How to get IP address

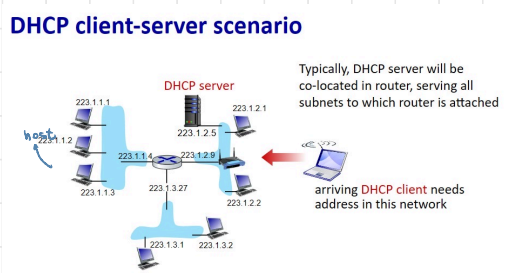
1. get host IP address
2. get network IP address

1. How does host get IP address

1. hard-coded by sysadmin in config file
2. DHCP

DHCP — Dynamic Host Configuration Protocol

↳ host dynamically obtains IP address from network server, when it joins network



pros

↳ can renew its lease on address in use (as IP is assigned for a certain amount of time)

↳ allows reuse of addresses (only holds addresses while connected/on)

↳ support for mobile users who join/leave network

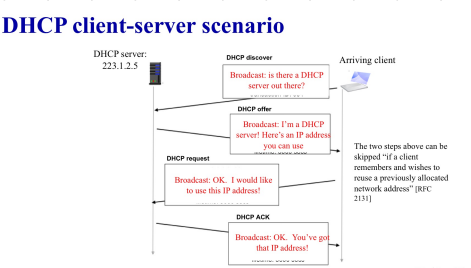
steps

↳ host broadcasts DHCP discover msg

↳ DHCP server responds with DHCP offer msg

↳ host requests IP address: DHCP request msg

↳ DHCP server sends address: DHCP ack msg



DHCP: more than IP address → NOT coming I think

↳ DHCP can return more than just allocated IP addresses on subnet

- ↳ address of first-hop router for client
- ↳ name and IP address of DNS server
- ↳ network mask (indicates network vs host portion of address)

Question # 1: **[CL0-1]** **[4 x 5 = 20 Points]**

Suppose a user Ali enters into the Class room E6 in CS block for CN Project demo. Assume that the IT department of the University is running various services to support network operations e.g. DHCP, DNS, Filestorage, and Web application (e.g. Flex, and SLATE). Describe in steps how Ali's laptop:

Part a) Proceeds to connect to the Internet.

Part b) After Project demo, Ali initiates a query to flex.nu.edu.pk to view his Project marks. Assume there is no Web cache server avialable on our institutional network. Which two techniques will be used by local DNS server to resolve the address of the webpage? Explain this with the help of a labelled diagram.

Part c) When obtaining the IP address for the host name flex.nu.edu.pk, assume the round trip time between local DNS server and DNS root server is 5RTT, between local DNS server and DNS TLD server is 4RTT, and between the clients and the local DNS server is 2RTT.

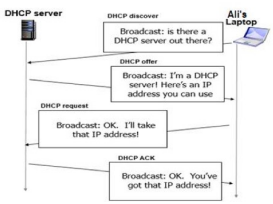
i) How long does it take for the Ali to obtain the IP address for flex.nu.edu.pk?

ii) After Ali, Bilal (on the same network) also wants to obtain the address for flex.nu.edu.pk. How long for the Bilal to obtain the IP address?

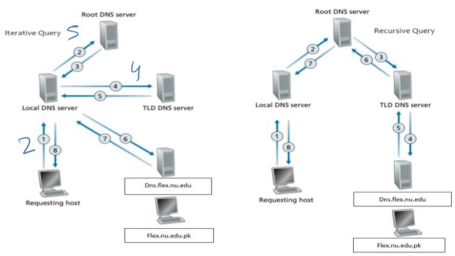
Part d) Suppose Ali is trying to access the hosted CN video lectures on Filestorage (Local server facility for accessing lecture material) and it has become bottleneck due to multiple users trying to access the hosted CN video lectures. What application level solution will you propose to overcome this problem? Explain.

Question # 1: **[CL0-1]** **[4 x 5 = 20 Points]**

Part a) Answer:Connects to especially setup-up WiFi connection. Set OS to get IP address using DHCP (50% marks if there are missing exchanges).



Part b) Answer:



Part c)

i) Answer: It takes 11RTT for the first client to obtain the IP address for google.com. 5+4+2 = 11

ii) Answer: It takes 2 RTT for the Bilal (as the IP address is already in local domain server) 2

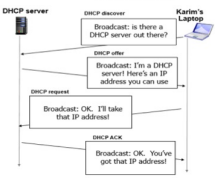
Part d) Answer:

This is a similar to the file upload to multiple peer we studied in the class.

- Client-server model need more computer, storage and network bandwidth as the number of users accessing the shared file folders grows to a large number (video lecture where size is ~1000s of MB).
- A P2P based application on each peer (like Bit torrent running on LAN) will allow a new users to use multiple copies of lectures those stored on different peers (as well as server copy) to take part of the download. Each peer PC will provide the bandwidth hence removing the bottleneck.

Question 2: **[5x2=10 points]**

Suppose a user Karim enters the university with his laptop. He is in room S2 in the CS block. Assume that the IT department of the University is running various services to support network operations e.g. DHCP, DNS, File storage, and web application (e.g. Flex). Describe in steps how Karim's laptop:



a) Connects to the Internet.

Solution:

i. Connect to WiFi and sets IP address manually by asking around (10% marks only).

ii. Connects to especially setup-up WiFi connection. Set OS to get IP address using DHCP (50% marks if there are missing exchanges).

b) Initiates a query to flex.nu.edu.pk.

Solution:

When karim initiates a query to flex.nu.edu.pk, the host makes DNS query, query is sent to its local DNS server which has local cache of recent name-to-address translation pairs. If not query will be forwarded to TLD Server and Root level servers.

OR

i. Do a DNS lookup to get IP address of flex domain.

ii. Show all HTTP exchanges to get the default home page.

c) Connects to another student Talha sitting in the same room (S2).

Solution:

He can connect to Talha by using ARP OR RARP protocol.

OR

i. Both you and Talha connected to LAN: Need a client-server application (Local web-server) or P2P application (Bit-Torrent client) running on Talha machine.

ii. Both you and Talha not connect to LAN: Make a Bluetooth, WiFi direct peer to peer (P2P) connection and use custom applications (such as smart wristwatch sync etc. and file transfer)

iii.

d) Connects to another instructor Farhan in the EE department.

Solution:

i. Cannot connect to Farhan using Bluetooth or WiFi direct as he is too far away.

ii. If Farhan is connect to LAN same as part (c) above option (i).

2. How does network get IP address → NOT coming I think

↳ 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

How does an ISP get block of addresses?

↓

ICANN → Internet Corporation for Assigned Names and Numbers

↳ allocated IP addresses through RRs → Regional Registries

↳ manages DNS root zone, including delegation of individual TLD management

↓ .com .edu

Q: are there enough 32-bit IP addresses?

- ICANN allocated last chunk of IPv4 addresses to RRs in 2011
- NAT (next) helps IPv4 address space exhaustion
- IPv6 has 128-bit address space

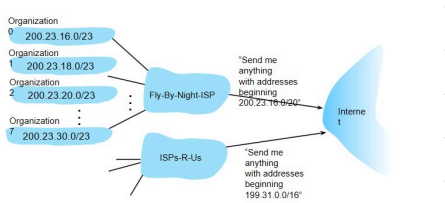
"Who the hell knew how much address space we needed?" Vint Cerf (reflecting on decision to make IPv4 address 32 bits long)

Network Layer: 4-62

Hierachical Addressing → NOT coming I think

Route Aggregation

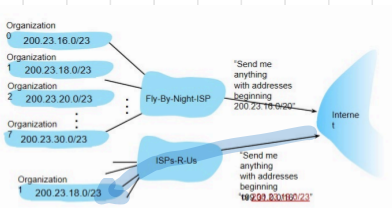
↳ it allows efficient advertisement of routing information



More specific routes

↳ Org 1 moves from Fly-By-Night ISP to ISPs-R-Us

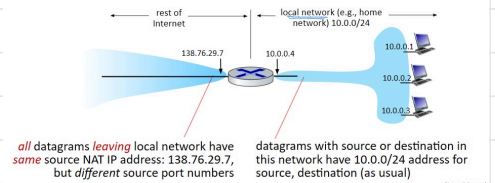
↳ ISPs-R-Us now advertises a more specific route to Org

