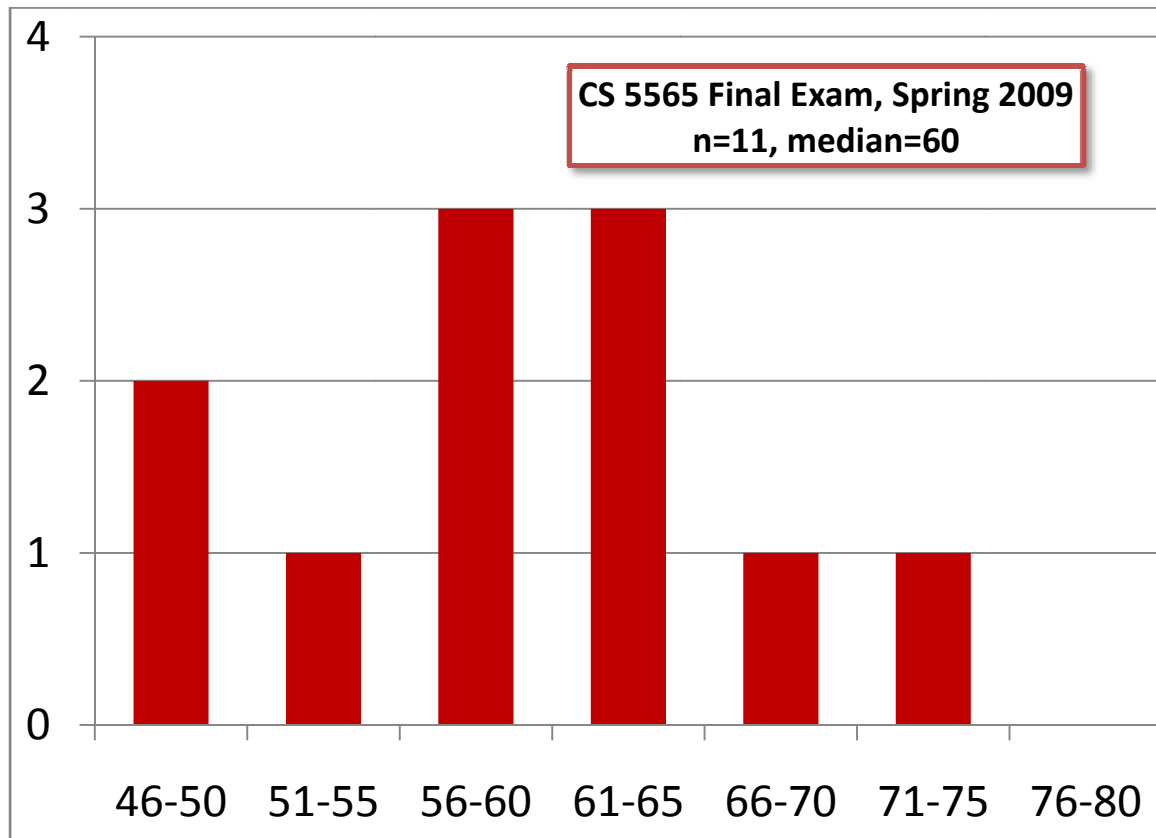


CS 5565 Final Exam Solution

11 Students took the final exam. The table below shows the distribution of points by problem. The overall median was low.

Problem	1	2	3	4	5	6	Total
Possible	12	24	20	20	12	12	100
Min	4	12	12	2	2	0	46
Max	11	24	19	15	12	10	72
Median	8	18	16	8	5	4	60
Average	7.5	18.5	15.5	8.5	5.6	4.0	59.8
Std Dev	2.2	3.3	2.8	3.7	3.1	3.2	7.6



Solutions are shown in this style.
Grading comments are shown in this style.

1 Error Correction and Detection (12 pts)

a) (8 pts) Hamming Codes

Consider the following (7,3) code¹, which maps 3 data bits to 8 code words A-H, listed below. The right half of the table lists the Hamming distance for each pair of code words.

