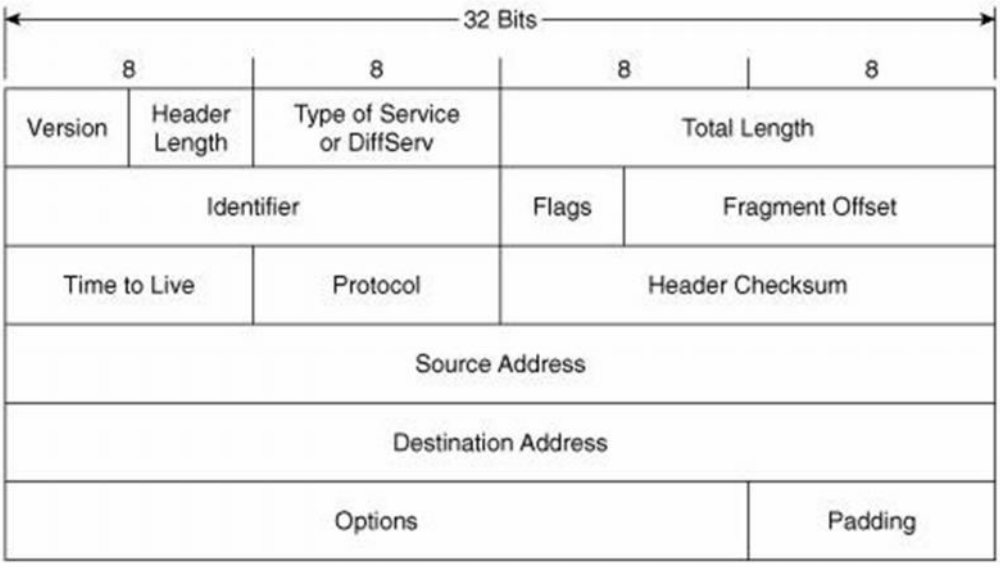
SSN COLLEGE OF ENGINEERING, KALAVAKKAM
Department of Computer Science and Engineering
B.E. (CSE) IV semester UNIT TEST – II
CS6551 - Computer Networks
Answer Key

Part A **[5 * 2 = 10]**

1. The six subnets are:
   201.80.64.0
   201.80.64.32
   201.80.64.64
   201.80.64.64
   201.80.64.96
   201.80.64.128
   201.80.64.160
   The remaining 2 are unused
   201.80.64.192
   201.80.64.224
2. First Fragment as both offset and M are 0
3. Router : belongs to Network Layer
   Performs routing based on IP address
   Bridge : performs the functions defined in datalink layer and physical layer
   Does data transfer based on MAC Address.
4. Discover Phase
   Offer Phase
   Request Phase
   Acknowledgement Phase
   Release Phase

Part B **[8 + 16 + 16]**

5.



The IPv4 packet header consists of 14 fields, of which 13 are required. The 14th field is optional named: options. The IPv4 packet header consists of 20 bytes of data.

- **Version**:- The first header field in an IP packet is the four-bit version field. The Version field indicates the format of the internet header. Version identifies the IP version to which the packet belongs. This four-bit field is set to binary 0100 to indicate version 4 (IPv4) or binary 0110 to indicate version 6 (IPv6).

- **Header length or Internet Header Length (IHL) :-** The second field (4 bits) is the Internet Header Length (IHL) telling the number of 32-bit words in the header. Since an IPv4 header may contain a variable number of options, this field specifies the size of the header
- **Type of Service(ToS):–** now known as **Differentiated Services Code Point (DSCP).** The TOS field is used to carry information to provide quality of service features.