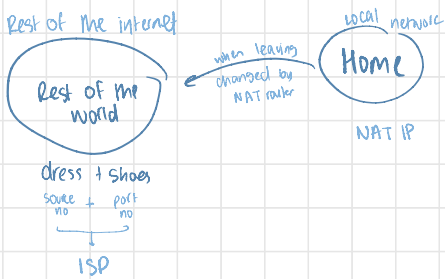


NAT: Network Address Translation

↳ all devices in local network share just one IPv4 addresses

↳ all devices in local network have 32-bit address in a private address space → (10/8, 172.16/12, 192.168/16 prefixes) that can only be used in local network





pros

↳ only '1' IP address needed from provider ISP (internet service provider) for all devices

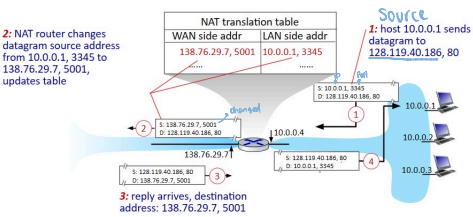
↳ can change addresses of host in local network w/o notifying outside world

↳ can change ISP w/o changing addresses of devices in local network

↳ security

↳ devices inside local network not directly addressable, visible by outside world

NAT: network address translation



steps

NAT router must

↳ outgoing datagrams

↳ replace S (source IP, port no) of every outgoing datagram to NAT (NAT IP, new port no)

↳ remote clients/servers will respond using NAT as destination address (NAT IP, new port no)

↳ remember in NAT table every S (source IP, port no) to NAT (NAT IP, new port no) pair

↳ incoming datagrams

↳ replace NAT (NAT IP, new port no) in destination fields of every incoming datagram with corresponding S (source IP, port no) stored in NAT table

NAT: network address translation

- NAT has been controversial:
  - routers "should" only process up to layer 3
  - address "shortage" should be solved by IPv6
  - violates end-to-end argument (port # manipulation by network-layer device)
  - NAT traversal: what if client wants to connect to server behind NAT?
- but NAT is here to stay:
  - extensively used in home and institutional nets, 4G/5G cellular nets

IPv6: motivation

- initial motivation
  - 32 bit IPv4 address space would be completely allocated
- additional motivation
  - speed processing: 40 byte fixed length header
  - enable diff network layer treatment of flows

IPv6 doesn't have compared to IPv4

- no checksum → to speed processing at routes
- no fragmentation/reassembly
- no options → all as upper layer, next header protocol at router

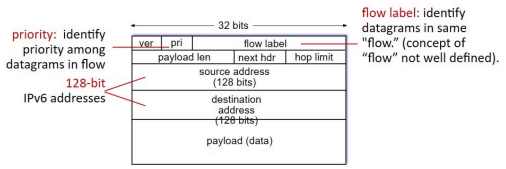
Transition from IPv4 to IPv6

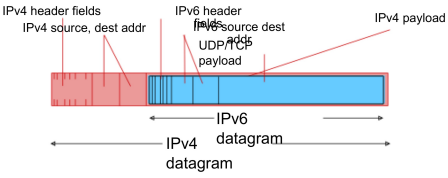
- not all routes can be upgraded simultaneously
  - no 'flag' days

Tunneling

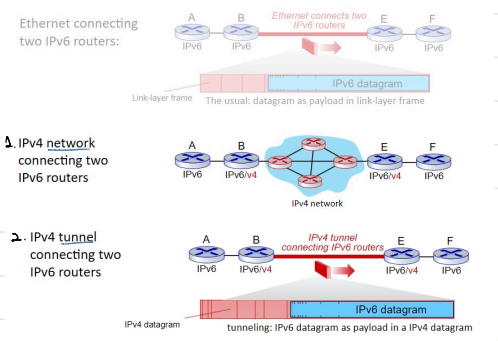
- allows network to operate with mixed IPv4 and IPv6 routes
- IPv6 datagram carried as payload in IPv4 datagram among IPv4 routes → packet within a packet

IPv6 datagram format

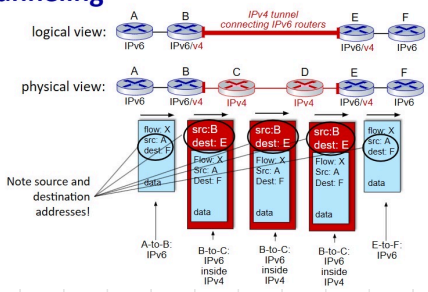




Tunneling and encapsulation



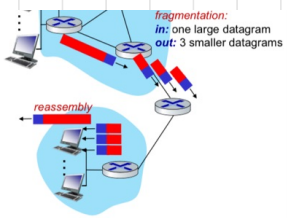
Tunneling



link-level frame

- different link types, different MTUs

- large IP datagram divided ("fragmented") within net
  - comes several
  - "reassembled" only at *destination*
  - IP header bits used to identify, order related fragments



63 of 64

Network Layer: 4-62

IP fragmentation, reassembly

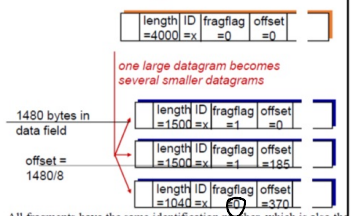
example:
- 4000 byte datagram
- MTU = 1500 bytes

Figure shows a datagram with a data size of 4000 bytes fragmented into three fragments. The bytes in the original datagram are numbered 0 to 3999.

The first fragment carries bytes 0 to 1479. The offset for this datagram is 0/8 = 0.

The second fragment carries bytes 1480 to 2959; the offset value for this fragment is 1480/8 = 185.

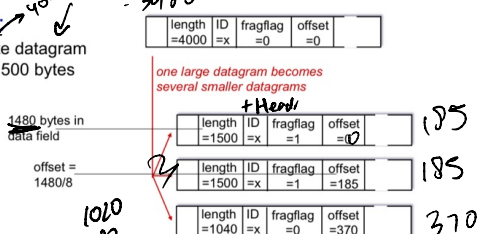
Finally, the third fragment carries bytes 2960 to 3999. The offset value for this fragment is 2960/8 = 370.



- All fragments have the same identification number, which is also the same as the original datagram.
- The offset is measured in units of 8 bytes.
- The maximum size of each fragment is the MTU minus the IP header size (Minimum 20 bytes and Maximum 60 bytes)
- If its value of flag field is 1, it means the datagram is not the last fragment; there are more fragments after this one. If its value is 0, it means this is the last or only fragment.

IP fragmentation/reassembly

example:
- 4000 byte datagram
- MTU =1500 bytes



Network Layer: 4-64

IP Fragmentation Example

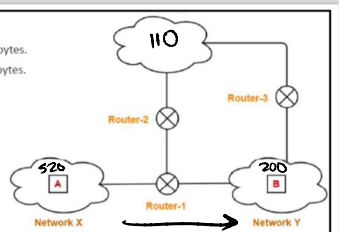
- There is a host A present in network X having MTU = 520 bytes.
- There is a host B present in network Y having MTU = 200 bytes.
- There exists a network Z having MTU = 110 bytes.
- Host A wants to send a message to host B.

Consider Router-1 receives a datagram from host A having-
Header length = 20 bytes
Payload length = 500 bytes
Total length = 520 bytes

Consider Router-1 divides the datagram into 3 fragments:

Maximum amount of data that can be sent in one fragment = 200 − 20 = 180 bytes.
Amount of data sent in a fragment must be a multiple of 8.
So, maximum data sent that can be in one fragment = 176 bytes.

Then,
First fragment contains the data = 176 bytes
Second fragment contains the data = 176 byes
Third fragment contains the data = 148 bytes



offset = data / 8 (no header)

Sum without header = data (without header) of fragments

Σ MTU - 20

Part (d)

ISPs are slowly changing their IP networks to IPv6. This means they have to interconnect between groups of routers running IPv4 with other groups running IPv6. Suppose a scenario shown in Figure 04, where IPv6 packets need to traverse an IPv4 tunnel. Describe how this tunnel is practically implemented.

