EE 122 Spring 1998

Midterm 2

1. Answer the following questions in a few *concise* sentences:

a. Project 1:
When writing networking code, when should you call ntohl() and why?
When copying a data item bigget than a byte (e.g. int) from the network buffer to a host data structure (at the reeiver). It is needed because different hosts may use different byte ordering.
Why didn't we need to call nothl() or htonl() on the data?
We received the data as a string of bytes. Any conversion would have to be done by the application.
Describe what would have happened if the receiver application (in this case readfile) never read any of the data (i.e., never called rfp_read())?
The receiver's buffer will fill so packets will be dropped. (Alternatively, the sender will keep retransmitting the same packets until it gives up).

b. Internet Routing
List TWO limitations on the size of an internetwork using RIP. For each, explain how the limitation is eliminated or reduced in OSPF. (Note: we will only grade your first two answers.)
1. Diameter of the network is limited by maximum cost of 15. OSPF overcomes this because it is a link state algorithm and thus advertises cost of links, net paths
2. Size of updates + state may grow large -> OSPF allows hierarchy within a routing domain
3. The time to converge can be large for RIP (because of counting to infinity) -> OSPF is link-state
List TWO reasons why BGP does not attempt to advertise the cost of paths. (Note: we will grade only your first two answers.)
1. Hard to connect from heterogeneous costs from various routing domains into a single cost for BGP
2. Many different routing policies can be used (e.g. shortest path, shortest delay, highest bandwidth, etc.). We cannot represent all policies in a single cost.

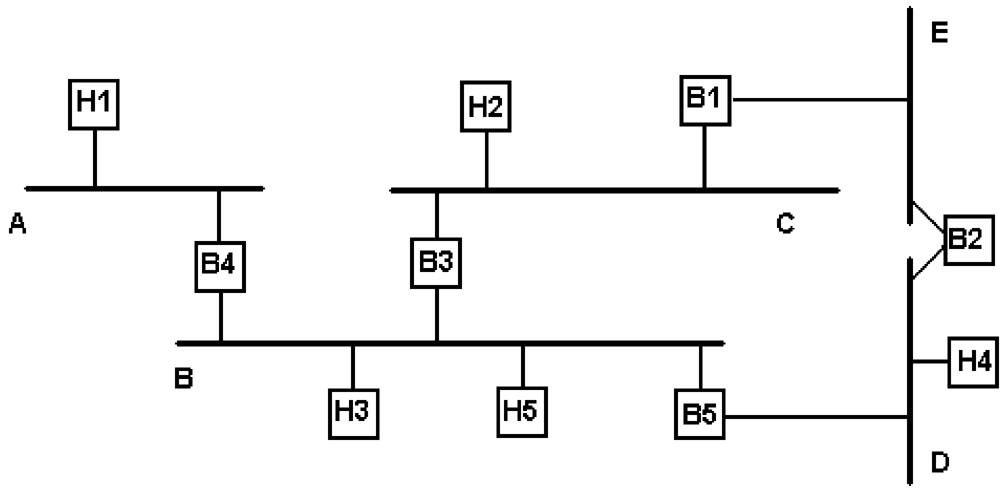
c. List THREE differences between IPv4 and IPv6. For each difference, explain the motivation for the change (i.e., why is it an improvement over IPv4?) We will grade your first three answers only.
1. Larger address space: Need more addresses, encode physical address in IP address
2. Non-global addresses: More efficient use of address space, autoconfiguration
3. Fragmentation moved to extension header: More efficient for most packets (not frag.)
4. Options moved to extension headers: More efficent, no limitation on size
5. TTL -> Hop Limit: more accurate name
6. Geographic addresses: Change provider w/o changing address (but keep route aggregation)
7. Multicast addresses: Core part of IPv6, not "optional" (no tunneling for IPv6 routes)

d. Can you tell from their names computer1.company.com and computer2.school.edu are in the same autonomous system (routing domain)? If so, how can you tell? If not, why not?
No, the naming domain is independent of the routing domain.

e. Give three scenarios in which a host sends an IGMP Host Membership Report for group G, but still doesn't receive any multicast packets. Assume the routing algorithm does not use prune messages (Truncated Reverse Path Broadcast).
1. No one is sending data!
2. Data packets are lost (congestion, TTL too small, . . .)
3. The Host Membership Report is lost

4. No path exists from source to this host (e.g. routing loop)
5. Router craches

2. the following diagram depicts an extended LAN of hosts and bridges. You can assume that bridge B1 has a lower ID than bridge B2, etc.



