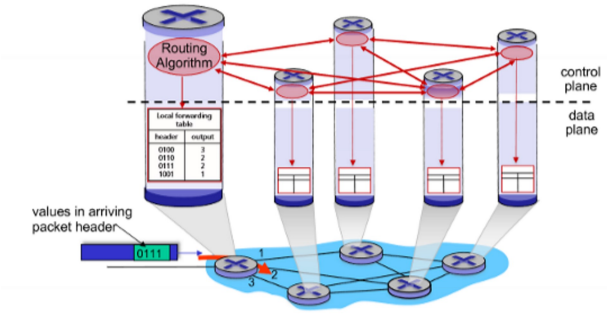
Chapter 4 Network Layer: Data Plane

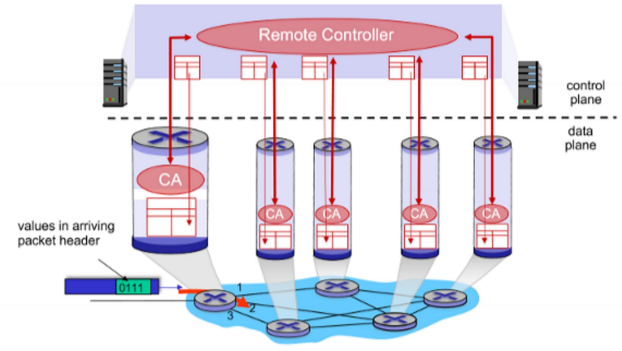
Wednesday, April 7, 2021 5:49 PM

4.1.1 Forwarding and Routing: The Data and Control Planes

- **Forwarding**
  - router moves packet to appropriate output link
  - router transferring a packet from input link interface to appropriate output link interface
- **Routing**
  - network layer must determine the route or path taken by packets as they flow from sender to receiver
  - routing algorithms calculate paths
  - network wide process to determine end to end paths that packets take
    - **forwarding table**
      - router forwards a packet by examining value of header, index into fwd table
- **Data Plane**
  - local, per router function
  - determines how datagram arriving on router input port is forwarded to router output port
- **Control Plane**
  - network wide logic
  - determines how datagram is routed among routers along end-end path from source host to dest host
  - **Traditional**
    - routing algorithm runs in every router



  - **Software defined networking SDN Approach**
    - implemented in remote servers
    - each router has routing component that communicates with routing componenet of other routers
    - controller computes forwarding tables and interats with routers



4.1.2 Network Service Model

- defines characteristics of end-to-end delivery of packets between sending and receiving hosts
- **Network layer services:**
  - **Guaranteed delivery**
    - packet sent by host will arive at destination host
  - **Guaranteed delivery with bounded delay**
    - guarantees packet delivery, and delay bound (ex. 100 msec)
  - **in order packet delivery**
    - guarantees that packet arrives at dest in order sent
  - **Guaranteed minimal bandwidth**

        - emulates behavior of transmission link of specified bit rate (ex ` Mbps) bw sending and receiing hosts
        - if host transmits bits at rate below spec bit rate, all packets are delivered
    - **Security**
        - network layer could encrypt datagrams at source and decrypt at dest

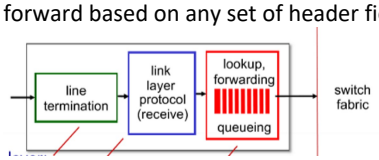
- **Network layer service model**

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no
ATM	Constant Bit Rate	Constant rate	yes	yes	yes
ATM	Available Bit Rate	Guaranteed min	no	yes	no
Internet	Intserv Guaranteed (RFC 1633)	yes	yes	yes	yes
Internet	Diffserv (RFC 2475)	possible	possibly	possibly	no

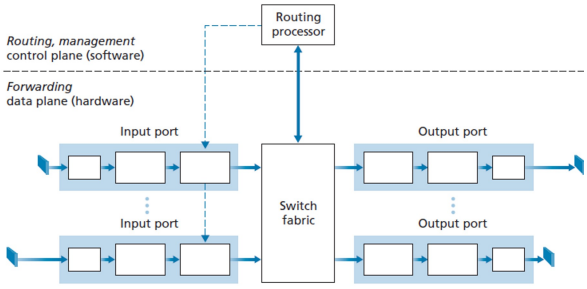
    - **Best effort service**
        - simplicity of mechanism
        - sufficient provisioning of bandwidth - real time application good enough most of the time
        - replicated, application layer distributed services, services provided from multiple locations
        - congestion control of elastic services

