Justin Ciocoi

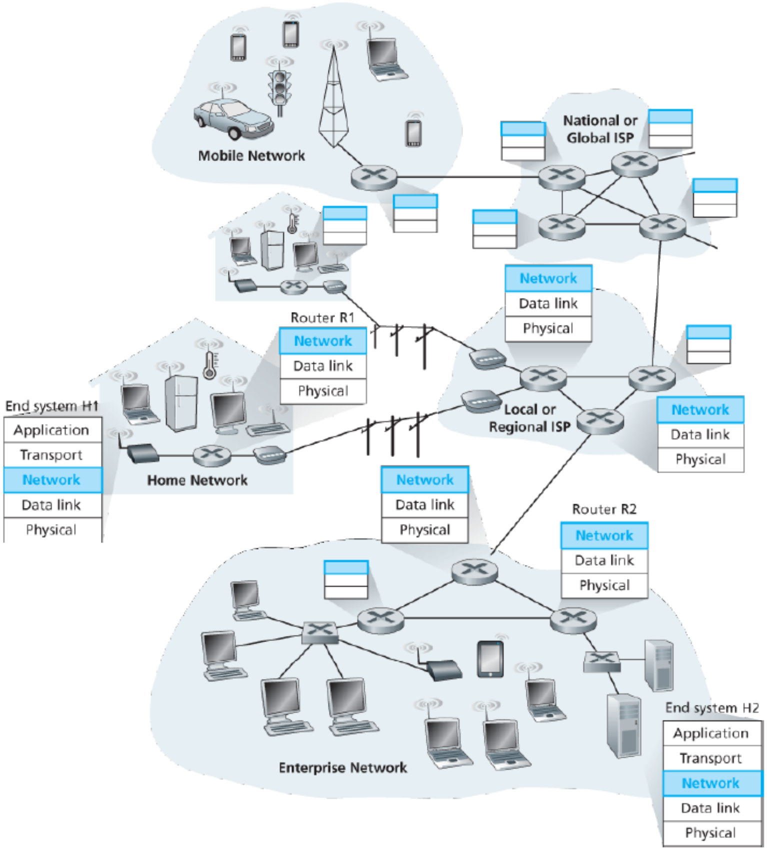
Oct. 6, 2023

CSCI 379 Textbook Notes

Computer Networking

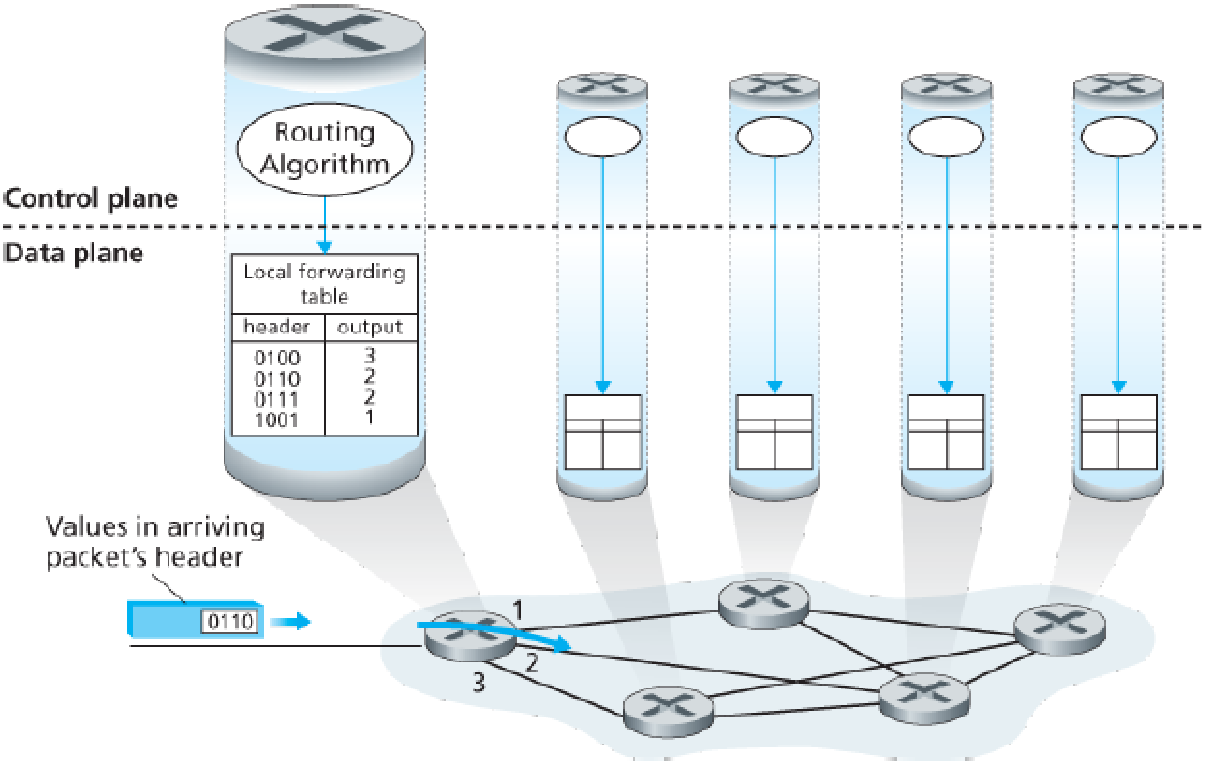
The Network Layer

4.1: Overview of the Network Layer



- Based on the above diagram, let us assume host **H1** is sending data to host **H2**
- The following will be the steps taken by the network to transmit that data
  - The network layer in **H1** takes segments from the transport layer in **H1** and encapsulates them into datagrams
  - **H1** then sends these datagrams to its nearby router, **R1**

  - On the receiving end, host **H2**'s network layer receives the datagrams from *its* nearby router, **R2**
  - **H2** then extracts the transport layer segments from the datagrams and delivers the segments to the transport layer in **H2**
- **4.1.1: Forwarding and Routing: The Data and Control Planes**
  - The network's layer primary role is to move packets from a sending host to a receiving host
  - Two important network layer functions in the facilitation of this movement can be identified
    - **Forwarding**
      - When a packet arrives at a router's *input* link, the router must move that packet to the appropriate *output* link
      - A packet might also be *blocked* if the origin is known to be malicious or if the destination is forbidden
      - Alternatively, data can also be *duplicated* and sent over multiple outgoing links
    - **Routing**
      - The network layer is responsible for determining the route that data will take from sender to receiver
      - *Routing algorithms* determine the path that the packet in the earlier example would take from host **H1** to host **H2**