Logical view

Figure 04

data / 8 : offset

Part (e)

A packet is to be forwarded to a network with MTU of 1500 bytes. The packet has an IP header of 20 bytes and data part of 3980 bytes. Find number of fragments using the given table.

	Total Length	Flag	Fragment offset
Original Packet	4000	1	0
Fragment 1	1500	1	497
Fragment 2	1500	1	994
Fragment 3	1020	1	
Fragment 4		0	

+20

3980 / 8 = 497

1480 / 1480 / 1020 — 2960 / 4440 — 2960 − 3980 : 1020

Question # 4: **CLO-03** **[05 * 08 = 40 Points]**

Part (a)

For the network shown in Figure 05, suppose that the desired forwarding behavior is that packets from h5 or h6 destined to h3 or h4 are to be forwarded from s3 to s1, and then from s1 to s2 (thus completely avoiding the use of the link between s3 and s2).

download. Each peer PC will provide the bandwidth hence removing the bottleneck.

Question 3: **[2x5=10 points]**

a) Why IP fragmentation was implemented in IPv4? Explain how IPv6 solves the issue for which IP fragmentation was designed in IPv4.

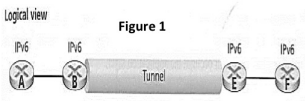
Solution:

- Each IP datagram need to fit in a frame at the source PC using its MTU size (e.g. ~1400). If it encounters a MTU size of < 1400 while passing through Internet (a mesh connection of routers) the router with lower MTU interface fragment IP datagram into two or more datagram. These datagrams travels the Internet without change (if they haven't encounter another lower MTU) and are assembled back by the destination network layer.
- IPv6 perform a MTU path discovery to the destination IP address before sending the packets and uses the lowest MTU.

b) ISPs are slowly changing their IP networks to IPv6. This means they have to interconnect between groups of routers running IPv4 with other groups running IPv6. Suppose a scenario shown in figure 1, where IPv6 packets need to traverse an IPv4 tunnel. How this tunnel is practically implemented?

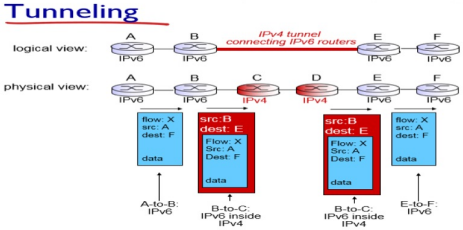
Logical view

Figure 1



Solution:

It is implemented using tunneling.IPv6 datagram carried as payload in IPv4 datagram among IPv4 routers as shown below.



Question 4: **[2x5=10 points]**

GENERALIZED FORWARDING

↳ each router contains a forwarding table → aka flow table

match PLUS action abstraction

↳ match bits in arriving packet

↳ take action

Generalized forwarding

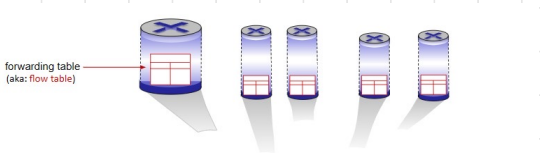
↳ many header files can determine action

↳ many action possible

- ↳ drop
- ↳ copy
- ↳ modify
- ↳ log packet

destination based forwarding

↳ forward based on destination IP address



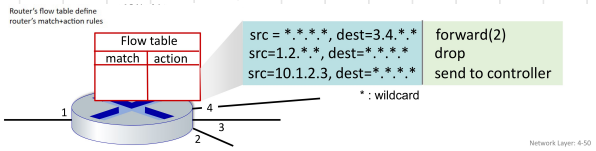
FLOW table Abstraction

↳ FLOW

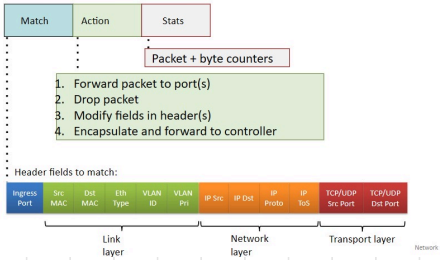
- ↳ defined by header feild values → in link, network, transport layer fields

↳ generalized forwarding

- generalized forwarding: simple packet-handling rules
  - match: pattern values in packet header fields
  - actions: for matched packet: drop, forward, modify, matched packet or send matched packet to controller
  - priority: disambiguate overlapping patterns
  - counters: #bytes and #packets



OpenFlow: flow table entries



OpenFlow: examples

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

Open Flow Abstraction

match + action

↳ abstraction unifies diff kinds of devices

Router

↳ match: longest destination IP prefix
↳ action: forward out a link

Firewall

↳ match: IP addresses and TCP/UDP port no.s
↳ action: permit / deny

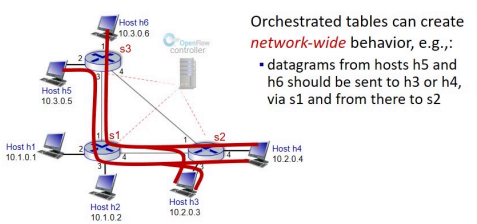
Switch

↳ match: destination MAC address
↳ action: forward or flood

NAT

↳ match: IP address and port
↳ action: rewrite address and port

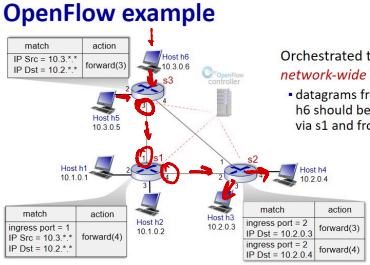
OpenFlow example



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

OpenFlow example



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

Generalized forwarding: summary

- "match plus action" abstraction: match bits in arriving packet header(s) in any layers, take action
- matching over many fields (link-, network-, transport-layer)
- local actions: drop, forward, modify, or send matched packet to controller
- "program" *network-wide* behaviors
- simple form of "network programmability"
- programmable, per-packet "processing"
- *historical roots:* active networking
- *today:* more generalized programming: P4 (see p4.org).

Part b) For the network shown in Figure 02, suppose that the desired forwarding behavior is that packets from Host H1 or Host H2 destined to Host H5 or Host H6 are to be forwarded from s3 to s1, and then from s1 to s2 (thus completely avoiding the use of the link between s3 and s2).

i) Write down the flow table entry for s3, so that datagram sent from Host H1 or Host H2 are forwarded to s1 over interface 3.

ii) Write down the flow table entry for s1, so that datagram arriving at port 1 of s1, from s3, are forwarded to s2 over outgoing interface 4.

iii) Finally write down the flow table entry for s2, for forwarding to required destinations.

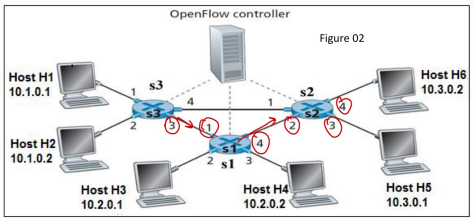


Figure 02

Part b) Answer:

i)

Match	Action
IP Src = 10.1.*.* ; IP Dst = 10.3.*.*	Forward(3)

ii)

Match	Action
Ingress Port = 1 ; IP Src = 10.1.*.* ; IP Dst = 10.3.*.*	Forward(4)

iii)

Match	Action
Ingress port = 2 ; IP Dst = 10.3.0.1	Forward(3)
Ingress port = 2 ; IP Dst = 10.3.0.2	Forward(4)

b) What is the key role of OpenFlow in SDN?

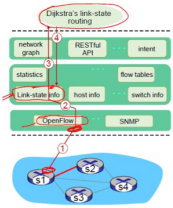
OpenFlow defines the communication protocol between SDN controllers and the data (forwarding) plane. It facilitate SDN controllers to send messages to switches consisting of queries, setting configuration parameters, modify state and even sending packets directly. Similarly, switch can send port and flow table status, or packets for processing at the controller.

c) Setup a flow table entry which prefers traffic from an IP telephone. Make suitable assumptions.