- **Explicit Congestion Notification (ECN) :-It** allows end-to-end notification of network congestion without dropping packets. ECN is an optional feature that is only used when both endpoints support it and are willing to use it. It is only effective when supported by the underlying network.
- **Total Length:-** This 16-bit field defines the entire datagram size, including header and data, in bytes. The minimum-length datagram is 20 bytes (20-byte header + 0 bytes data) and the maximum is 65,535 bytes — the maximum value of a 16-bit word. The minimum size datagram that any host is required to be able to handle is 576 bytes,
- **Identification:–** This field is an identification field and is primarily used for uniquely identifying fragments of an original IP datagram. Some experimental work has suggested using the ID field for other purposes, such as for adding packet-tracing information to datagrams in order to help trace back datagrams with spoofed source addresses.
- **Flags:–**A three-bit field follows and is used to control or identify fragments. They are (in order, from high order to low order):
    - bit 0: Reserved; must be zero.
    - bit 1: Don’t Fragment (DF)
    - bit 2: More Fragments (MF)
- **Don’t Fragment:-** Sets the Don’t Fragment bit in sent packets. When an IP datagram has its DF flag set, intermediate devices are not allowed to fragment it so if it needs to travel across a network with a MTU(Maximum Transmission Unit) smaller that datagram length the datagram will have to be dropped.
- **More Fragments:-** Sets the More Fragments bit in sent packets. The MF flag is set to indicate the receiver that the current datagram is a fragment of some larger datagram. When set to zero it indicates that the current datagram is either the last fragment in the set or that it is the only fragment.
- **Fragment Offset:-**The fragment offset field, measured in units of eight-byte blocks, is 13 bits long and specifies the offset of a particular fragment relative to the beginning of the original unfragmented IP datagram. The first fragment has an offset of zero. This allows a maximum offset of $(2^{13} – 1) \times 8 = 65{,}528$ bytes which would exceed the maximum IP packet length of 65,535 bytes with the header length included (65,528 + 20 = 65,548 bytes).

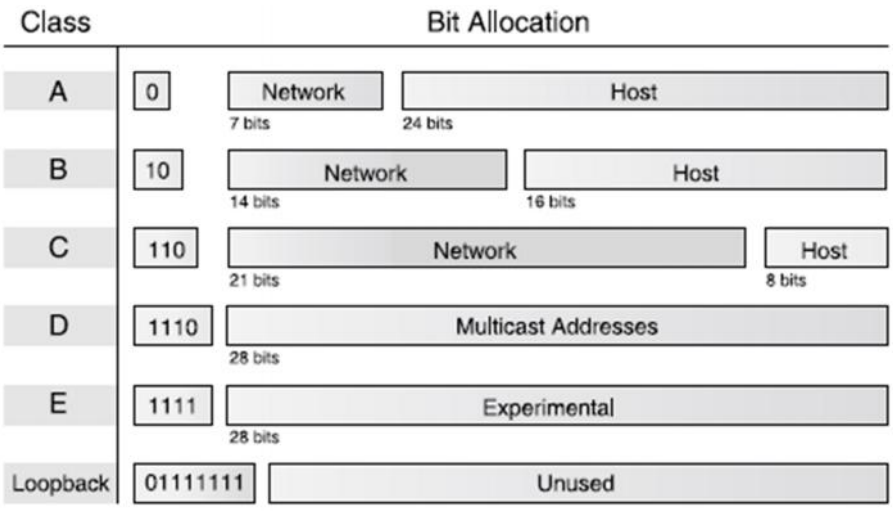
- **Time To Live (TTL):-**It is of 8 bit field. This field indicates the maximum time the datagram is allowed to remain in the internet system. If this field contains the value zero, then the datagram must be destroyed. This field is modified in internet header processing.

The time is measured in units of seconds, but since every module that processes a datagram must decrease the TTL by at least one even if it process the datagram in less than a second, the TTL must be thought of only as an upper bound on the time a datagram may exist. The intention is to cause undeliverable datagrams to be discarded, and to bound the maximum datagram lifetime. <*hops*> must be a number in the range [0–255].

- **Protocol:**-This field defines the protocol used in the data portion of the IP datagram.
- **Header Checksum:-** The 16-bit checksum field is used for error-checking of the header. At each hop, the checksum of the header must be compared to the value of this field. If a header checksum is found to be mismatched, then the packet is discarded. Errors in the data field must be handled by the encapsulated protocol and both UDP and TCP have checksum fields.

- **Source address :-** Sets the source IP address. This option lets you specify a custom IP address to be used as source IP address in sent packets. This allows spoofing the sender of the packets. <*addr*> can be an IPv4 address or a hostname.
- **Destination address :-** An IPv4 address indicating the receiver of the packet. As with the Source address, this may be changed in transit by a network address translation device.
- **Options:-**Additional header fields may follow the destination address field, but these are not often used. The value in the IHL field must include enough extra 32-bit words to hold all the options (plus any padding needed to ensure that the header contains an integral number of 32-bit words). The list of options may be terminated with an EOL (End of Options List) option; this is only necessary if the end of the options would not otherwise coincide with the end of the header.