    - Often, routing and *forwarding* are used interchangeably in writings about the network layer, but more specifically, forwarding refers to a *local* process and routing refers to a *network-wide* process



Local forwarding table

header	output
0100	3
0110	2
0111	2
1001	1

Figure 4.2 Routing algorithms determine values in forward tables

-
  - The above depicts a *forwarding table*
  - A router will look at certain values arriving in the header of a packet, and based on those values will forward it to the appropriate outgoing link
  - In this diagram, a packet containing the value `0110` arrives at the router, and the forwarding table tells the router to send that packet to output link 2
  - The composition of forwarding tables is done on a network-wide scale by using network routing algorithms to allow each and every router to compute adequate values for their forwarding tables
- 4.1.2 Network Service Model
  - So, having discussed the basics of network layer functionality, we can now consider the different services that the network layer is capable of providing
    - *Guaranteed Delivery*, meaning that when a packet is sent, it is guaranteed to eventually arrive at its destination
    - Guaranteed Delivery with *Bounded Delay*, meaning guaranteed delivery within a specified window of time

        - *In-Order Packet Delivery*, meaning the packets will arrive at their destination in the same order in which they were sent
        - *Guaranteed Minimal Bandwidth*
        - *Security*, since the network layer can encrypt datagrams at the source and decrypt them at the destination, which provides confidentiality to transport layer segments
    - It is important to note that this is just a *partial* list of potential network layer functionality, and in reality there are many more variations

4.2: What's Inside a Router?