4.2 Router Architecture

- **Input ports**
    - terminating incoming pysical link at a router
    - control packets are fwd from input port to routing processor
    - **decentralized switching**
        - using header field value, lookup output port using fwd table in input port memory
        - complete input port processing at line speed
        - **import port queuing**
            - if datagram arrive faster than forwarding rate into switch fabric
        - **destination based forwarding**
            - forward based only on destination IP address
        - **generalized forwarding**
            - forward based on any set of header field value



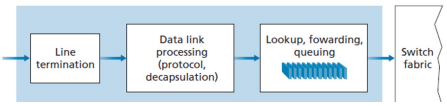
- **Switching fabric**
    - connects router input ports to output port, network inside network router
- **output ports**
    - stores packets received from switching fabric
    - transmits packets on outgoing link with necessaryi link layer functions
- **routing processor**
    - performs control plane functions
        - in SDN, routing processor responsible for communicating with remote controller to receive fwd table entries computed by remote controller



re 4.4 ♦ Router architecture

4.2.1 Input Port Processing

- output port with incoming packet is to be switched based on packet destination address



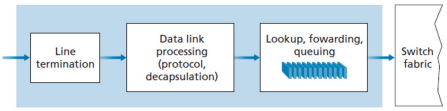


Figure 4.5 ♦ Input port processing

- **Longest prefix matching**
  - for given dest addr, use longest address prefix that matches dest addr

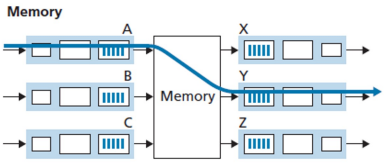
Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
Otherwise	3

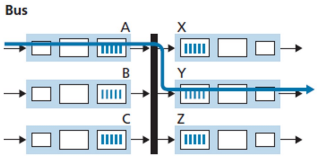
Prefix	Link Interface
11001000 00010111 00010	0
11001000 00010111 00011000	1
11001000 00010111 00011	2
Otherwise	3

  - router matchees a prefix of packet's destination addr
    - ex. for packet dest addr 11001000 00010111 00010110 10100001
    - 21 bit prefix of this addr matches 1st entry
  - if multiple matches, router uses **longest prefix matching rule**
    - finds longest matching entry in the table and forwards packet to link
  - often performed using ternary content addresable memories (TCAMs)
    - **content addressable**
      - present address to TCAM: retrieve address in 1 clk cycle regardless of table size
      - Cisco Catalyst routers and switches can holda million TCAM fwd table entries

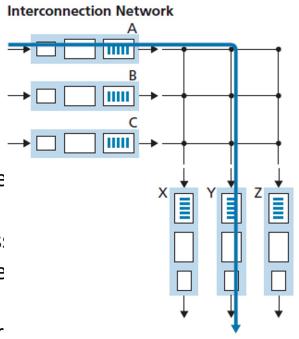
4.2.2 Switching Fabrics

- transfer packet from input link to appropriate output link
- **switching rate:** rate at which packets can be transferred from inputs to outputs
- **Types:** Memory, bus, interconnection network

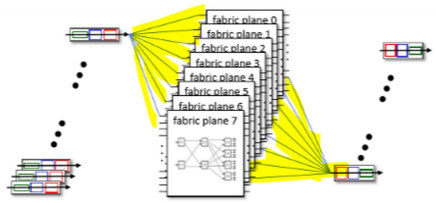


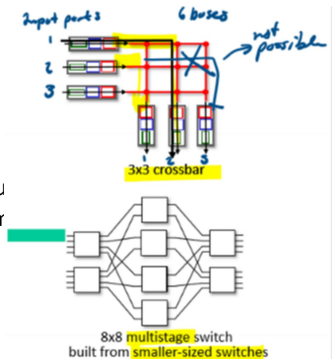


- **Switching via memory**
  - traditional computers with switching under CPU
  - packet copied to system memory
  - speed limited by bandwidth (2 bus crossings
- **Switching via bus**
  - datagram from input to output port memory via shared bus
  - **bus contention:** switching speed limited by bus bandwidth
  - input port pre-pend switch header to packet, indicates local output port transferred, and transmitting packet onto the bus
  - all output ports receive packet, but only port matching label keeps packe
  - label is removed at output port, only used within switch to cross bus
  - if multiple packets arrive to router, each at diff ports, onely one can cros
  - switching speed of router limited to bus speed bandwidth of single share