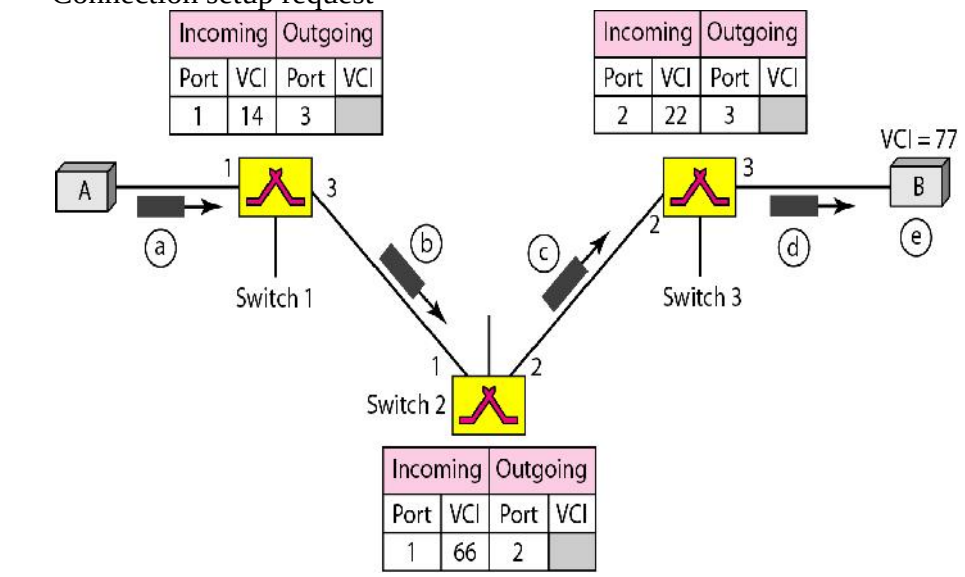
b.



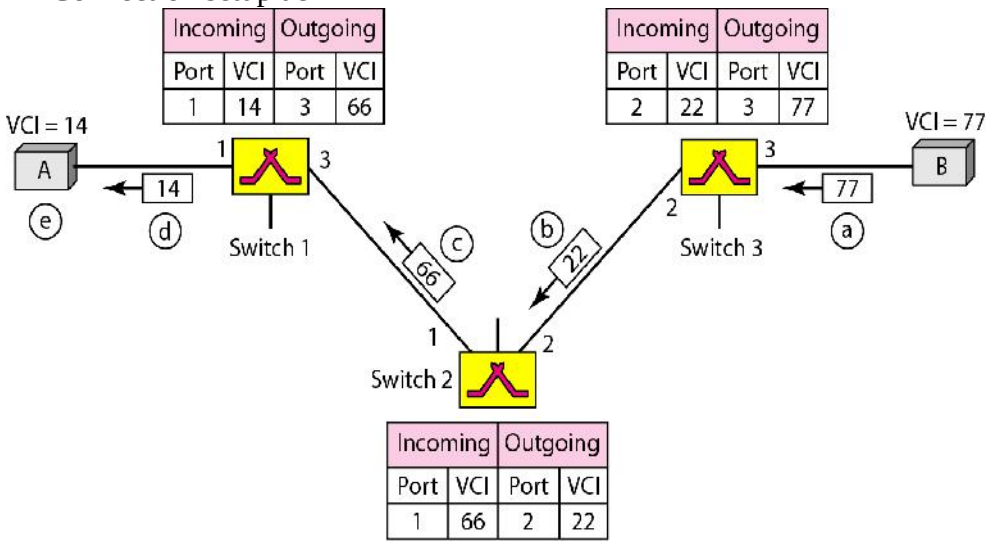
6. Virtual Circuit Networks
   A virtual circuit network is a cross between a circuit switched network and a datagram network.
   Characteristics of VCN
   a. 3 phases as in CSN (connection setup, data transfer, connection teardown)
   b. Resources allocated during connection setup (CSN) or on demand (DN)
   c. Data in form of packets
   d. All packets follow the same path (CSN)