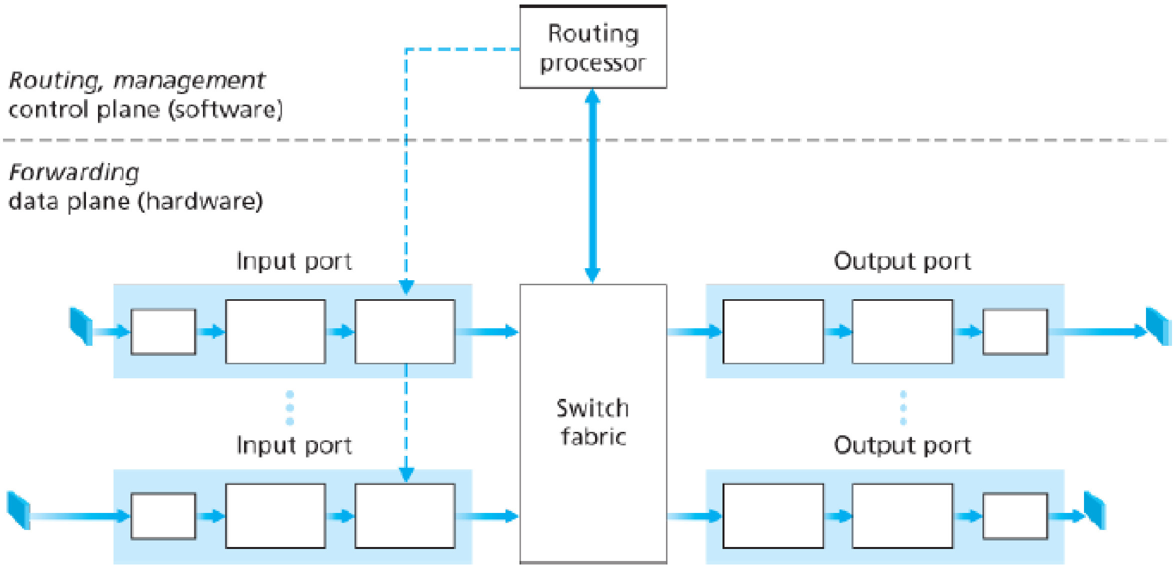


Figure 4.4 Router architecture

-
    - From this high-level diagram of the inner workings of a router four main components can be identified
        - ***Input Ports***, which perform several key functions, including
            - Performing the *physical layer* function of terminating an incoming physical link at a router
            - They also perform *link layer* functions needed to interoperate with the link layer on the other side of the incoming link
            - Most importantly, a *lookup function* is also performed in which the forwarding table is consulted to determine the appropriate output link to which to send a packet through the switch fabric

    - ***Switching Fabric***, which is a fabric that connects a router's input ports to its output ports
    - ***Output Ports***
      - It performs functions very similar to the input port but in an outgoing direction
      - When a link is *bidirectional*, an output port will usually be paired with the input port for that link on the same line card
    - ***Routing Processor***, which performs *control plane* functions such as
      - Executing routing protocols
      - Maintaining routing tables
      - Computing a router's forwarding table
  - These components are almost *always* implemented in hardware
  - The data plane operates at the nanosecond time scale, whereas the control plane operates at the millisecond or second time scale, and is thus usually implemented in software
  - Let us consider two types of information that could be used for forwarding of information over a network
    - *Destination-Based Forwarding*, where a router forwards information based on the most efficient possible route to its specified destination
    - *Generalized Forwarding*, where a router uses a set of predetermined rules to direct traffic regardless of a packet's specified destination
- **4.2.1: Input Port Processing and Destination-Based Forwarding**