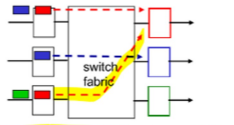
- **Switching via interconnection network**
  - crossbar, clos network, nets, developed to connect processors in multipr
  - **multistage switch:** nxn switch from multiple stages of smaller switches
  - **exploiting parallelism:**
    - fragment datagram into fixed length cells on entry
    - switch cells through fabric, reassemble datagram at exit
    - scaling using multiple switching planes in parallel
    - speedup, scaleup via parallelism
  - overcome bandwidth limitation of single shared bus
  - corssbar switch is non blocking, packet will not be blocked form reaching output port as long as no other packet is being forwarded to that output port



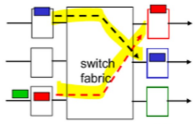


4.2.3 Output port processing

- **Input port queuing**
  - if switch fabric slower than input ports combined, queuing may occur at input qu
  - **HOL blocking:** queued datagram at front of queue prevents others in queue from

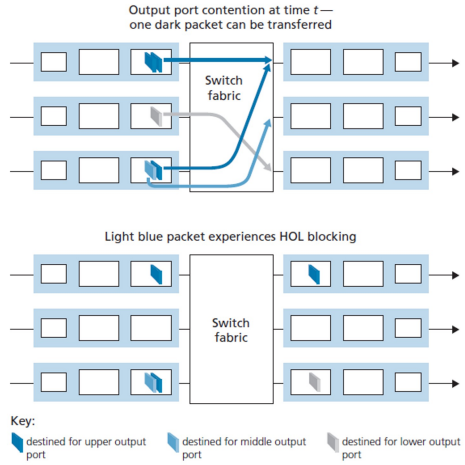


output port contention: only one red datagram can be transferred. lower red packet is blocked



one packet time later: green packet experiences HOL blocking

- **Output Queueing**
  - **Buffering**
    - required when datagrams arrive from fabric faster than link trans rate
    - datagram can be lost due to congestion, lack of buffers
  - **Scheduling discipline**
    - chooses among queued datagram for transmission
    - priority scheduling who gets best performance, network neutrality
- **How much buffering is needed**
  - average buffering = typical RTT * link capacity (C)
  - with N flows: buffering = RTT*C/sqrt(N)
  - too much buffering can increase delays
    - long RTTs
    - keep bottleneck link just full enough but no fuller



ure 4.8 ♦ HOL blocking at and input-queued switch

4.2.5 Packet Scheduling

- **Buffer management**
  - order which queued packets are transmitted over link
  - **Drop**
    - Tail drop: drop arriving packet
    - Priority: drop/remove on priority basis
- Decide which packet to send next on link
  - **FIFO:** first in first out

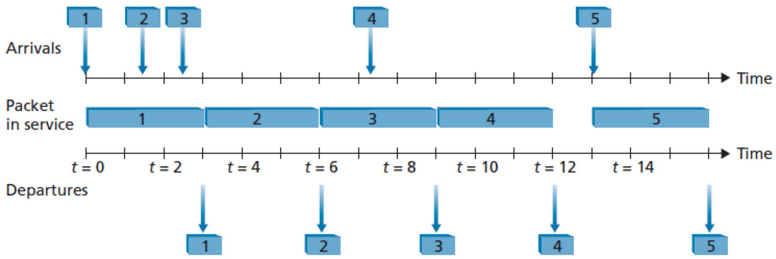


Figure 4.12 ♦ The FIFO queue in operation

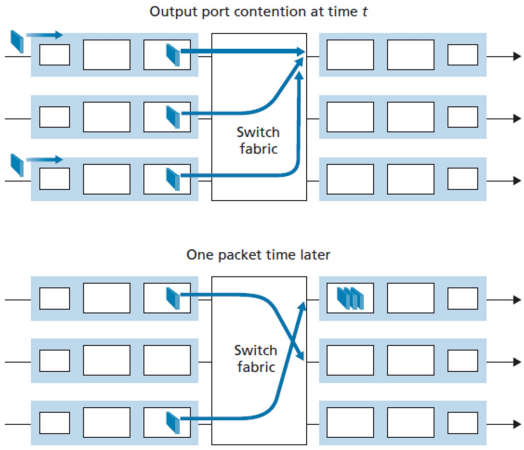
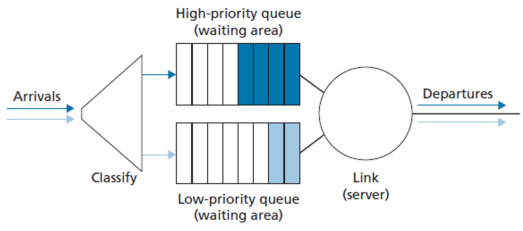
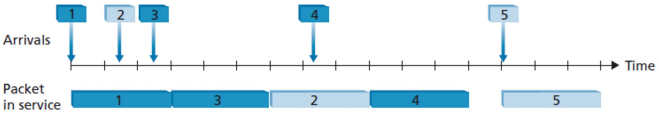