	A	B	C	D	E	F	G	H	
A 0000000	: 0	4	4	4	4	4	4	4	A
B 0001111	: 4	0	4	4	4	4	4	4	B
C 0110011	: 4	4	0	4	4	4	4	4	C
D 0111100	: 4	4	4	0	4	4	4	4	D
E 1010101	: 4	4	4	4	0	4	4	4	E
F 1011010	: 4	4	4	4	4	0	4	4	F
G 1100110	: 4	4	4	4	4	4	0	4	G
H 1101001	: 4	4	4	4	4	4	4	0	H
	A	B	C	D	E	F	G	H	

- i. (4 pts) How many bit errors can this code detect?

Since the Hamming distance between each pair of code words is 4, it can detect up to 3 bit errors before a corrupted code word is mistaken for a different code word.

- ii. (4 pts) How many bit errors can this code correct?

It can correct 1 bit errors (just like the (7,4) code). To correct 2 bit errors, a Hamming distance of 5 is needed between each pair of codeword, so that exactly 1 codeword will be closest if a 2 bit error occurs.

b) (4 pts) Layering and Error Detection

Currently, error detection is commonly performed at the link-layer (via CRC-32) and the transport layer (via TCP/UDP Internet checksum). Is this redundant? Discuss the consequences of removing this feature from either layer!

- i. (2 pts) If error detection were implemented at layer 2 only:
- Then TCP/UDP could only be run over Ethernet or other link layers that provide error detection
 - Then errors introduced by failures at layer 3 (such as because of software- or hardware caused memory corruption in intermediate routers) might go undetected. (End-to-end principle!)
- ii. (2 pts) If error detection were implemented at layer 4 only:

¹ Created with Richard Tarvo's Hamming Code tool at <http://www.ee.unb.ca/cgi-bin/tarvo/hamming.pl?X=+Generate+&L=7&D=4&T=000>

- Then corrupted data might be carried end-to-end, long after the corruption could have been detected, resulting in wasted bandwidth and slower error detection and recovery.

2 Understanding IP Addressing (24 pts)

a) (3 pts) On Gateways and Netmasks

You're being asked to set up a machine for your research lab, which uses static IP addresses. The network administrator gives you the following information:

IP Address: 128.173.237.141
Default Gateway: 128.173.236.1
Netmask: 255.255.255.0

Can this information be correct? Justify your answer!

It can't be correct because the default gateway is not in the 128.173.237/24 subnet to which your machine is connected. (Likely, the subnet mask should have been 255.255.252.0 or smaller, or the default gateway should have been 128.173.237.1.)

b) (3 pts) Static DHCP

Your network admin tells you that they are using "static DHCP," in which your laptop will be provided with the same IP address every time it sends out a DHCP Discover request.

What information will the network administrator need from you?

Since DHCP Discover messages are broadcast at the link layer, the source MAC address is the only identifier by which the server can identify the source. Thus, the administrator will need your MAC address so that they can configure their DHCP server to respond with the same IP address whenever your machine requests one. It's also possible to trigger the assignment on some other unique identifier, such as a hostname provided by the user in the DHCP Discover request.

