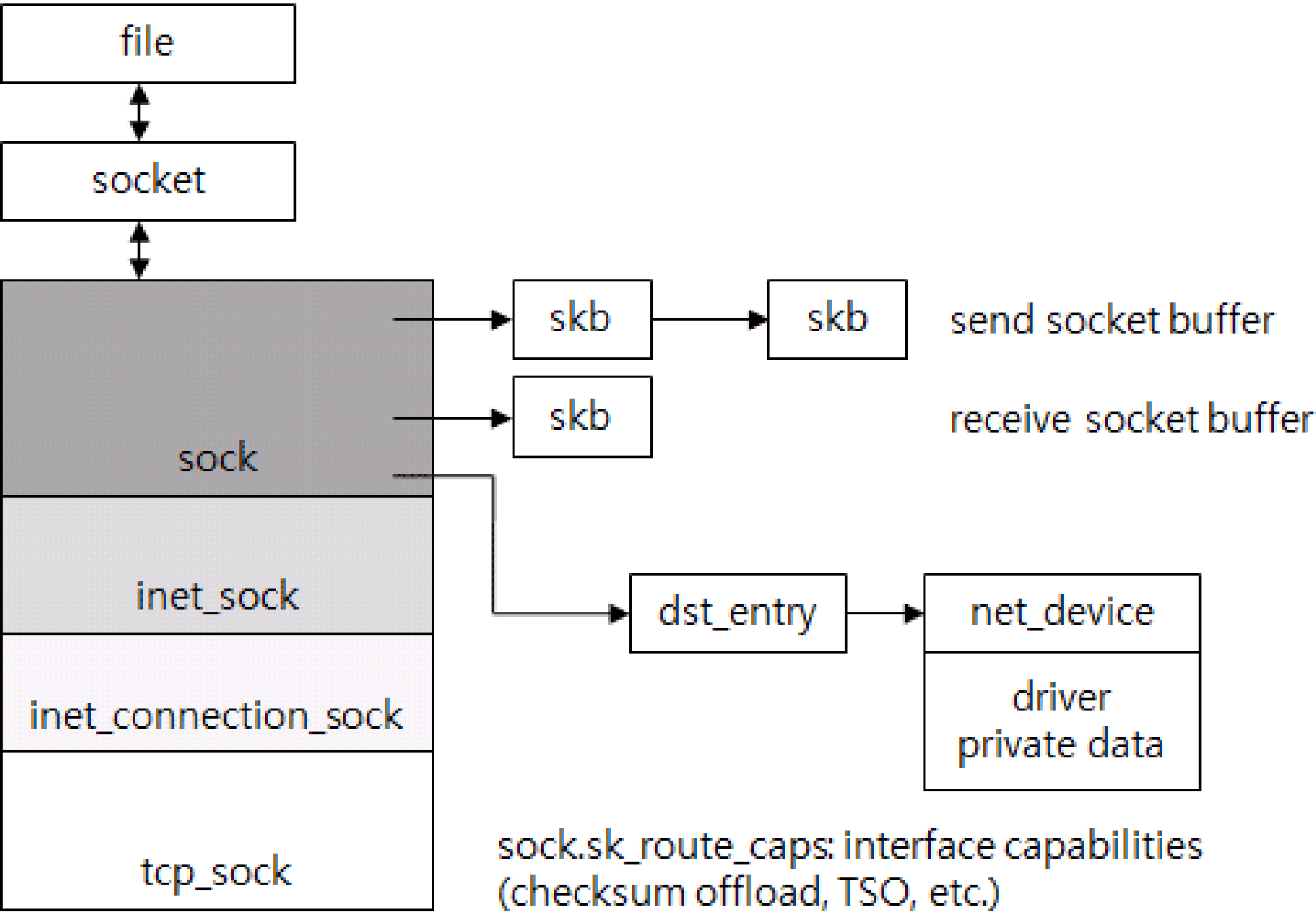
TCP Control Block - 1

TCP Control Block - 1

- Linux uses tcp_sock to represent the TCP connection.

TCP Control Block - 1

- Linux uses tcp_sock to represent the TCP connection.
- Here is the relationship among the file, the socket, and the tcp_sock.


file
socket
sock
inet_sock
inet_connection_sock
tcp_sock
skb
skb
skb
send socket buffer
receive socket buffer
dst_entry
net_device
driver
private data
sock.sk_route_caps: interface capabilities
(checksum offload, TSO, etc.)

TCP Control Block - 2

TCP Control Block - 2

- When a system call has occurred, it searches the file in the file descriptor used by the application that has caled the system call.

TCP Control Block - 2

- When a system call has occurred, it searches the file in the file descriptor used by the application that has caled the system call.
- For the Unix-series OS, the socket, the file, and the device for general file system for storage are abstracted to a file.