Connection setup phase

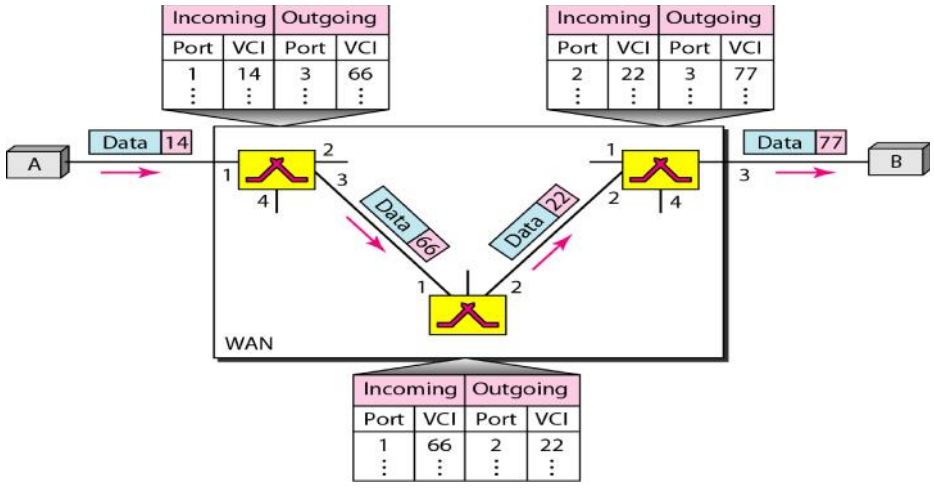
- Connection setup request



- Connection setup ack



Data Transfer in a VCN



7. Distance Vector Routing

Commonly called as Distributed Bellman Ford

Routing Algorithm and the Ford Fulkerson Algorithm

Completely decentralized algorithm

No node has complete information about the costs of

No node has complete information about the costs of all network links

Gradual calculation of path by exchanging information with neighbors

a. Each node constructs a one dimensional array (vector) containing the distances (costs) to all other nodes (as it relates to its knowledge) and distributes it to its immediate neighbors.
b. Each node knows the cost of links to its immediate neighbor.
c. If no link exists between two nodes, the cost between the nodes is marked as infinity.

Representation

The cost from X to Y via Z is the cost from X to Z plus the "minimum" cost from Z to Y.

$$Dx(Y,Z) = C(X, Z) + \min_W\{ Dz(Y,W)\}$$

Dx(Y,Z) Cost from X to Y via Z

C(X , Z) Cost from X to Z

minW{ Dz(Y,W)}

Minimum cost from Z to Y via W.

Minimum cost from Z to Y computed by taking all possible paths into consideration.

Example with Explanation

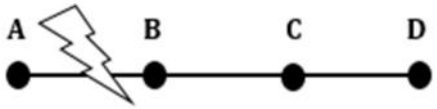
Count to Infinity

A goes down.

_ None of the nodes actually knows that A is unreachable

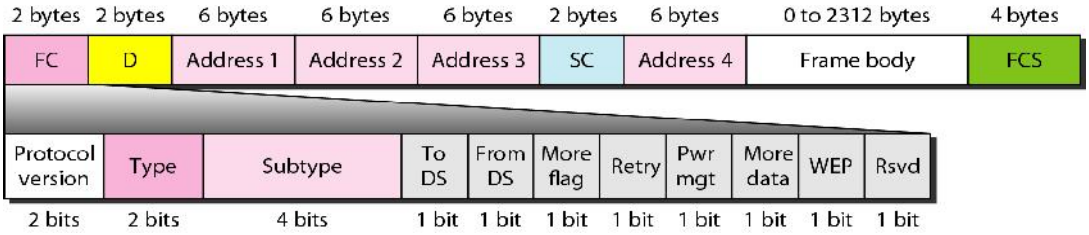
_ This situation is known as the count-to-infinity problem.

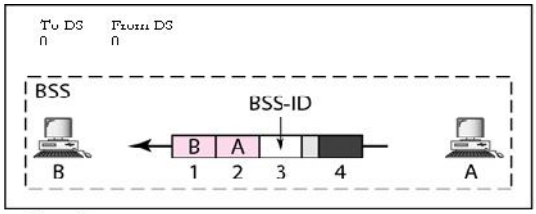
Link Between A & B is Broken



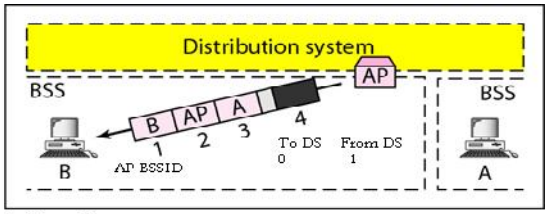
	A	B	C	D
A	0, -	1, A	2, B	3, C
B	1, B	0, -	2, C	3, D
C	2, B	1, C	0, -	1, C
D	3, B	2, C	1, D	0, -