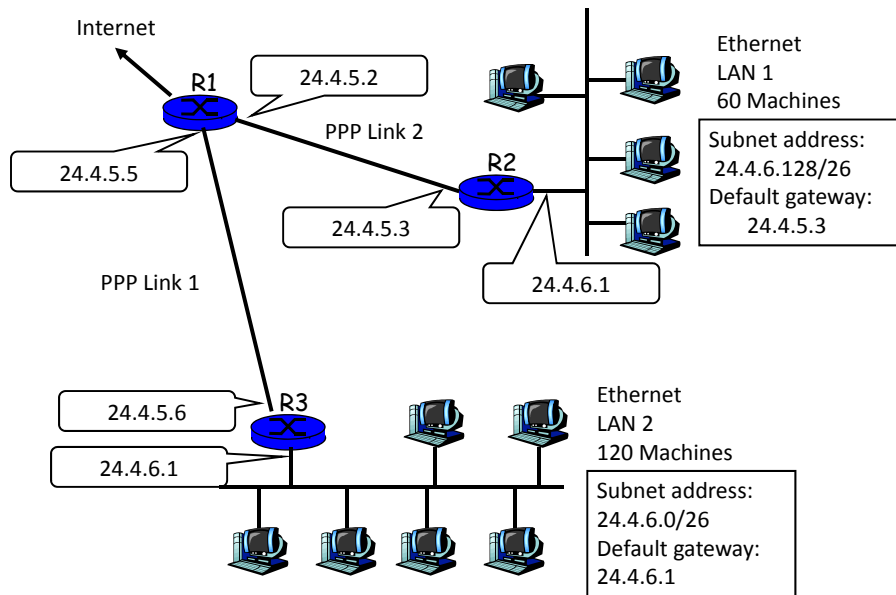
c) (4 pts) Prefixes vs. Subnets

Explain the difference between a prefix and subnet!

A subnet is a set of interfaces attached to hosts or routers that can communicate with each other without any intervening router. Prefixes are routing objects based on which router look up the next hop in the forwarding tables (FIB). Each subnet has a prefix, but not every prefix corresponds to one subnet because ISPs may aggregate multiple subnets into a single prefix.

d) (6 pts) On Subnets

You are being brought in as a consultant to debug the configuration of a company's network. The network consists of three IP routers connected by two PPP links to a border router connecting to the company's ISP. Each IP router serves a LAN of 60 and 120 machines, respectively. The current configuration is shown below; the rectangles on the right denote information that is distributed via DHCP in each LAN.



Identify 3 mistakes in this setup!

i. (2 pts) Mistake 1:

In LAN 1, the advertised default gateway refers to the outward facing interface of R2, it should refer to the inward-facing interface 24.4.6.1.

ii. (2 pts) Mistake 2:

In LAN 1, R2's inward-facing interface is not on the 24.4.6.128/26 subnet, making it impossible for hosts to reach.

iii. (2 pts) Mistake 3:

LAN 2 needs to accommodate 120 machines, but a /26 can only accommodate 62.

A fourth mistake is that 24.4.6.1 is used twice.

e) (4 pts) IPv6

Suppose you ported your project 2 to be compatible with IPv6.

Discuss which changes you would have to make!

1. You would have to replace all data structures that refer to IP addresses with their IPv6 equivalents; for instance, if you used Java, instead of `java.net.InetAddress`, you would use `java.net.Inet6Address`.
2. Unfortunately, since the simulator assumes that IP addresses are 32 bit during the initial establishment of the outbound TCP connection, you will also have to change `init_node` to not use a `int32` type – you could use a `bytearray` instead.

Since IPv6 changes only the network layer, no other changes should be necessary.

f) (4 pts) IPv6 Adoption

IPv6 adoption is lagging because IPv6 is said to have a “chicken-and-egg” problem. Explain what this problem is!

The key problem is that IPv6 hosts can talk only to IPv6 hosts, and IPv4 hosts can talk only to IPv4 hosts. As long as there are few clients attempting to communicate via IPv6, content providers will not set up IPv6-reachable servers. As long as there is little content available via IPv6, clients do not have an incentive to use IPv6.

I accepted other problems related to IPv6 deployment as long as they were of a chicken-and-egg nature.

3 Accountable Internet Protocol (20 pts)**a) (2x3 pts) Lack of Accountability**

The paper by Anderson et al laments a “lack of accountability” at the current network layer. Name two attacks that can arise from this shortcoming!

1. Route hijacking attacks in which a rogue party advertises a prefix to a destination that it does not own, thus redirecting traffic away from the legitimate route.
2. Source spoofing attacks, such as denial of service attacks, in which hosts send out packets with made-up source addresses.

b) (4 pts) Preventing Spoofing

If the Internet used AIP, what would happen if a malicious host attempted to disguise as another host by using that host's EID address?

When the first-hop router or switch receives such spoofed packets, it would send a nonce to the source and ask it to sign it and returned the signed nonce to the router, which then verifies the signature based on the public key whose hash is the source address. Since the spoofing host is not in possession of the private key, it can't provide that signature, so the first-hop router will not forward packets coming from this particular MAC host. See Section 3.1, EID verification.

A key element of this answer is linking the use of public key cryptography to addresses, and requiring a signature only the rightful owner of the private key can provide.

c) (6 pts) Routing Scalability

AIP has been criticized because CIDR-like aggregation cannot be used.