TCP Control Block - 2

- When a system call has occurred, it searches the file in the file descriptor used by the application that has caled the system call.
- For the Unix-series OS, the socket, the file, and the device for general file system for storage are abstracted to a file.
- Therefore, the file structure includes the least information.

TCP Control Block - 2

- When a system call has occurred, it searches the file in the file descriptor used by the application that has caled the system call.
- For the Unix-series OS, the socket, the file, and the device for general file system for storage are abstracted to a file.
- Therefore, the file structure includes the least information.
- For a socket, a separate socket structure saves the socket-related information and the file refers to the socket as a pointer.

TCP Control Block - 2

- When a system call has occurred, it searches the file in the file descriptor used by the application that has caled the system call.
- For the Unix-series OS, the socket, the file, and the device for general file system for storage are abstracted to a file.
- Therefore, the file structure includes the least information.
- For a socket, a separate socket structure saves the socket-related information and the file refers to the socket as a pointer.
- The socket refers to the **tcp_sock** again.

TCP Control Block - 2

- When a system call has occurred, it searches the file in the file descriptor used by the application that has caled the system call.
- For the Unix-series OS, the socket, the file, and the device for general file system for storage are abstracted to a file.
- Therefore, the file structure includes the least information.
- For a socket, a separate socket structure saves the socket-related information and the file refers to the socket as a pointer.
- The socket refers to the **tcp_sock** again.
- **tcp_sock** is classified into **sock, inet_sock**, etc. to support various protocols except TCP (kind of polymorphism).

TCP Control Block - 3

TCP Control Block - 3

- All status information used by TCP protocol is saved in **tcp_sock** (sequence number, receive window, etc.).

TCP Control Block - 3

- All status information used by TCP protocol is saved in **tcp_sock** (sequence number, receive window, etc.).
- send socket buffer and receive socket buffer are the **sk_buff** lists and they include the **tcp_sock**.

TCP Control Block - 3

- All status information used by TCP protocol is saved in **tcp_sock** (sequence number, receive window, etc.).
- send socket buffer and receive socket buffer are the **sk_buff** lists and they include the **tcp_sock**.
- **dst_entry** (the IP routing result) is referred to in order to avoid frequent routing.

TCP Control Block - 3

- All status information used by TCP protocol is saved in **tcp_sock** (sequence number, receive window, etc.).
- send socket buffer and receive socket buffer are the **sk_buff** lists and they include the **tcp_sock**.
- **dst_entry** (the IP routing result) is referred to in order to avoid frequent routing.
- **dst_entry** allows for easy search of the ARP result, i.e., the destination MAC address.

TCP Control Block - 3

- All status information used by TCP protocol is saved in **tcp_sock** (sequence number, receive window, etc.).
- send socket buffer and receive socket buffer are the **sk_buff** lists and they include the **tcp_sock**.
- **dst_entry** (the IP routing result) is referred to in order to avoid frequent routing.
- **dst_entry** allows for easy search of the ARP result, i.e., the destination MAC address.
- **dst_entry** is part of the routing table.

TCP Control Block - 3

- All status information used by TCP protocol is saved in **tcp_sock** (sequence number, receive window, etc.).
- send socket buffer and receive socket buffer are the **sk_buff** lists and they include the **tcp_sock**.
- **dst_entry** (the IP routing result) is referred to in order to avoid frequent routing.
- **dst_entry** allows for easy search of the ARP result, i.e., the destination MAC address.
- **dst_entry** is part of the routing table.
- NIC is expressed as the **net_device** structure.

TCP Connection Lookup Table

TCP Connection Lookup Table

- Hash table used to search TCP connection where the received data belongs.

TCP Connection Lookup Table

- Hash table used to search TCP connection where the received data belongs.
- Hash is calculated by using Jenkins hash algorithm of the input data of:

TCP Connection Lookup Table

- Hash table used to search TCP connection where the received data belongs.
- Hash is calculated by using Jenkins hash algorithm of the input data of:
- source IP, target IP, source port, target port

TCP Connection Lookup Table

- Hash table used to search TCP connection where the received data belongs.
- Hash is calculated by using Jenkins hash algorithm of the input data of:
- source IP, target IP, source port, target port
- Hash function has been selected by considering defense against attacks to the hash table.

Next Week InchALLAH

Next Lecture

- Following Code
- Lab setup
- Basic *nix Networking commands
- Socket Programming in C++

Next Lab

- **Continue** Review TCP/IP Suite (Solve selected Qs)
- Qs can be found at `http://technologyeye.weebly.com/uploads/9/6/1/4/9614102/interview_question_networkingrajkumar.pdf`

Homework

- Study this lecture (very well)
- Prepare for the Next Lecture
- Check some Kernel code (mainly Networking Subsystem)
- Study the Lab file
- Check the github Course repository


YOU GO
NOW!