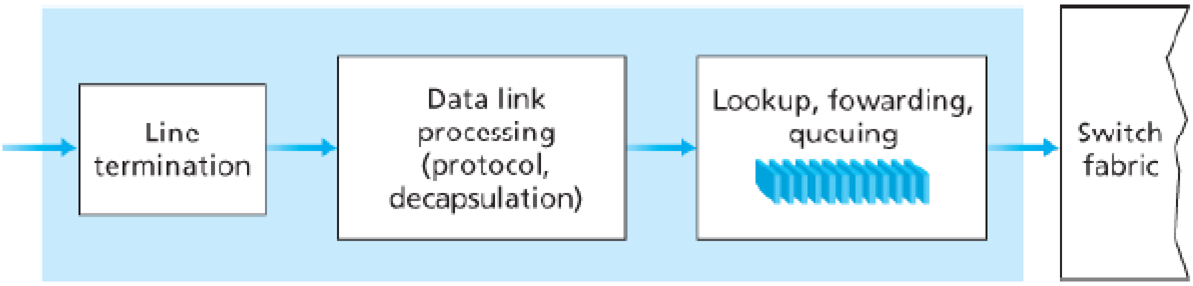
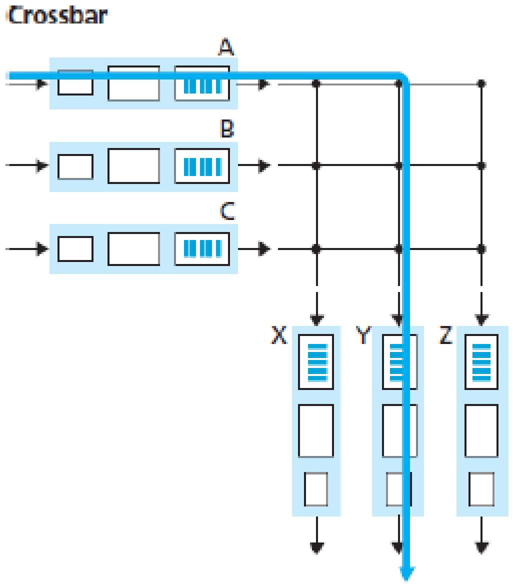

Line termination
Data link processing (protocol, decapsulation)
Lookup, fowarding, queuing
Switch fabric

Figure 4.5 Input port processing

-

    - Here we can see a more detailed view of the input port processing described earlier and shown in Figure 4.4
    - The forwarding table for an input link will either be computed by the routing processor or is received from a remote *SDN controller*
    - The forwarding table is copied from the routing processor to each input line card such that forwarding decisions can be made locally at each input port without the need to consult the routing processor
    - Using a forwarding table, the operation of lookup seems fairly simple, but at gigabit transmission rates, these operations must be performed in nanoseconds
        - To achieve this, not only is lookup performed in hardware, but fast lookup algorithms are used
    - Once the lookup has determined the appropriate output port for a packet, the packet can then be sent to the switching fabric
    - Packets can be temporarily blocked from entering a busy switching fabric and queued to enter at a later time
    - Apart from lookup, input processing is also made up of
        - Physical and link layer processing
        - Packet version number, checksum, and time-to-live bust be checked and the latter two rewritten
        - Counters used in management of networks must be updated
- **4.2.2: Switching**
    - The switching fabric is at the heart of the router, and is the thing that actually forwards packets
    - There are a number of different ways in which switching can be accomplished
        - *Switching via memory*
            - This operates similar to a traditional computer system
            - The routing processor is tasked with switching and input and output ports function as traditional I/O devices in a traditional OS

        - Headers are copied into memory such that the CPU (routing processor) is able to facilitate switching
        - Because only one memory read/write can be done at a time over the shared system bus, two packets cannot be switched simultaneously under a memory switching model
        - In some modern routers, this processing is done on individual line cards which thus resemble shared-memory multiprocessing systems
    - *Switching via a bus*
        - In this approach, the routing processor is not involved as the input port pre-pends a switch-internal header to the packet indicating the appropriate output port
        - All output ports will receive this packet, but only the one matching the packet's header will keep it
        - Because all packets must travel over a shared system bus, switching speed is limited to the bus speed
    - *Switching via an interconnection network*
        - Using a more sophisticated system such as this is a way to overcome the limitations presented by the previous two models



        -
- 4.2.3: Output Port Processing

- Output port processing takes packets that have been stored in the output port's memory and transmits them over the output link, which includes
  - *Selecting* and *de-queueing* packets for transmission
  - Performing the necessary physical and link layer operations needed in transmission
- The below illustrates a basic view of output port processing