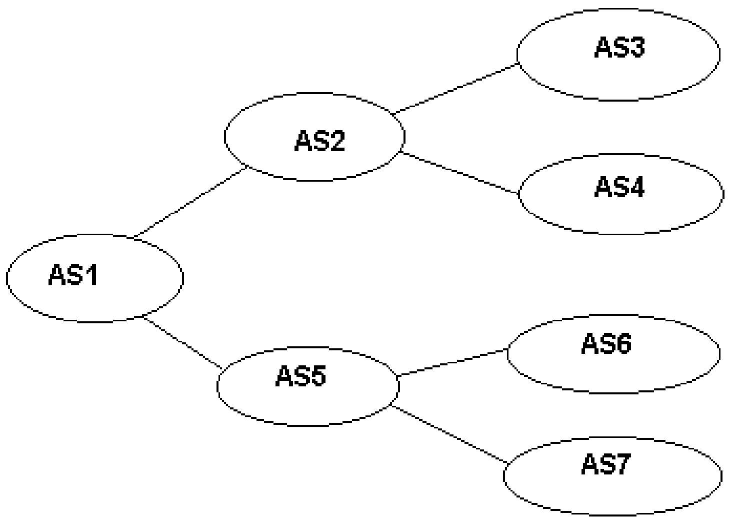
a. Assuming the bridges have run a spanning tree algorithm as described in class, will bridge B3 forward frames from host H4 from subnet C to subnet B? Why or why not?
+1 yes
+1 It is the designated bridge because it has the shortest path to the rot (B1).

b. A technician using a network monitor noticed that bridge B4 forwarded one fram from host H3 onto subnet A, but didn't forward the next. What is the most likely reason for this behavior? You may assume there are no equipment failures.
B4 is a learning bridge, and the fram is destined for a host that B4 knows is on subnet B (e.g. H5)

c. Now assume all the bridges are replaced with IP Multicast routers using the DVMRP algorithm. In that case, will router B3 forward packets from host H4 from subnet C onto subnet B?
No
(For DVMRP, trees are source-specific. For source H4, B5 is the dominant router on subnet B, since it has the shortest path from the source.)

d. Again, assuming DVMRP routers, list the condition under which roter B4 will not forward packets from host H3 unto subnet A. Assume no equipment failures.
No hosts on subnet A have joined the group to which the packet is addressed (e.g. group C)

3. In the internetwork shown below, each autonomous system (AS) contains N subnets and each subnet contains H hosts. Perform the computations below. You must indicate where each part of your answer comes from.



a. Assume the internetwork uses the original IP addressing and routing schemes (hierarchical addresses and routing; no subnetting or CIDR). Compute the minimum number of routingentries required by a router in AS3.
N+1
N: all networks in AS3
1: default router for all other networks

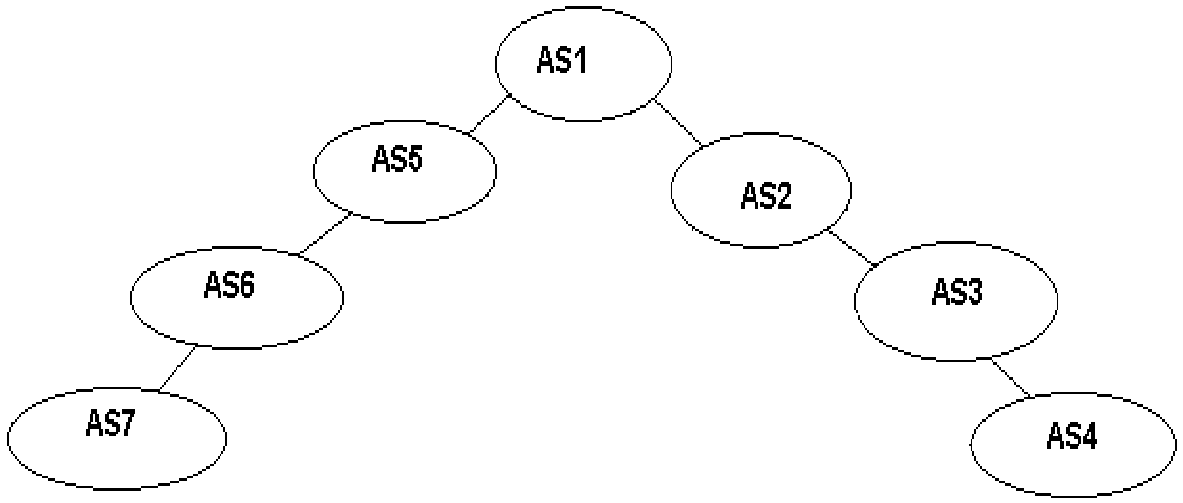
b. Perform the same computation for a host (i.e., not a router) in AS3.
2
1: default route as above
1: entry for local subnet