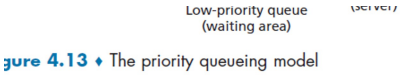
Figure 4.9 ♦ Output port queueing

  - **Priority scheduling**
    - arriving traffic classified, queued by class
    - send packet from highest priority queue that has buffered packets

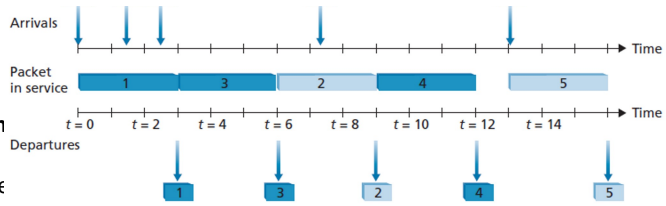


gure 4.13 ♦ The priority queueing model



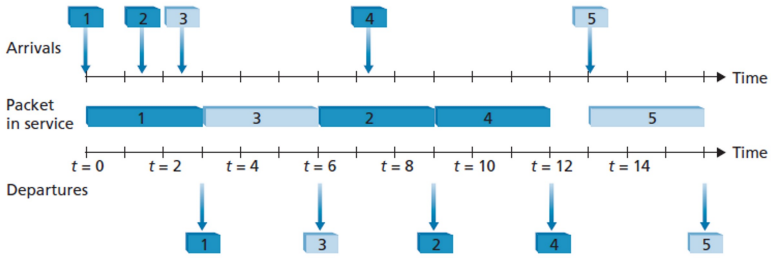


gure 4.13 ♦ The priority queueing model



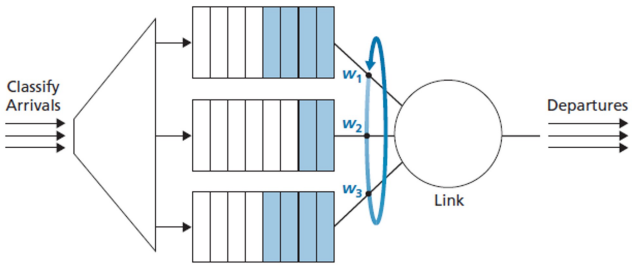
re 4.14 ♦ The priority queue in operation

- **Round Robin and Weighted Fair Queuin**
  - **RR**
    - arriving traffic classified, que
    - server repeatedly scans class turn



ure 4.15 ♦ The two-class robin queue in operation

  - **WFQ**
    - Generalized round robin
    - each class i has weight w, gets weighted amount of service in each cycle
    - minimum bandwidth guarantee



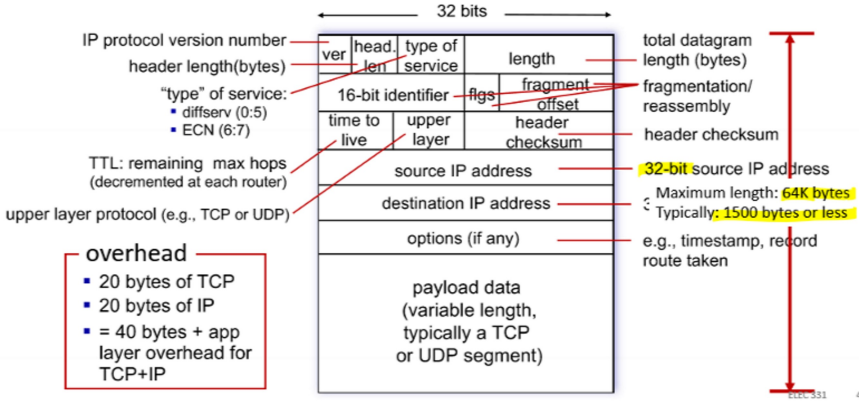
$$\frac{w_i}{\sum_j w_j}$$

igure 4.16 ♦ Weighted fair queueing

- **Network Neutrality**
  - how an ISP should share/allocation its resources (packet scheduling, buffer management, etc)
  - protecting free speech, encouraging innovation, competition
  - enforced legal rules and policies
  - **FFC Order on protecting and promoting open internet**
    - **No blocking**
      - shall not block lawful content
    - **No throttling**
      - shall not impair or degrade lawful internet traffic on basis of internet content, application, or service
    - **no paid prioritization**