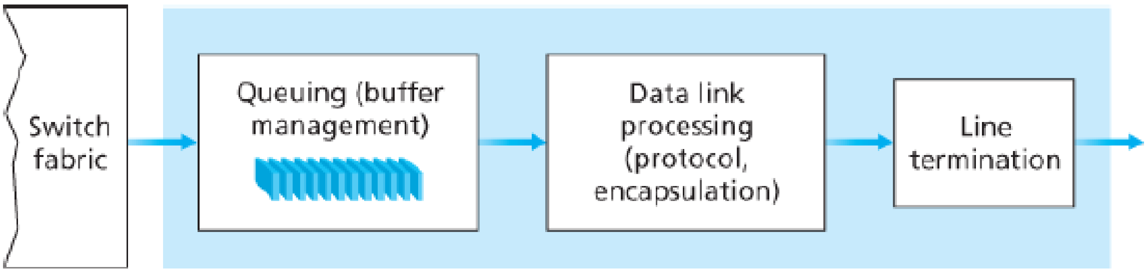

Switch fabric
Queuing (buffer management)
Data link processing (protocol, encapsulation)
Line termination

Figure 4.7 Output port processing

-

- **4.2.4: Where Does Queueing Occur?**
  - Considering the models discussed in input and output processing, it is clear that packet queues could form in either side of a router's links
  - When packet queues become large enough, they can eventually exhaust the memory of a router and cause *packet loss*
  - Assume R_{line} is the identical transmission rate of N input links and N output links, and R_{switch} is the switching fabric transfer rate or the rate at which packets can be moved from input to output port
  - If $R_{switch} \geq N * R_{line}$, then only negligible queueing will occur
  - However, if $R_{switch} < N * R_{line}$, then packet queueing will occur at the *input ports*
    - In order to illustrate the consequences of this type of queueing, we must assume that all link speeds are identical, that input and output speeds are the same, and that packets are serviced by the switching fabric in a first-come first-served manner
    - In this model, multiple packets can be transferred in parallel, as long as their desired output link is different
    - However, packets destined for the same output link must wait

    - In an input queued switch, however, it is possible for a packet to block the progression of another at the head of the switching fabric despite there being no competition for the blocked packet's desired output port
    - This phenomenon, called *head-of-the-line (HOL) blocking* and is illustrated in the below diagram