Match plus action type generalized forwarding. IP telephone traffic can be identified using Layer 2 attributes (MAC address or VLAN ID) or Layer 3 attributes (e.g. source or destination IP address).

```
Match [IP Scr=10.20.30.*, optionally IP Dest= 172.16.20.*]
Action [send to controller for type of service setting]
```

d) How SDN controller (explain in a step-by-step way) make use of different network application to provide link state routing in the SND setup? Explain.

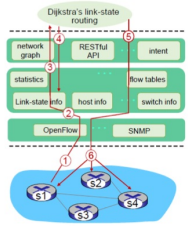


① S1, experiencing link failure uses OpenFlow port status message to notify controller

② SDN controller receives OpenFlow message, updates link status info

③ Dijkstra's routing algorithm application has previously registered to be called when ever link status changes. It is called.

④ Dijkstra's routing algorithm access network graph info, link state info in controller, computes new routes



⑤ link state routing app interacts with flow-table-computation component in SDN controller, which computes new flow tables needed

⑥ controller uses OpenFlow to install new tables in switches that need updating

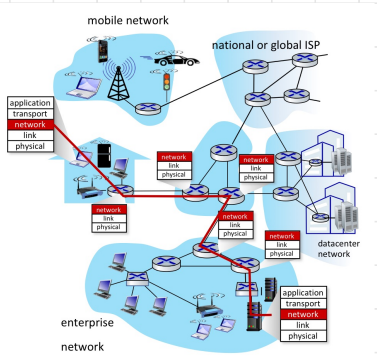
CHP 5 NETWORK LAYER: CONTROL PLANE

Routing Protocols

↳ to determine good paths (least cost, fastest, least congested) from sending host to receiving hosts, through network of routers

Path

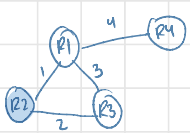
↳ sequence of routers packets traverse from given initial source host to final destination host



ROUTING ALGORITHM CLASSIFICATION

1. Global Routing Protocol

↳ all routers have complete
  ↳ topology
  ↳ link cost info
↳ link state algo
  ↳ e.g. Dijkstra

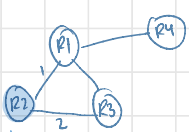


R2 knows link cost of everyones

VS

2. Decentralized Routing Protocol

↳ iterative process of computation
↳ exchanges info with neighbours
↳ routers only know link costs to attached neighbours
↳ distance vector algos
  ↳ e.g. Bellman Ford



R2 only knows link cost of neighbours R1, R3

1. Static Routing

↳ routes changes slowly over time

VS

2. Dynamic Routing

↳ routes change more quickly
  ↳ periodic updates
  ↳ in response to link cost changes

Link State Routing → Dijkstras Algo

↳ Greedy Algo

↳ computes least cost path from source to all other nodes

1. Centralized

↳ network topology

↳ link costs known to all nodes

↳ accomplished via "link state broadcast"

↳ all nodes have same info

2. Iterative

↳ after k iterations,

know least cost path to k destination

Dijkstras Algo

↳ doesn't work for -ve weights

↳ shortest path from s to any node

↳ Algo complexity $O(n^2)$

- each of *n* iteration: need to check all nodes, *w*, not in *N*
- $n(n+1)/2$ comparisons: $O(n^2)$ complexity
- more efficient implementations possible: O(*n*log*n*)

↳ Message complexity $O(n^2)$

- each router must *broadcast* its link state information to other *n* routers
- efficient (and interesting!) broadcast algorithms: O(*n*) link crossings to disseminate a broadcast message from one source
- each router's message crosses O(*n*) links: overall message complexity: $O(n^2)$

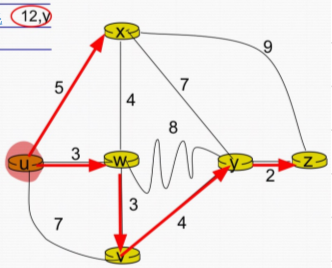
distance, vertex name

Step	N'	D(v)	D(w)	D(x)	D(y)	D(z)
		neighbouring nodes				
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

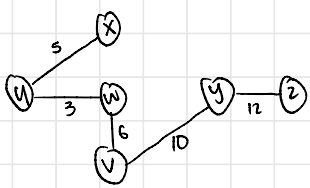
5 vertices so 6 columns

notes:

- construct shortest path tree by tracing predecessor nodes
- ties can exist (can be broken arbitrarily)

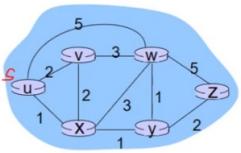


Shortest path tree from u



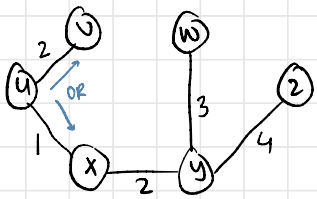
Dijkstra's algorithm: another example

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Activate Win
Go to Settings to

shortest path tree from u



resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

Distance vector routing → Bellman Ford Algo

↳ dynamic programming

1. Distributed, self stopping

distributed, self-stopping: each node notifies neighbors *only* when its DV changes

- neighbors then notify their neighbors – *only if necessary*
- no notification received, no actions taken!

2. Iterative, asynchronous

iterative, asynchronous: each local iteration caused by:

- local link cost change
- DV update message from neighbor

Distance vector algorithm:

each node:

wait for (change in local link cost or msg from neighbor)

↓

recompute DV estimates using DV received from neighbor

↓

if DV to any destination has changed, *notify* neighbors

Bellman-Ford Algo

↳ works for -ve weights

↳ relax each node: V-1 times

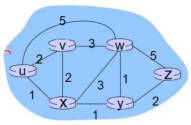
↳ find if negative cycle: if reduced ans when relaxed V times → means not optimal solution

$d_x(y)$: cost of least path - cost of path from x to y

$$d_x(y) = \min_v \{c(x,v) + d_v(y)\}$$

- $d_x(y)$: cost of least path from x to y
- $\min_v$: min taken over all neighbors v of x
- $c(x,v)$: direct cost of link from x to v
- $d_v(y)$: v's estimated least cost path cost to y

→ Bellman Ford equation



Neighbors of u:
$d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$d_u(z) = \min \{c(u,v) + d_v(z), \; c(u,x) + d_x(z), \; c(u,w) + d_w(z)\}$$
$$= \min \{2 + 5, \; 1 + 3, \; 5 + 3\} = 4 \text{ (via x)}$$

Node that achieves minimum is next hop in shortest path (via x above)
→ that goes in *forwarding table*

what?

↳ Neighbours of u = (v, x, w)

$c(u,v) = 2$ $\quad d_v(z) = 5$ $\quad u \to x \to y \to z$

$c(u,x) = 1$ $\quad d_x(z) = 3$ $\quad x \to y \to z$

$c(u,w) = 5$ $\quad d_w(z) = 3$ $\quad w \to y \to z$

} min distances to z

↳ $d_u(z) = \min\{[c(u,v) + d_v(z)], [c(u,x) + d_x(z)], [c(u,w) + d_w(z)]\}$ → BF eq

$= \min\{2+5, 1+3, 5+3\}$

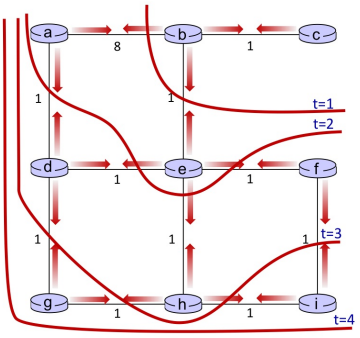
$= \min\{7, 4, 8\}$

$= 4$ via x

Distance vector: state information diffusion

Iterative communication, computation steps diffuses information through network:

- t=0 c's state at t=0 is at c only
- t=1 c's state at t=0 has propagated to b, and may influence distance vector computations up to **1** hop away, i.e., at b
- t=2 c's state at t=0 may now influence distance vector computations up to **2** hops away, i.e., at b and now at a, e as well
- t=3 c's state at t=0 may influence distance vector computations up to **3** hops away, i.e., at d, f, h
- t=4 c's state at t=0 may influence distance vector computations up to **4** hops away, i.e., at g, i



Comparisions	Link State	Distance Vector
1. Message Complexity	• n routers → cuz broadcast • $O(n^2)$ messages sent	• exchange b/w neighbours • convergence time varies
2. Speed of Convergence	• $O(n^2)$ algo • $O(n^2)$ messages • may have oscillations	• convergence time varies ↳ may have routing loops ↳ count to infinity problem } 4 CONS
3. Robustness what happens if router malfunctions	• router can advertise incorrect link cost • each router computes only its own table ↳ so not a huge issue	• router can advertise incorrect path cost (→ back-hoing) • each routers DV is used by others ↳ error propagates through network

1. Global

↳ all routers have complete
- ↳ topology
- ↳ link cost info

↳ link state algo
- e.g Dijkstra

2. Decentralized

↳ iterative process of computation

↳ exchanges info with neighbours

↳ routers only know link costs to attached neighbours

↳ distance vector algos
- e.g Bellman Ford

Internet Approach to Scalable Routing
intra-AS and inter-AS routing

Assumptions

↳ all routers identical

↳ network flat

... not true in practice

Scale

↳ billions of destinations

↳ cant store all destinations in routing tables

↳ routing table exchange would swamp links

Administrative Autonomy

↳ internet
- ↳ a network of networks

↳ each network admin may want to control routing in its own network

Autonomus systems (AS)

aka domain

↳ routers aggregated into regions

Intra AS

aka intra domain

↳ routing among routers within same AS network

↳ all routers in AS must run same intra-domain protocol

↳ gateway router

↳ at edge of it's own AS

↳ has links to routers in other AS'es

Inter AS

aka → inter domain

↳ routing among AS'es

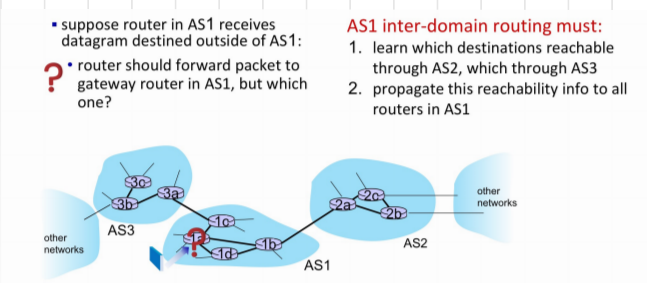
↳ gateways perform inter domain routing → as well as intra domain routing

routing within an AS

↳ Intra AS routing Protocols

1. Routing Information Protocol (RIP)
   - ↳ classic DV
     - ↳ DVs exchanged every 30 sec
   - ↳ no longer widely used
2. Enhanced Interior Gateway Routing Protocol (EIGRP)
   - ↳ DV based
   - ↳ formerly Cisco Proprietary
3. Open Shortest Path First (OSPF)
   - ↳ link-state routing
   - ↳ IS-IS protocol

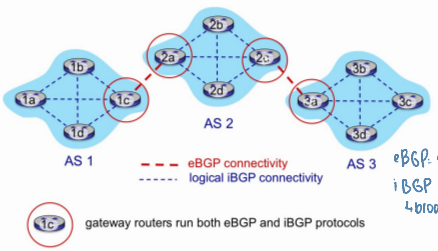
Inter-AS routing → a role in intradomain forwarding



Internet inter-AS routing: BGP

- BGP (Border Gateway Protocol): *the* de facto inter-domain routing protocol
  - "glue that holds the Internet together"
- allows subnet to advertise its existence, and the destinations it can reach, to rest of Internet: *"I am here, here is who I can reach, and how"*
- BGP provides each AS a means to:
  - obtain destination network reachability info from neighboring ASes (eBGP)
  - determine routes to other networks based on reachability information and *policy*
  - propagate reachability information to all AS-internal routers (iBGP)
  - advertise (to neighboring networks) destination reachability info

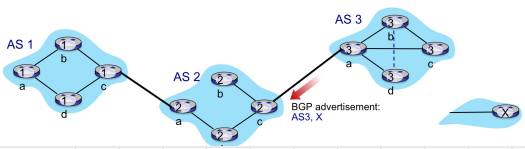
eBGP, iBGP connections



eBGP: gateways
iBGP
broadcasts data to routes

BGP essentials

- BGP session: two BGP routers ("peers") exchange BGP messages over semi-permanent TCP connection:
  - advertising *paths* to different destination network prefixes (BGP is a "path vector" protocol)
- when AS3 gateway 3a advertises path AS3,X to AS2 gateway 2c:
  - AS3 *promises* to AS2 it will forward datagrams towards X



Question 7: Consider the network shown in Figure 3. Suppose AS3 and AS2 are running OSPF for their intra-AS routing protocol. Suppose AS1 and AS4 are running RIP for their intra-AS routing protocol where weight of l1 < l2 both connected to router 1d. Suppose eBGP and iBGP are used for inter-AS routing protocol. Initially suppose there is no physical link between AS2 and AS4. **[10 points]**

There is no physical link between AS2 and AS4 for part (a) and (b)

a) Router 3c learns about prefix x from which routing protocol: OSPF, RIP, eBGP, or iBGP?

3c learn from 4c (AS4)using eBGP.

b) Router 3a learns about x from which routing protocol?

iBGP running on 3c.

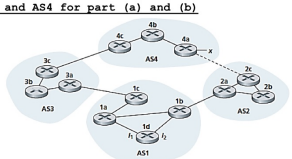


Figure 3

Now suppose that the link between AS2 and AS4 has been restored.

Physical link exists between AS2 and AS4 for part (c) and (d)

c) Router 1c learns about x from which routing protocol?

Learns x from 3a using eBGP. OR Learns x from 1b using iBGP.

d) Router 1d learns about x from which routing protocol?

Learns x from 1b using iBGP. OR Learns x from 1c using iBGP.

CHP 6

LINK LAYER & LANS

TERMINOLOGY

nodes
- ↳ hosts
- ↳ routers

comm. channels (links)
- ↳ wired
- ↳ wireless
- ↳ LAN

layer 2 packet
- ↳ frame
- ↳ encapsulates datagram

Link Layer

↳ transfers datagram from 1 node to physically adjacent node over a link

↳ datagram transfered by diff link protocols over diff links

e.g. Wifi on 1st link, Ethernet on 2nd link

↳ each link protocol provides diff services

e.g. may/may not provide reliable data transfer over link

transportation analogy:

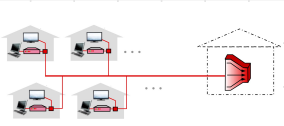
- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link-layer protocol
- travel agent = routing algorithm

LINK LAYER SERVICES

Framing link access
- ↳ encapsulate datagram into frame
  adding header, trailer
- ↳ channel access if shared medium
- ↳ MAC addresses in frame headers → diff from IP address!
  identify source, destination

Reliable delivery b/w adjacent nodes
- ↳ rarely used on low error bits
- ↳ wireless links: high error rates

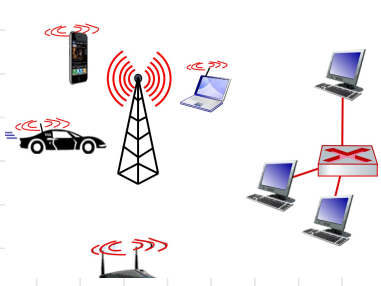
flow control
- ↳ pacing b/w adjacent
  sending and recving nodes



half duplex
- ↳ nodes at both ends of link can transmit
  - ↳ but not simultaneously

full duplex
- ↳ nodes at both ends of link can transmit
- ↳ simultaneously

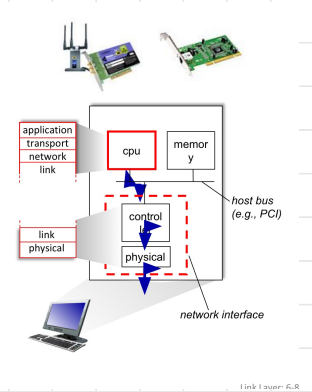
Error correction
- ↳ receiver identifies and corrects
  bit errors, w/o retransmission