c. Perform the same computation for a router in AS1.
*7*N all networks in the internet*

d. Perform the same computations as in parts a and c, assuming CIDR is used.
(a) N+1 as before
(c) N+2 since all the networks accessed via AS2 can be represented by a single entry, and

e. Perform the same computation as in parts a, b, and c, assuming flat addressing. (Recall that flat addressing means there is no network component of the address, as opposed to IP hierarchical addresses which have both a network and a host component.)
(a) NH+1 Need an entry for every host in AS3 (NH) + default route
(b) H+1 Need an entry for every host in Network + default route
(c) 7NH Need entry for every host in the internet!

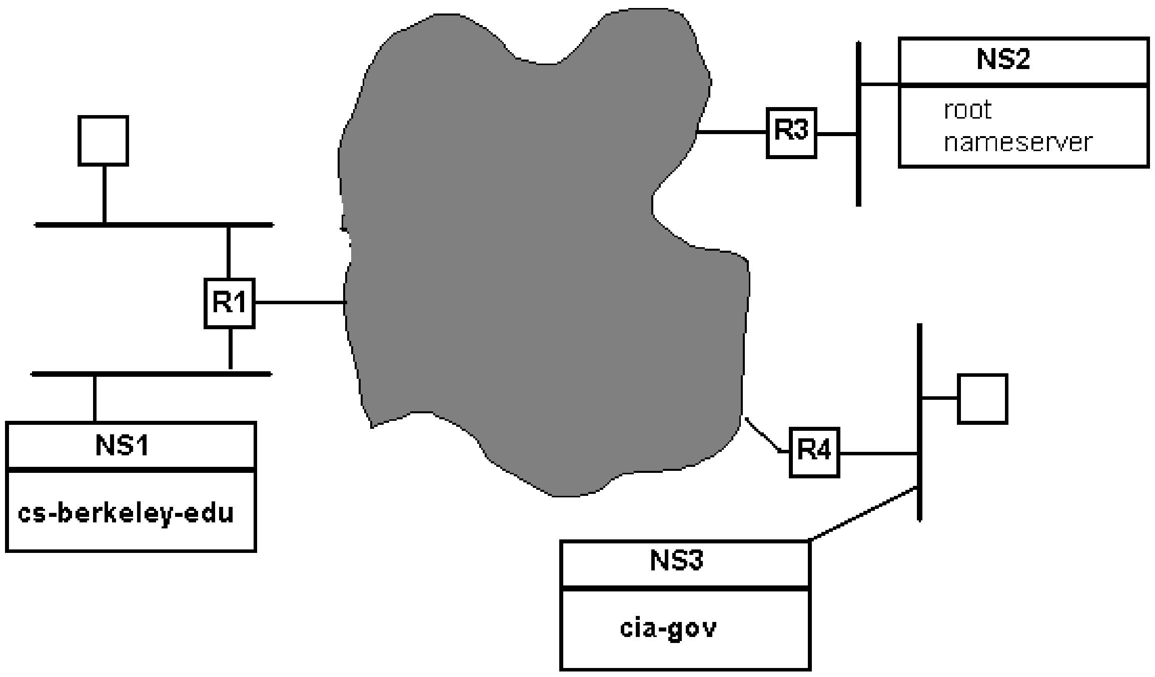
f. Which of your answers in parts a or c would change for the linear internetwork shown below? For the answers that would change, explain why.