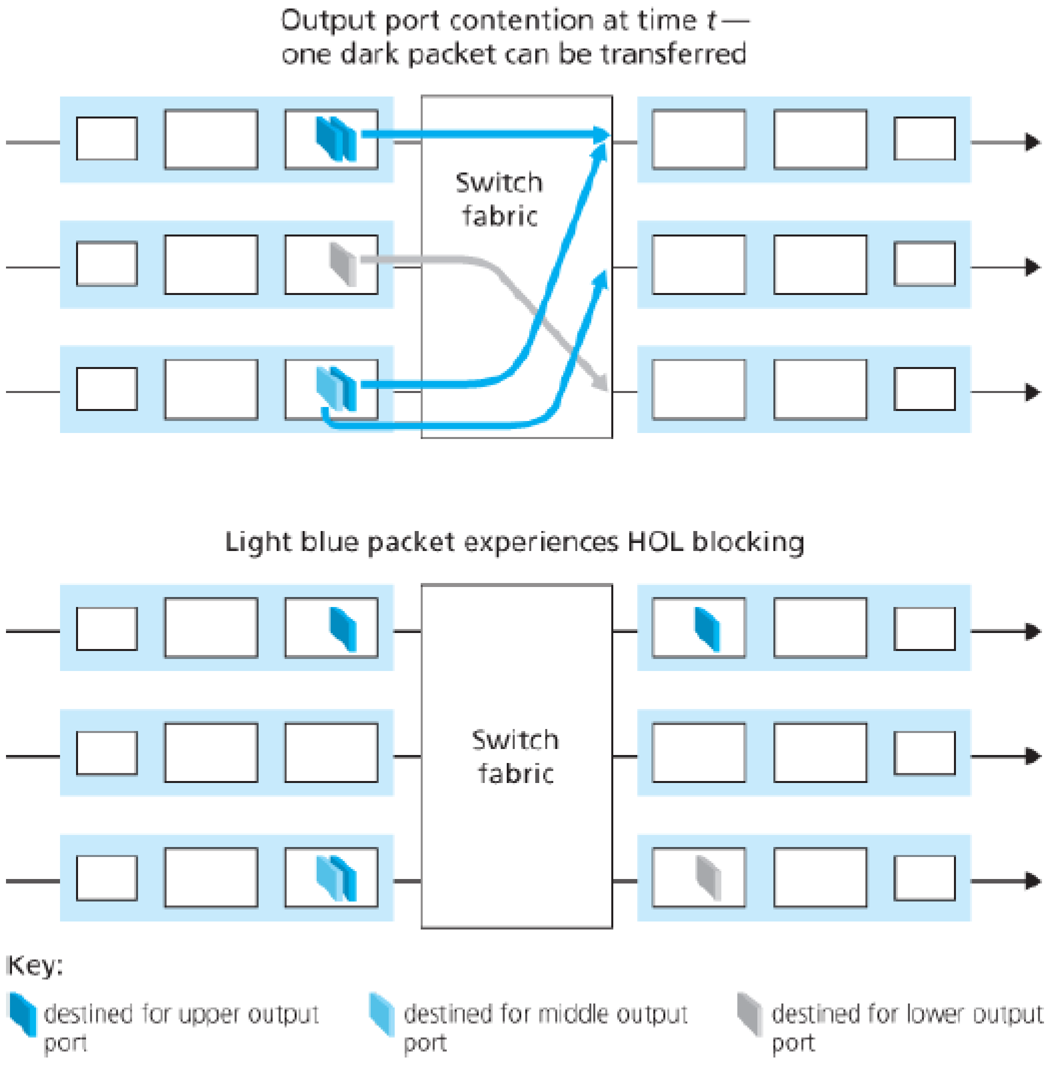

Output port contention at time t—
one dark packet can be transferred
Switch
fabric
Light blue packet experiences HOL blocking
Switch
fabric
Key:
destined for upper output port
destined for middle output port
destined for lower output port

Figure 4.8 HOL blocking at and input-queued switch

    -

  - Even if $R_{switch} \geq N * R_{line}$, since multiple packets can arrive at an output port in the time it takes for one packet to be transmitted, queuing can clearly occur at output links as well
    - If there is not sufficient memory to buffer an incoming packet, a decision must be made to either
      - *drop* the arriving packet

        - *remove* one or more already-queued packets to make room for the new packet

    - Sometimes packets are dropped before a buffer is totally full to indicate congestion on a network

    - There are a number of proactive packet dropping and marking policies known as *active queue management (AQM) algorithms*

    - According to recent theoretical and experimental efforts, the amount of buffering needed in an output port is $B = RTT * \frac{C}{N}$ where

        - C is the link capacity

        - N is the number of TCP flows passing through a link

        - RTT is the average round trip time

- **4.2.5: Packet Scheduling**

  - When discussing packet scheduling, as with CPU scheduling, there are a number of different ways in which it might be implemented, and these various ways have their own benefits and detriments

  - *First-in-First-out (FIFO)*

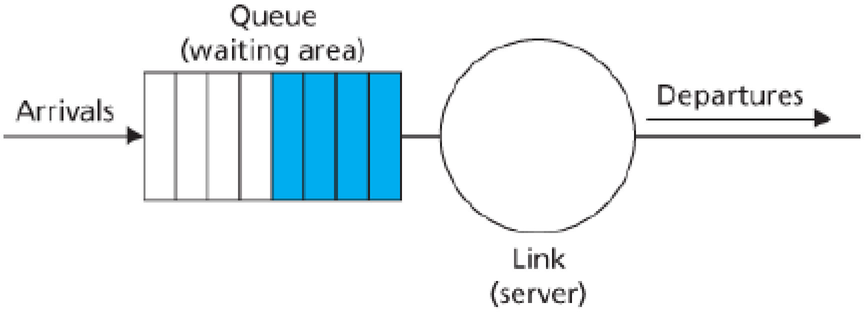
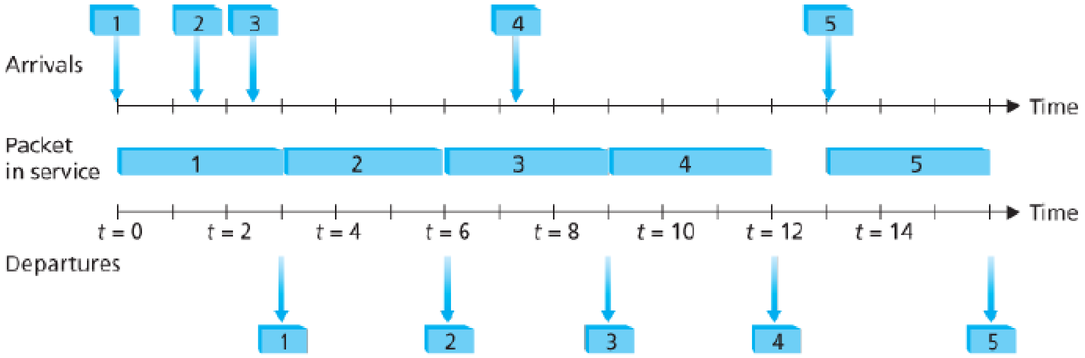