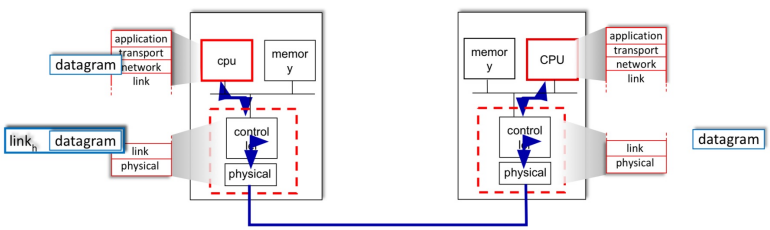
Error detection
- ↳ errors caused by
  - ↳ signal attenuation
  - ↳ noise
- ↳ receiver detects errors
- ↳ signals retransmission / drops frame

Where is Link Layer implemented

- in each and every host
- in NIC (network interface card) or on a chip
  - ethernet card
  - wifi card/chip
- attaches into host's system buses
- combination of hardware, software, firmware



Interfaces communicating



Sending Side

- encapsulates datagram in frame
  - adds error checking bits
  - reliable data transfer
  - flow control
  - etc

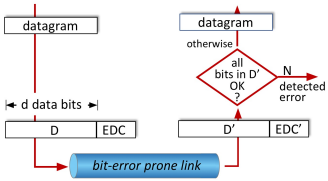
Receiving Side

- looks for errors
  - reliable data transfer
  - flow control
  - etc
- extracts datagram
  passes to upper layer
  at receiving side

ERROR DETECTION

EDC: Error Detection and correction bits

D: data Protected by error checking
may include header files

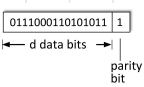


Error detection not 100% reliable!
- protocol may miss some errors, but rarely
- larger EDC field yields better detection and correction

PARITY CHECKING

Single bit Parity

↳ detect single bit errors

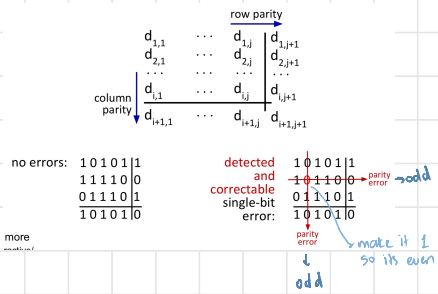


Even parity: set parity bit so there is an even number of 1's

make 1's even

2-dimensional bit Parity

↳ detect and Correct single bit errors



Internet checksum

↳ detect errors in transmitted segment
(flipped bits)

Sender

↳ treat contents of UDP segment as sequence of 16 bits

↳ checksum
↳ addition of segment content (1's complement sum)

↳ checksum value put into UDP checksum feild

Receiver

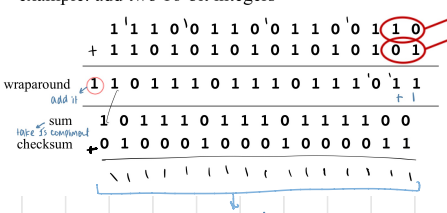
↳ compute checksum of received segment

↳ check if computed checksum equals checksum feild value

↳ not equal → error detected

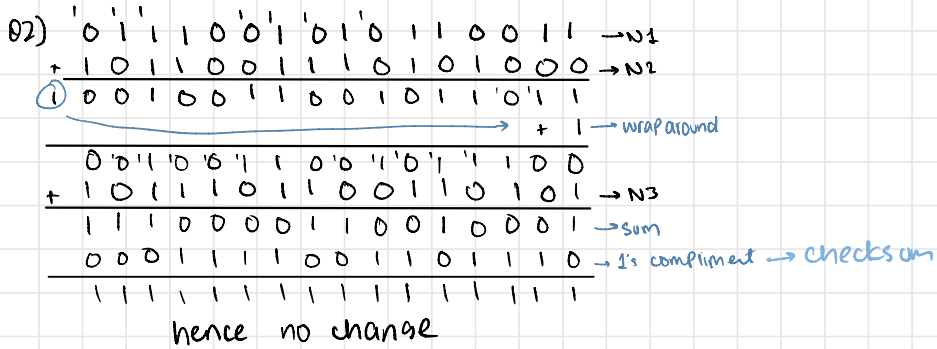
↳ equal → no error
BUT
maybe errors nonetheless...?

example: add two 16-bit integers



Even though numbers have changed (bit flips), *no* change in checksum!

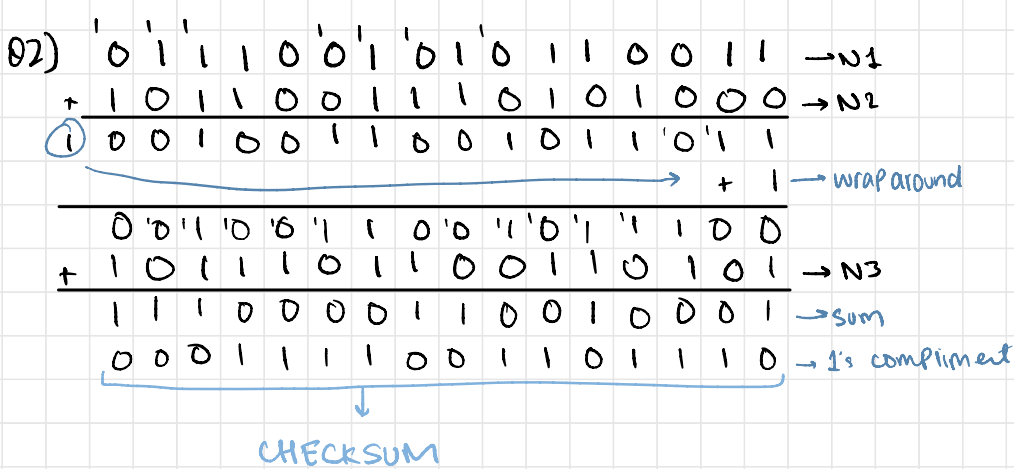
if all 1's : no change
if any "0" : change → error



Question 2: **[10 Marks]**

Checksum is an error-detecting code used in many Internet standard protocols, including IP, TCP, and UDP. You have to generate Checksum for the following data blocks, which are transmitted using UDP.

N1	N2	N3
0111001010110011	1011001110101000	1011101100110101



Cyclic Redundancy Check (CRC)

↳ more powerfull error detection coding

D: data bits → given → think of these as binary number

G: bit pattern generator → given

of r+1 bits

A	B	⊕ XOR
0	1	1
0	0	0
1	1	0
1	0	1

same = 0

diff = 1

~~skip~~

Sender side

↳ add bits: len(divisor)-1

↳ XOR dividing

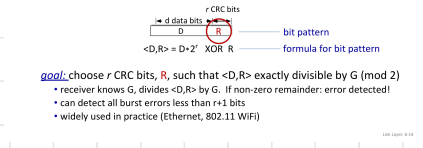
↳ sends data + CRC to receiver

Receiver side

↳ XOR dividing

↳ if all 0's → no error

↳ if any 1 → error



goal: choose r CRC bits, R, such that <D,R> exactly divisible by G (mod 2)

- receiver knows G, divides <D,R> by G. If non-zero remainder: error detected!
- can detect all burst errors less than r+1 bits
- widely used in practice (Ethernet, 802.11 WiFi)

Link Layer: 6-14

Cyclic Redundancy Check (CRC): example

We want:

$D \cdot 2^r$ XOR $R = nG$

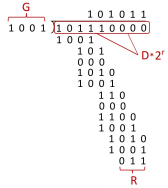
or equivalently:

$D \cdot 2^r = nG$ XOR R

or equivalently:

if we divide $D \cdot 2^r$ by G, want remainder R to satisfy:

$R = \text{remainder}\left[\frac{D \cdot 2^r}{G}\right]$

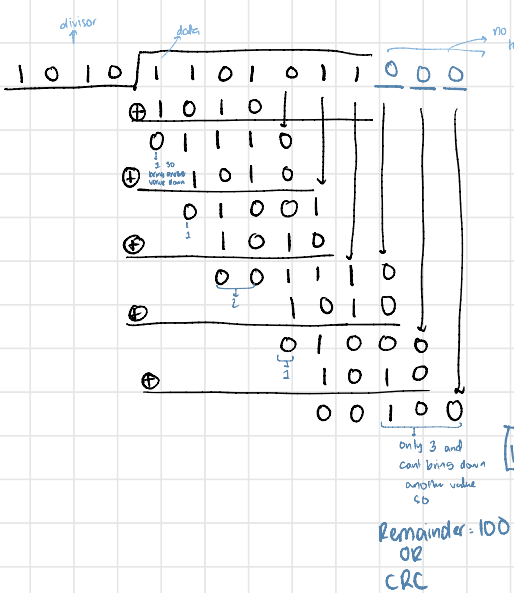


* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Link Layer:

8) data = 1101011 divisor = 1010

SENDER SIDE



Remainder: 100

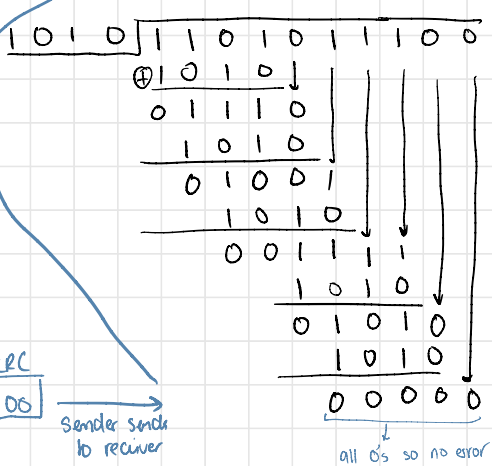
OR

CRC

data	CRC
1101 011	100

Sender sends to receiver

RECEIVER SIDE



all 0's so no error

Figure *Division in CRC encoder*

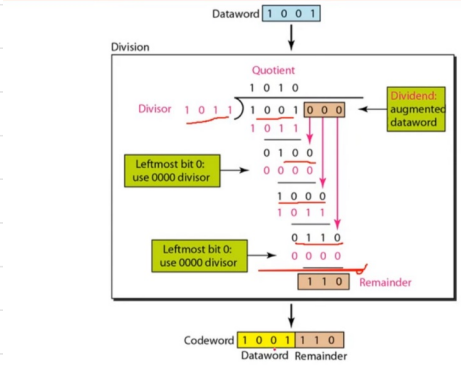
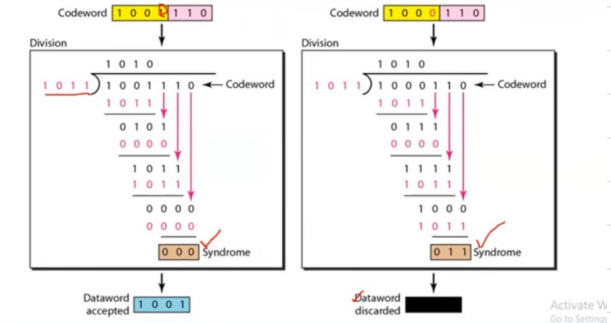


Figure *Division in the CRC decoder for two cases*



multiple access links

two types of links

1. point to point link
   - ↳ is b/w Ethernet switch and host
   - ↳ PPP for dial up access

2. broadcast → shared wire / medium
   - ↳ old fashioned Ethernet
   - ↳ upstream HFC in cable based access network

multiple access protocols

- ↳ single shared broadcast channel
- ↳ two or more simultaneous transmissions by nodes → interference
  - ↳ collision if node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
  - no out-of-band channel for coordination

ideal multiple acess protocol

Given: multiple access channel (MAC) of rate R bps

Desiderata:

1. when 1 node wants to transmit, it can sent at rate R
2. when M nodes want to transmit, each can send at avg rate R/M
3. Fully decentralized
   - ↳ no special node to coordinate transmissions
   - ↳ no synchronization of clocks, slots
4. Simple

MAC PROTOCOLS : TAXONOMY

Three broad classes

1. Channel Partitioning

↳ divide channel into smaller pieces → time slots, frequency, code

↳ allocate piece to node for exclusive use

2. Random Access

↳ channel not divided → allows collisions

↳ recover from collisions

3. Taking Turns

↳ nodes take turns

↳ but nodes with more to send can take longer turns

1. CHANNEL PARTITIONING MAC PROTOCOLS

TDMA: Time Division Multiple Access

↳ access to channel in rounds

↳ each station gets fixed length slot → length: packet transmission time in each round

↳ unused slots go idle

- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle

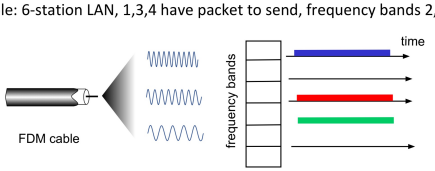
FDMA: Frequency Division Multiple Access

↳ channel spectrum divided into frequency bands

↳ each station assigned fixed frequency bands

↳ unused transmission time in freq bands go idle

- example: 6-station LAN, 1,3,4 have packet to send, frequency bands 2,5,6 idle



2. Random Access Protocols

↳ how to detect collisions

↳ how to recover from collisions → via delayed retransmissions

style

↳ when node has packet to send

↳ transmit at full channel data rate R

↳ no prior coordination among nodes

↳ two or more transmitting nodes → COLLISION

example PROTOCOLS

↳ slotted ALOHA, unslotted ALOHA (Also: Pure ALOHA)

↳ CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA

↳ ASSUMPTIONS

↳ all frames of same size

↳ time divided into equal size slots (time to transmit 1 frame)

↳ nodes start to transmit only slot beggining

↳ nodes are synchronized

↳ if 2 or more nodes transmit in slot

all nodes detect collision

↳ OPERATION

↳ when node obtains fresh frame

transmits in next slot

↳ if no collision

node can send new frame in next slot

↳ if collision

↳ node retransmits frame in each subsequent slot

with probability P (randomization), until success

PROS

↳ single active node can continously transmit at full rate of channel

↳ highly decentralized

↳ only slots in node need to be in sync

↳ simple

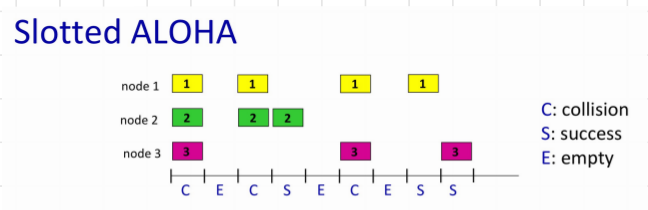
CONS

↳ collisions, wasting slots

↳ idle slots

↳ nodes may be able to detect collisions in less than time to transmit packet

↳ clock synchronization



EFFICIENCY 0.37%

↳ long-run fraction of successful slots → many nodes, all with many frames to send

- *suppose:* N nodes with many frames to send, each transmits in slot with probability p
  - prob that given node has success in a slot $= p(1-p)^{N-1}$
  - prob that *any* node has a success $= Np(1-p)^{N-1}$
  - max efficiency: find p^* that maximizes $Np(1-p)^{N-1}$
  - for many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives:

  max efficiency = 1/e = .37
- *at best:* channel used for useful transmissions 37% of time!

PURE ALOHA (unslotted) 18% efficiency

↳ unslotted Aloha

↳ simpler

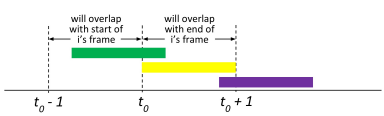
↳ no synchronization

↳ when frame first arrives, transmit immediately

↳ collision probability increases, with no synchronization

andoun ki tarna
aagaay frame bhej dayta hai

- frame sent at t_0 collides with other frames sent in $[t_0-1, t_0+1]$



- pure Aloha efficiency: 18% !