(a) AS3 is now a transit network. It needs to know about all networks in AS3 and AS4, (2N), plus a default route (1)
2N+1
(b) Same as before 7N

4. Joe clueless has designed his own name system, Joe's Name System (JNS) which he claims is simpler and better than DNS. JNS is the same as the Domain Name System (DNS) except for the following changes:

- The system has only two levels of hierarchy (name.domain)
- If a domain is not used very often, it may be implemented by a single name server.



a. An application running on aladdin.cs-berkeley-edu wants to send a packet to another application running on sectrats.cia-gov. Besides the informations shown in the diagram above, what other information is needed ahead of time? (You should list information needed by aladdin, NS1, and NS2).
Aladdin: *IP address of at least one nameserver (NS1)*
NS1: *IP address of at least one root nameserver (NS2)*
NS2: *IP address of at least one nameserver for each domain, in this case cia-gov (NS3)*

b. List the steps taken to resolve the name, assuming that all caches are empty at the start.
1. Application on aladdin callls local JNS client
2. JNS client on aladdin sends the request to NS1
3. NS1 sends the request to NS2
4. NS2 replies with the name and address of NS3 as a NS for cia-gov
5. NS1 sends the request to NS3
6. NS3 replies with the IP address of secret.cia-gov
7. NS1 sends the reply to the JNS client on aladdin
8. The JNS client on aladdin returns the result to the application

c. If the application needed to resolve the name lies.cia.gov next, which steps listed above would be skipped?
3+4

d. List TWO ways in which DNS is better than JNS and briefly explain why it is better. (Note: We will grade only your first two answers!)
1. DNS is more robust
Each domain is implemented by at least 2 nameservers, while in JNS a domain can be implemented by a single nameserver. If that nameserver is unavailable, names in the domain cannot be translated
2. DNS is more scalable
It has multiple levels of hierarchy -> less state per domain

Root servers do not need to know about all domains, only the top level domains.
3. DNS is more flexible for administrations
domains can be created/subdivided by local administrators; e.g., berkeley.edu to cs.berkeley.edu or eecs.berkeley.edu

5. In the internetwork shown below, an application on the client wants to send a packet to the server. Table 1 lists actions that may be taken in the process of forwarding this packet.

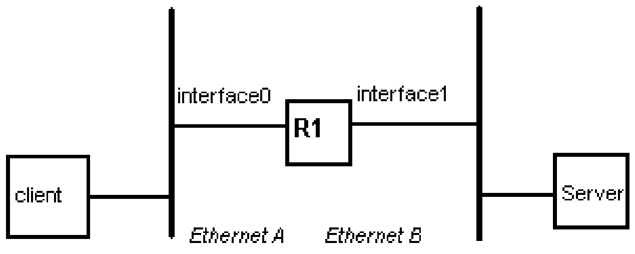


Table 1: Possible Actions

Action	Arguments to be specified
Lookup a mapping in an ARP cache	IP Address being looked up Corresponding Ethernet Address (if found)
Lookup the next hop in a routing table	Destination (IP address) Next Hop
Call ARP to map an IP address into an Ethernet address	IP address to be mapped
ARP sends Request (via Ethernet)	Destination Ethernet Address (where to send request) IP Address to be mapped IP Address of sender Ethernet Address of sender
ARP sends Response (via Ethernet)	Destination Ethernet Address (where to send response) IP Address that was mapped Corresponding Ethernet Address
Call Ethernet to send a packet	Destination Ethernet Address
Call IP to send a packet	Destination IP Address

a. List the actions taken to forward the packet from the client to the server. Specify addresses as Protocol:Host (e.g., IP:R1 or Ethernet:Server).
1. Application on client calls IP to send the request to IP:server
2. IP on client looks up next hop for IP:server -> IP:R1
3. IP on client calls ARP to look up physical address of IP:R1
4. NRP on client sends Request : <Ethernet:broadcast, IP:R1, IP:client, Ethernet:client>
5. ARP on R1 sends Response : <Ethernet:client, IP:R1, Ethernet:R1>
6. IP on client calls Ethernet to send request to Ethernet:R1

7. IP on R1 looks up next hop to IP:server->IP:server
8. IP on R1 calls ARP to get physical address of IP:R1 server
9. ARP on R1 sends Request : <Ethernet:broadcast, IP:server, IP:R1, Ethernet:R1>
10. ARP on server sends Response : <Ethernet:R1, IP:server, Ethernet:server>
11. IP on R1 calls Ethernet to send request to Ethernet:server

b. List the steps needed to send the response back to the client
1. Application on server calls IP to send response message to IP:client
2. IP on server looks up next hop for IP:client->IP:R1
3. IP on server calls ARP to get physical address
4. ARP looks up value in cache: IP:R1->Ethernet:R1
5. IP calls Ethernet to send response message to Ethernet:R1
6. IP on R1 looks up next hop to IP:client->IP:client
7. IP on R1 calls ARP to get physical address of IP:client
8. ARP on R1 looks up value in ARP cache IP:client->Ethernet:client
9. IP on R1 calls Ethernet to send response message to Ethernet:client