Figure 4.10 FIFO queueing abstraction

    -

    - In this structure, packets are serviced in the order in which they arrive at the router's output link



- Figure 4.11 The FIFO queue in operation

- *Priority Queue*
  - In a priority queuing structure, packets are assigned a priority class upon arriving at the output link
  - Typically, each priority class has its own queue
  - There is *preemptive* and *nonpreemptive* priority queuing where preemptive queuing allows a high priority packet to interrupt the transmission of a lower priority packet
  - The below are basic representations of priority queueing

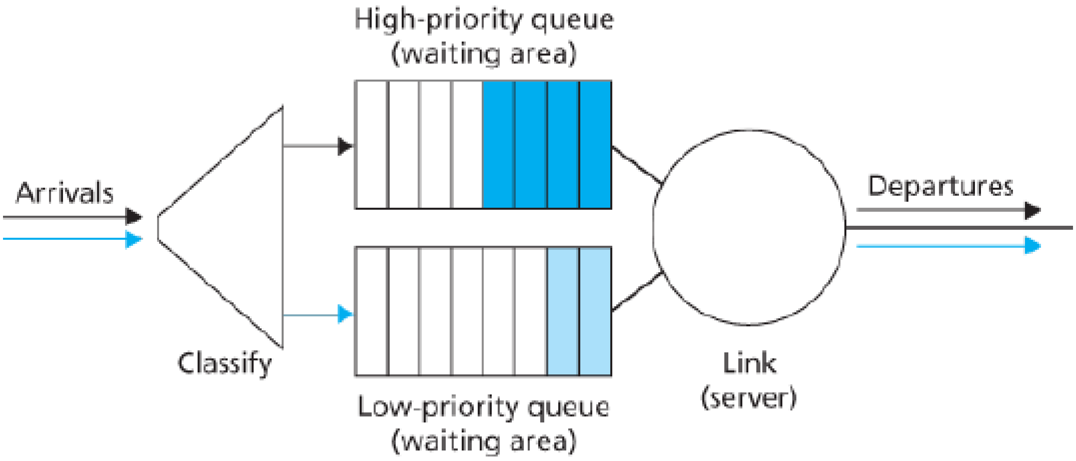


Figure 4.12 The priority queueing model

-

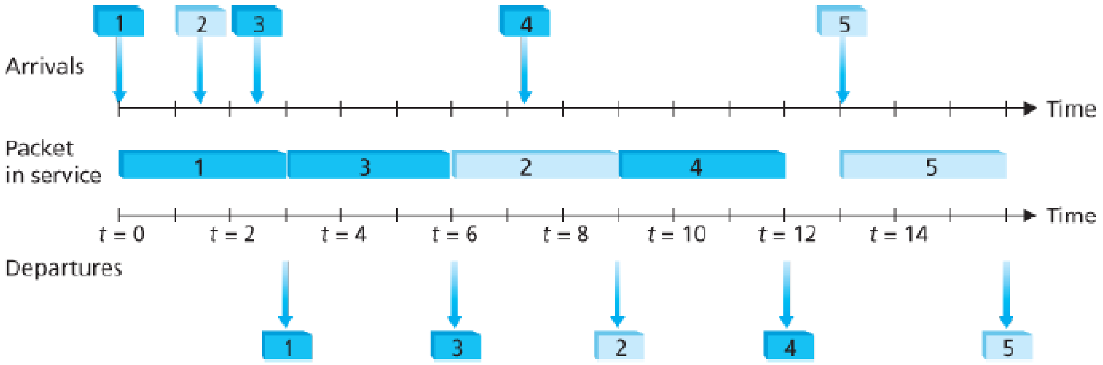


Figure 4.13 The priority queue in operation

- Round Robin and Weighted Fair Queueing (WFQ)
  - Similar to priority queueing, packets are separated into classes as they arrive at on output link
  - However, rather than there being a strict service priority, the scheduler will *alternate* service between the classes
  - A *work-conserving queueing* discipline will never allow the link to remain idle whenever there are packets of any class that are queued for transmission
    - If it finds no queued packets in the current class, it will move to another and begin to check for queued packets
  - Below is an example of *two-class* round robin queueing