4.3 Internet Protocol (IP): IPv4, Addressing, IPv6

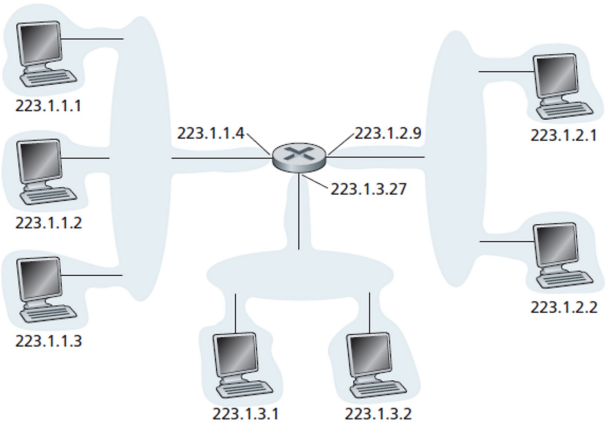
4.3.1 Ipv4 Datagram Format



- **IP address:** 32 bit identifier
- **Interface:** connection bw host/router and physical link

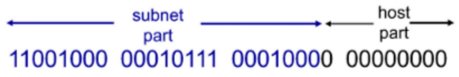
4.3.2 IPv4 Addressing

- IP requires each host and router interface to have its own IP
- 32 bits/4 bytes long, written in **dotted decimal notation**
  - each byte of address written in decimal form, separated by dot from other bytes in address
    - ex. 193.32.216.9 is 11000001 00100000 11011000 00001001
- portion of interface IP addr determiend by subnet to which it is connected
  - IP address form 223.1.1.xxx (24 first bits same)
  - these 4 interfaces interconnected to each other by no router network
- **Subnet**
  - network interconnecting 3 host interfaces and 1 router interface
  - IP addressing assigns address to subnet: 233.1.1.0/24
    - **subnet mask** /24 indicates leftmost 24 bits of 32 bit quantity defines subnet addr
    - **host part:** remaining low order bits
  - 223.1.1.0/24 subnet has 3 host interfaces (223.1.1.1, 223.1.1.2, and 223. interface (223.1.1.4)
  - any additional hosts attached to subnet required to have addr of form 223.1.1.xxx



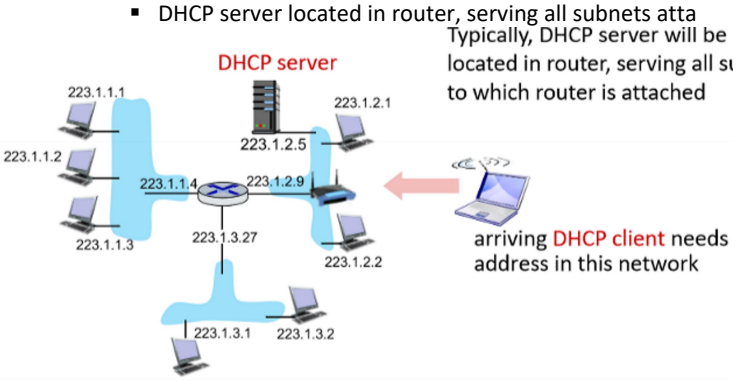
jure 4.18 ♦ Interface addresses and subnets

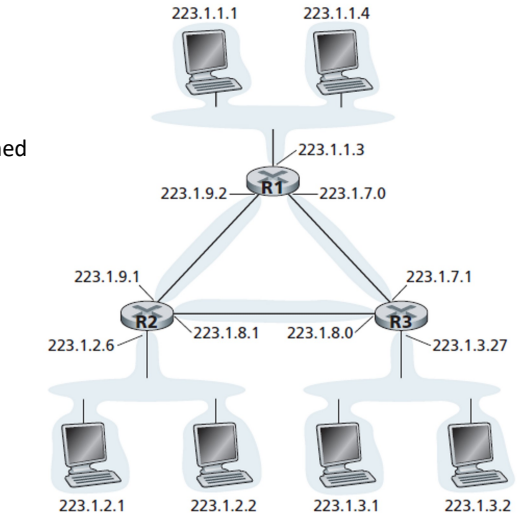
- **CIDR Classless InterDomain Routing**
  - subnet portion of address of arbitrary length
  - address format: a.b.c.d/x, x is # bits in subnet portion