8. Explanation of frame format with the techniques for collision avoidance (How hidden terminal problem and exposed terminal problem is overcome using RTS and CTS)

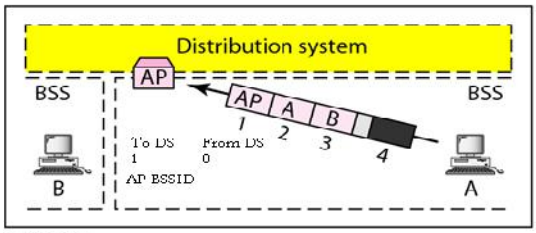




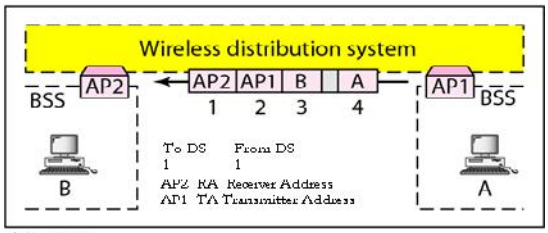
a. Case 1



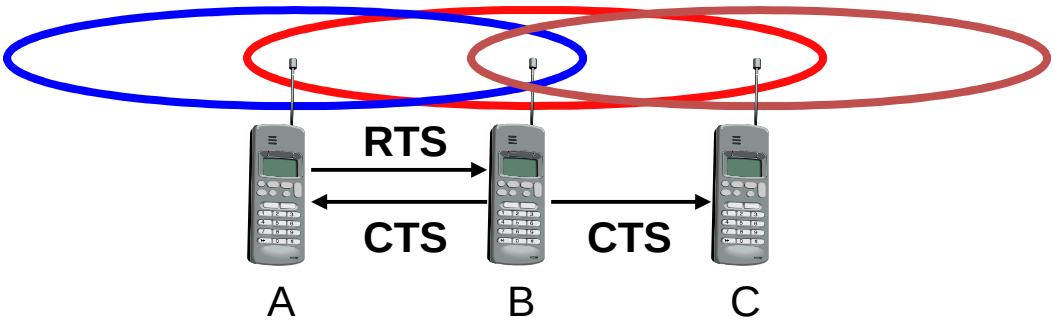
b. Case 2

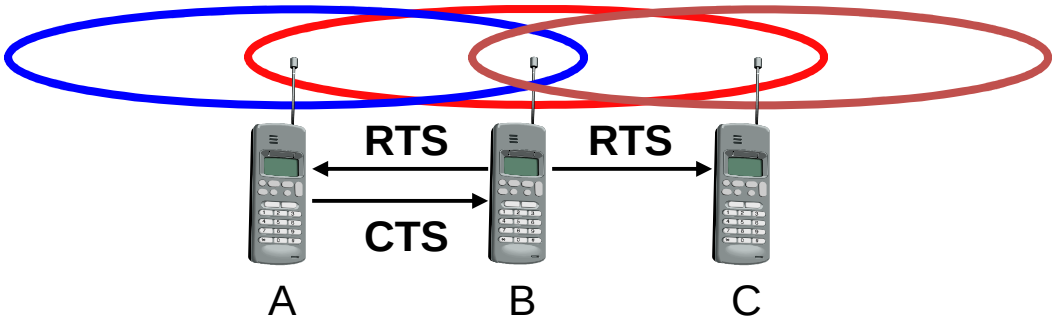


c. Case 3



d. Case 4





(or)

9. A) Explanation of Learning Bridge , Loop problem , Explanation of spanning tree algo. With example

B) TCP Packet = 900 (data) + 20 (IP header) = 920 Bytes.
IP Packet = 920 (data) + 20 (IP header) = 940 Bytes

Link A-R1: Can support 1024 bytes including 14-byte frame header (e.g., 1010-byte IP data packet) thus no fragmentation is needed.

Length = 940, ID = x, DF = 0, MF = 0, Offset = 0.