CSMA: carrier sense multiple Access

Simple CSMA

↳ listen before transmit
  ↳ if channel sensed idle
    transmit entire frame
  ↳ if channel sensed busy
    defer transmission

human analogy: DONT INTERUPT OTHERS!

CSMA/CD

better efficiency than ALOHA

↳ CSMA with collision detection
  ↳ collision detected within shorter time
  ↳ colliding retransmissions aborted
    ↳ reducing channel wastage
  ↳ collision detection easy in wired
    difficult with wireless

HUMAN ANOLOGY: THE POLITE CONVESATIONALIST

CSMA: collisions

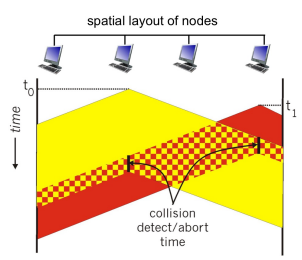
↳ collisions can still occur with carrier sensing
  ↳ propagation delay means two nodes
    may not hear each others just-started transmission

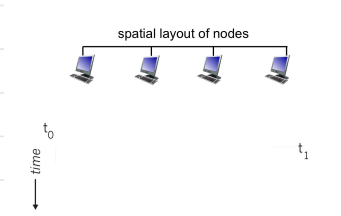
leads to

↳ collision
  con
  ↳ entire packet transmission time wasted
    ↳ distance & propagation delay play role
      in determining collision probability

pro

↳ reduces the amount of time wasted in collisions
  ↳ transmission aborted on collision detection





CSMA/CD efficiency

- T_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

$$efficiency = \frac{1}{1 + 5t_{prop}/t_{trans}}$$

- efficiency goes to 1
  - as t_{prop} goes to 0
  - as t_{trans} goes to infinity
- better performance than ALOHA: and simple, cheap, decentralized!

3. Taking Turns MAC Protocols

Polling

↳ controller node invites one nodes (clients) to transmit in turn

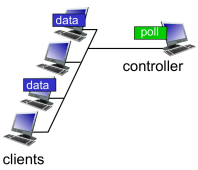
↳ typically used with dumb devices

cons

↳ polling overhead

↳ latency

↳ single point failure (controller)



Token Passing

↳ control token passed from one node to next seaurentially

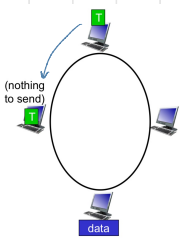
↳ token message

cons

↳ token overhead

↳ latency

↳ single point failure (token)



C O M P A R I S I O N S

CHANNEL PARTITION

↳ share channel efficiency and fairly at high load

↳ inefficient at low load

↳ delay in channel access

↳ 1/N bandwidth allocated even if only 1 active node!

RANDOM ACCESS

↳ efficient at low load

↳ single load can fully utilize channel

↳ high load

↳ collision overhead

TAKING TURNS

↳ look for the best of both worlds

ishma hafeez notes
repshfee

Q web server 10.11.21.13
mail server 10.11.21.14

↓ Authoritative name server
GoDaddy alias dns1.GoDaddy.com
192.168.10.1

ANS

Question 2: Systemslabs.com is registered and hosted with GoDaddy. Both the web server and mail server of Systemslabs are associated with 10.11.21.13 and 10.11.21.14 respectively. The primary authoritative name server of GoDaddy is dns1.GoDaddy.com which is mapped to 192.168.10.1. List down all the resource records (RRs) that will be inserted into the authoritative name server and .com's TLD (top level domain) server. **[10 points]**

Solution:
RR Inserted in Authoritative NS
(Systemslabs.com, 10.11.21.13, A)
(Systemslabs.com, 10.11.21.14, MX)
RR Inserted in TLD
(Systemslab.com, dns1.GoDaddy.com, NS)
(dns1.GoDaddy.com, 192.168.10.1, A)

1000

8
0000

Question 6: A provider has been assigned the network **209.118.127.0/23** and wants to divide it among three customers. **Ufone** needs to accommodate up to 250 hosts, **Zong** needs to accommodate up to 48 hosts and **Telenor** needs to accommodate up to 120 hosts. Fill the following table in your answer script with the details of the sub-networks that the provider can create to fit its customers' needs. **[10 points]**

NID HID

Subnet No.	Network Addr	Netmask	Host Range	No. Of Hosts
Ufone	**209.118.126.0/ 24**	255.255.255.0	**209.118.126.0** --- **209.118.126.255**	254 250
Zong	**209.118.127.0/25**	255.255.255.128	**209.118.127.0** --- **209.118.127.127**	126 48
Telenor	**209.118.127.128/25**	255.255.255.128	**209.118.127.128** --- **209.118.127.255**	126 120

Class C 32

Network Bits	Subnet Mask	Number of Subnets	Number of Hosts
/24	255.255.255.0	0	254
/25	255.255.255.128	2 (0)	126
/26	255.255.255.192	4 (2)	62
/27	255.255.255.224	8 (6)	30
/28	255.255.255.240	16 (14)	14
/29	255.255.255.248	32 (30)	6
/30	255.255.255.252	64 (62)	2

Q) 192.168.100.0/24 → class C
4 sub networks?

Possible Hosts

Sunny Subnetting Table

Subnet	1	2	4	8	16	32	64	128	256
Host	256	128	64	32	16	8	4	2	1
Subnet Mask	/24	/25	/26	/27	/28	/29	/30	/31	/32

Subnet = 4
Possible hosts = 64
Usable hosts = 62

network ID

NID +64	Subnet mask	HID Range (+1 NID, -1 BID)	BID (broadcast ID, -1 NID)
192.168.100.0	255.255.255.192	100.1 – 100.62	192.168.100.63
192.168.100.64	255.255.255.192	100.65 – 100.126	192.168.100.127
192.168.100.128		100.129 – 100.190	192.168.100.191
192.168.100.192		100.193 – 100.254	192.168.100.255

↓ subnet same for all

??

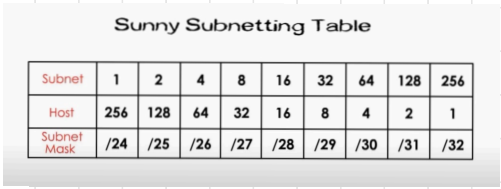
The network has the following addressing requirements:

i) The Department-1 LAN-1 will require 50 host IP addresses.
ii) The Department-1 LAN-2 will require 120 host IP addresses.
iii) The Department-2 LAN 1 will require 10 host IP addresses.
iv) The Department-2 LAN 2 will require 18 host IP addresses.

Fill the following table in your answer script with the details of the sub-networks that the Departments can create to fit its LAN's needs.

Subnet No.	Network Address	Custom Subnet Mask	Host Range	No. Of Hosts
Department-1 LAN-1				
Department-1 LAN-2				
Department-2 LAN-1				
Department-2 LAN-2				

dress 192.168.9.0/24. Now you have
k Figure 05.



192.168.9.160 ---192.168.9.175
192.168.9.176 ---192.168.9.191
192.168.9.192 ---192.168.9.207
192.168.9.208 ---192.168.9.223
192.168.9.224 ---192.168.9.239
192.168.9.240 ---192.168.9.255

The Department-2 LAN 2will require 18 host IP addresses.
Determine the following.

a) Number of Bit Borrowed= 3
b) Total Number of Subnets= $2^3 = 8$
c) Total Number of Host Addresses= 32
d) Number of Usable Addresses= 30
e) Custom Subnet Mask= 255.255.255.224
Subnet Range (Network & Broadcast address)

192.168.9.0 ---192.168.9.31
192.168.9.32 ---192.168.9.63
192.168.9.64---192.168.9.95
192.168.9.96 ---192.168.9.127
192.168.9.128 ---192.168.9.159
192.168.9.160 ---192.168.9.191
192.168.9.192 ---192.168.9.223
192.168.9.224---192.168.9.255

The MD LAN-1 will require 28 host IP addresses.
We can assign any of the remaining subnet from part (d).