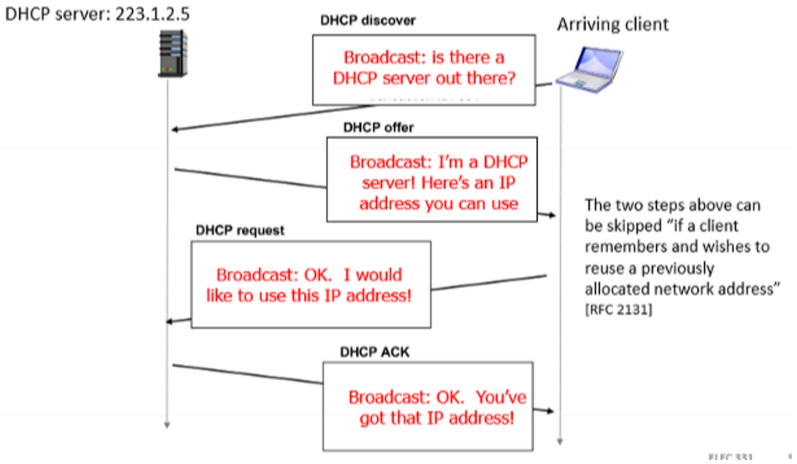
200.23.16.0/23

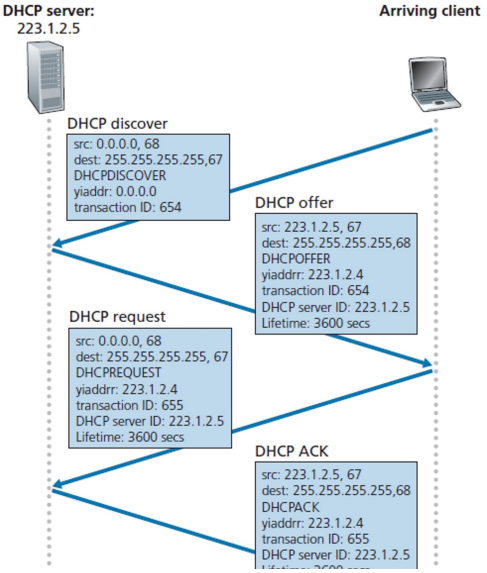
- **How does Host get IP address**
  - hard coded by sys admin in config file UNIX
  - **DHCP Dynamic Host Configuration Protocol**
    - dynamically get address from server
    - Host dynamically obtains IP address from network server when joined
      - can renew lease on address in use
    - DHCP server located in router, serving all subnets atta





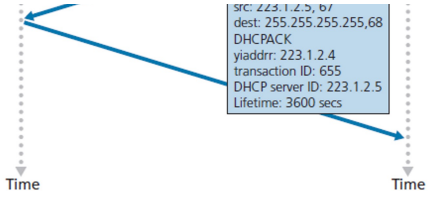
.20 ♦ Three routers interconnecting six subnets





  - DHCP can return more than IP address
    - address of first-hop router for client
    - name and IP of DNS server

- DHCP can return more than IP address
  - address of first-hop router for client
  - name and IP of DNS server
  - network mask
- **DHCP Process**
  - host use DHCP to get IP addr, addr of first hop router, addr of I
  - DHCP request msg in UDP, in IP, in Ethernet
  - Ethernet frame broadcast on Lan received at router running DH
  - Ethernet demux to IP demux to UDP demux to DHCP
  - DCP server forms DHCP ACK with client IP addr, IP addr of rist hop router, name and IP addr of DNS
  - encapsulate DHCP server reply fwd to client, demux up to DHCP at client
  - client knows its IP addr, name and IP addr of DNS, IP addr of first hop router



4.24 ♦ DHCP client-server interaction

- **How does Network get subnet part of IP address**
  - gets allocated portion of its provider ISP's address space

ISP's block 11001000 00010111 00010000 00000000 200.23.16.0/20

ISP can then allocate out its address space in 8 blocks:

Organization 0	11001000 00010111 00010000 00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010 00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100 00000000	200.23.20.0/23
...	....	....
Organization 7	11001000 00010111 00011110 00000000	200.23.30.0/23

- **Hierarchical Addressing**
  - efficient advertisement of routing information
  - use a single prefix to advertise multiple networks
  - **Route Aggregation**