Link R1-R2: Can support 512 bytes including 8-byte header (e.g., 512-8=504 bytes of IP data packet and 504 – 20 (IP header) = 484). Since IP data in a frame must be multiple of 8 bytes:

$8*x \leq 484 \Rightarrow x \leq 60.5$ bytes. Use x = 60 bytes
IP data 1 = 60*8 = 480 Bytes
IP packet size 1 = 480 + 20 = 500 Bytes.
IP data 2 = 920 – 480 = 440 Bytes
IP packet size 2 = 440 + 20 = 460 Bytes.

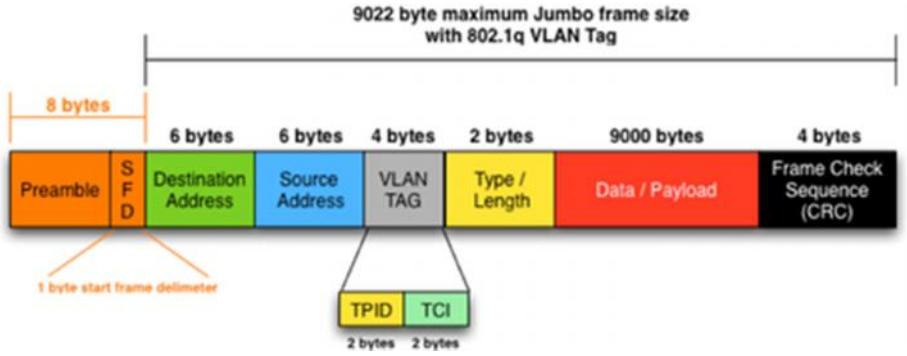
[1] Length = 500, ID = x, DF = 0, MF = 1, Offset = 0.
[2] Length = 460, ID = x, DF = 0, MF = 0, Offset = 60.

Link R2-B: Can support 512 bytes including 12-byte header (e.g., 512-8=500 bytes of IP data packet and 500 – 20 (IP header) = 480). This implies that no additional fragmentation is needed. Thus

[1] Length = 500, ID = x, DF = 0, MF = 1, Offset = 0.
[2] Length = 460, ID = x, DF = 0, MF = 0, Offset = 60.

(Or)

10. a.



Preamble : It is field which is used for bit synchronization by Ethernet.

Start frame delimiter: it is used by Ethernet for byte alignment purposes.

Source Address : it is a layer 2 address (48 bit ot 6 octets MAC Address) of source device/host.

Destination Address : it is a layer 2 address (48 bit ot 6 octets MAC Address) of destination device/host**.**

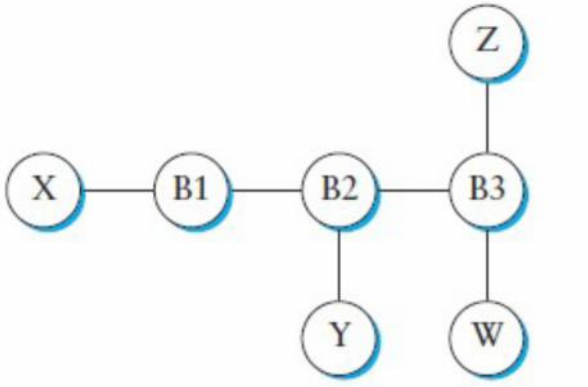
Vlan Tag : it is used to identify the vlan number of the frame.

Length/Type: it is the number of bytes of data in that frame.

Data : Actual data or payload in the frame.

FCS : Frame check sequence is used to detect corrupt data by doing cyclic redundancy check.

b)



(a) When X sends to Z the packet is forwarded on all links; all bridges learn where X is. Y's network interface would see this packet.

(b) When Z sends to X, all bridges already know where X is, so each bridge forwards the packet only on the link towards X, that is, B3 → B2 → B1 → X. Since the packet traverses all bridges, all bridges learn where Z is. Y's network interface would not see the packet as B2 would only forward it on the B1 link.

(c) When Y sends to X, B2 would forward the packet to B1, which in turn forwards it to X. Bridges B2 and B1 thus learn where Y is. B3 and Z never see the packet.

(d) When Z sends to Y, B3 does not know where Y is, and so retransmits on all links; W's network interface would thus see the packet. When the packet arrives at B2, though, it is retransmitted only to Y (and not to B1) as B2 does know where Y is from step (c). All bridges already knew where Z was, from step (b).