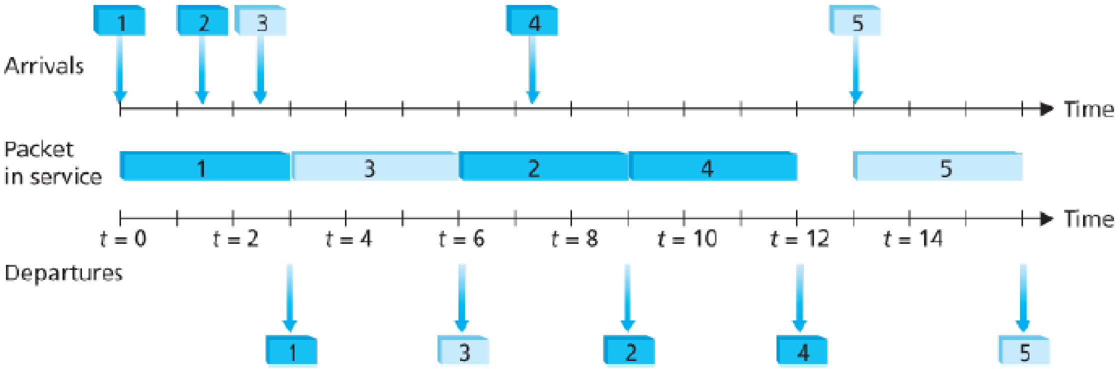


Figure 4.14 The two-class robin queue in operation

  - A generalized form of round robin queueing is the weighted fair queueing discipline
  - In weighted fair queueing, the different classes will be cycled between, but the time intervals during the which a class, i, will be serviced is determined by a

weight, w_i

  - A class, i, will then be guaranteed to receive $\frac{w_i}{\sum w_j}$ fraction of service where $\sum w_j$ is the sum taken over all classes that have packets queued for transmission

- Therefore, for a link with transmission rate, R, under WFQ, a class i will always achieve a throughput of at least $R * \frac{w_i}{\sum w_j}$
- Below is a representation of weighted fair queuing

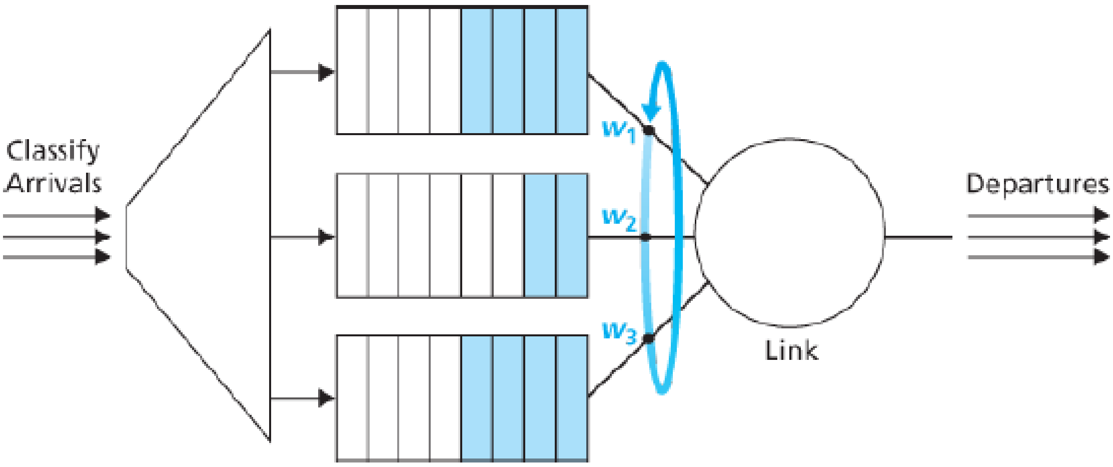


Figure 4.15 Weighted fair queueing

-