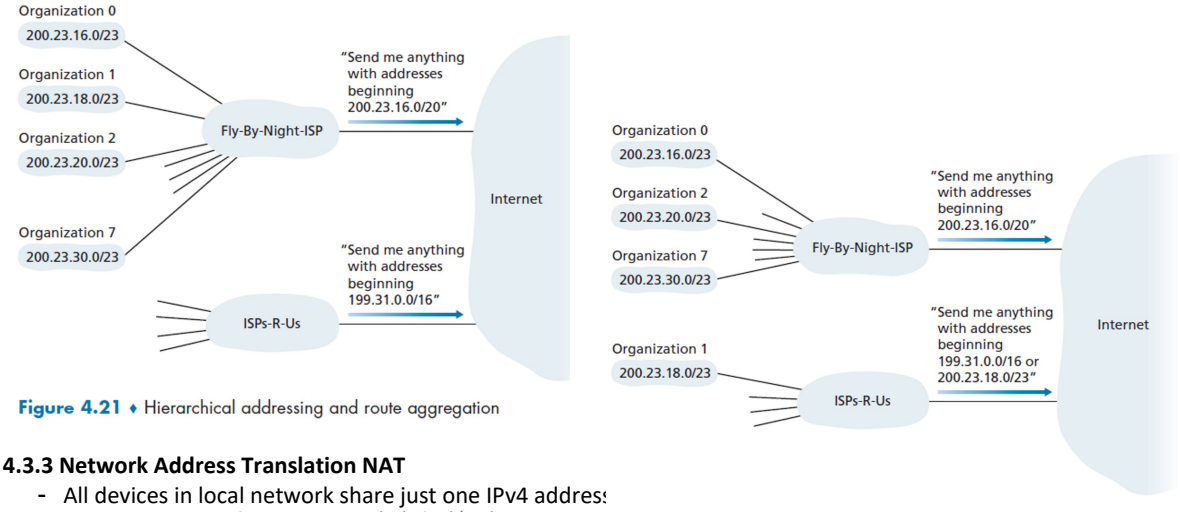


Figure 4.21 ♦ Hierarchical addressing and route aggregation

4.3.3 Network Address Translation NAT

- All devices in local network share just one IPv4 address

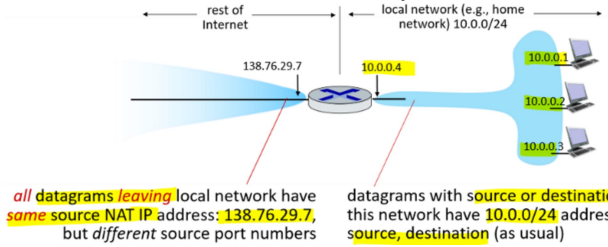
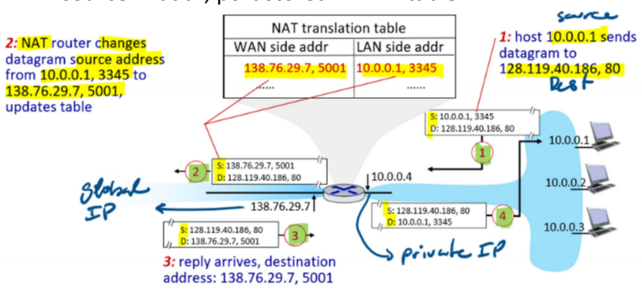


Figure 4.22 ♦ ISPs-R-Us has a more specific route to Organization 1

- All devices in local network have 32 bit addr in private IP addr space (10/8, 172.16/12, 192.168/16 prefixes) only be used in local network
- **Pros**
  - 1 IP addr needed from provider ISP for all devices
  - can change addr of host in local network
  - can change ISP without changing addr of devices in local
  - security - devices inside local net not directly addressable or visible
- **NAT Router**
  - replaces source IP addr, port of outgoing datagrams
    - remote servers respond using NAT IP addr, new port as dest addr
  - remember in NAT translation table every source IP addr, port to NAT IP addr, port translation pair

- replace NAT IP addr, port in destination fields of every incoming datagram with corresponding source IP addr, port stored in NAT table



-

4.4 Generalized Forwarding and SDN

- each router had forwarding table
  - match bits in arriving packet, take action
  - destination based forwarding: based on dest IP
  - generalized forwarding: many header fields can determine action

- **Flow Table Abstraction**
  - **Flow:** defined by header field value
  - **Generalized Forwarding:**
    - **Match:** pattern values in packet header fields
    - **Actions:** for matched packet: drop, fwd, modify matched packet or send to controller
    - **Priority:** disambiguate overlapping patterns
    - **Counters:** #bytes and #packets
  - **Flow Table**
    - set of header field values to which incoming packets will be matched
    - A set of counters updated as packets are matched to flow table entries
    - A set of actions to be taken when a packet matches a flow table entry

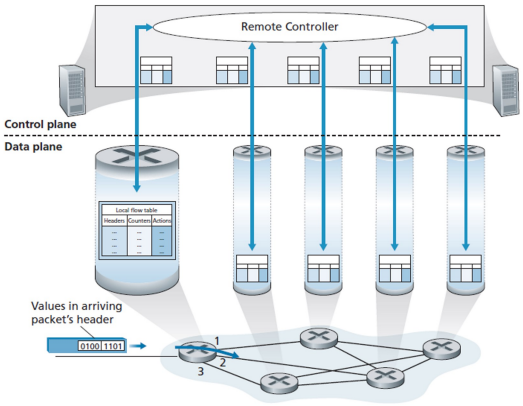
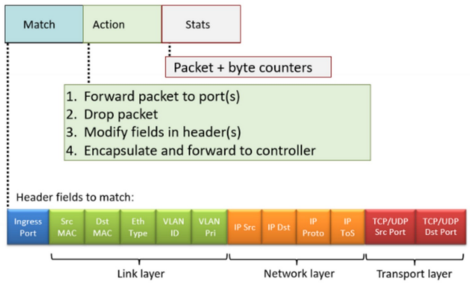
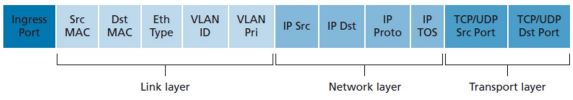


Figure 4.28 ♦ Generalized forwarding: Each packet switch contains a match-plus-action table that is computed and distributed by a remote controller

4.1.1 Match



- Flow table entry can have wildcards, ex. 128.119.*.*
- each flow table entry has associated priority
- not all fields in IP header can be matched, Open Flow does not allow matching on basis of TTL or length



igure 4.29 ♦ Packet matching fields, OpenFlow 1.0 flow table

4.4.2 Action

- each flow table entry has list of actions that determing processing to be applied to packet that matches a flow table entry
- if multiple actions, performed in order specified in list
  - **Forwarding**
    - forwarded to physical output port or broadcast over all ports
  - **Dropping**
    - flow table entry with no actions indicate a matched packet should be dropped
  - **Modify field**
    - values in 10 packet header field may be re written

Destination-based forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	*	*	*	*	*	51.6.0.8	*	*	*	*	port6

IP datagrams destined to IP address 51.6.0.8 should be forwarded to router output port 6

Firewall:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	*	*	*	*	*	*	*	*	*	22	drop

Block (do not forward) all datagrams destined to TCP port 22 (ssh port #)

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	*	*	*	*	128.119.1.1	*	*	*	*	*	drop

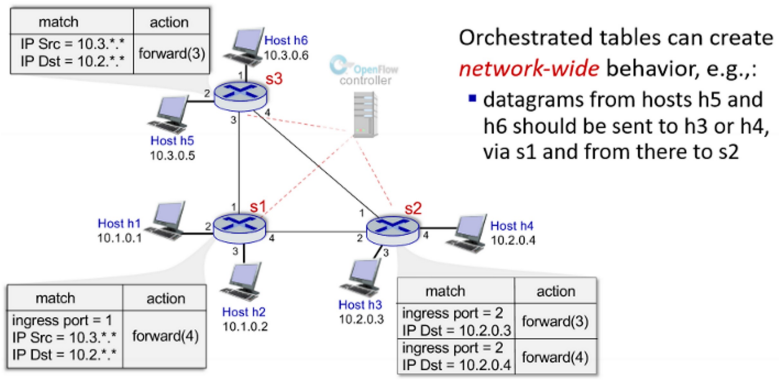
Block (do not forward) all datagrams sent by host 128.119.1.1

Layer 2 destination-based forwarding:

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	VLAN Pri	IP Src	IP Dst	IP Prot	IP ToS	TCP s-port	TCP d-port	Action
*	*	22:A7:23:11:E1:02	*	*	*	*	*	*	*	*	*	port3

layer 2 frames with destination MAC address 22:A7:23:11:E1:02 should be forwarded to output port 3

- **OpenFlow Abstraction**
  - **Router**
    - Match: longest dest IP prefix
    - Action: Fwd out a link
  - **Switch**
    - Match: dest MAC address
    - Action: fwd or flood
  - **Firewall**
    - Match: IP addr and TCP/UDP port num
    - Action: permit or deny
  - **NAT**
    - Match: IP addr and port
    - Action: rewrite addr and port



Orchestrated tables can create *network-wide* behavior, e.g.,:

- datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2