- i. (2 pts) Explain why not:

Since AD addresses are hashes of keys, they do not contain common prefixes, as the collision-resistant hash functions used to compute the hashes produce randomly distributed bit patterns containing roughly equal numbers of zeros and ones.

- ii. (2 pts) Explain what potential negative impact results from this:

With no aggregation, the size of the forwarding tables (FIB) will grow – it must contain as many entries as there are ADs.

Some of you restated the question that “CIDR-like aggregation cannot be used.” That’s not the impact.

- iii. (2 pts) Can the use of multiple AD: prefixes ameliorate the problem?

Potentially, yes. A large organization could create a hierarchy of ADs, and advertise only a single AD publicly.

d) (4 pts) Forwarding Performance

Would AIP, as proposed, work well with the hybrid hardware/software approach proposed by Casado et al in the “Rethinking Packet Forwarding Hardware?”

Justify your answer!

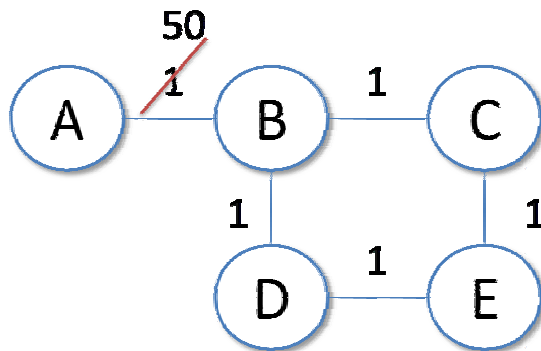
Likely, yes. AIP performs a flat lookup, which the content-addressable memory assumed in Casado’s paper can perform well. In addition, the amount of state needed to make a forwarding decision is well defined, avoiding the problems with dependency identification discussed in sections 4.1 and 4.2 of the paper. Note that Anderson’s paper already points out that flat lookups are cheaper in conventional routers than prefix-based lookups, requiring fewer memory accesses.

A valid concern is whether AIP would achieve necessary hit rates; we have insufficient data to judge that.

4 Routing Algorithms (20 pts)

a) (6 pts) Count To Infinity

The “poisoned reverse” technique can reduce the count to infinity problem, but not eliminate it in all cases. Give a concrete topology, and a concrete link cost change, for which count to infinity would occur even if poisoned reverse is being used!



In this example, if the link cost between B and A changes from 1 to 50, B must accept the higher cost. However, D will advertise to B a route with cost 4 (via D-E-C-B-A), unaware that B is on that route. Poisoned reverse doesn't help since B is not the first hop on the route D believes it has to A.

The question asked for a topology and link change, which many of you missed. Note further that the phenomenon will not occur in any ring topology.

b) (4 pts) Route Oscillations

- i. (2 pts) What are route oscillations and how can they arise?

Route oscillations can arise when link costs are related to the amount of traffic on a link. High traffic links increase in cost, favoring routes that avoid those links. This effect increases traffic along those routes, which in turns increases their cost and makes the original routes more attractive, thus leading to a cycle in which traffic patterns swing (or oscillate) between the two routes.

- ii. (2 pts) In lecture, we had discussed oscillations in the context of link-state algorithms.

Can oscillation also occur when a distance vector algorithm is used?
Justify your answer!

Absolutely. Oscillations can occur in any routing scheme that relates link costs to traffic if nodes perform simultaneous or near-simultaneous updates of their routing tables. See the discussion in the book.

c) (4 pts) Trade-offs

In project 2B, you needed to decide on suitable time intervals in which to a) check for cost changes and b) poll your neighbors to identify whether a link is up

or not. For each of these two parameters, describe the trade-off involved in choosing an appropriate value.

- i. (2 pts) The trade-off involving the Link Cost Change Polling Interval:

The longer the interval, the longer the period of time in which traffic may flow along non-optimal routes. The shorter the interval, the more CPU overhead is incurred, and the more routing traffic is sent to propagate routing updates, such as link-state advertisements or distance vectors.

- ii. (2 pts) The trade-off involving Neighbor Polling Interval:

The longer the interval, the longer the period of time in which link failures are undetected, thus leading to the loss of all traffic whose route passes those failed links. As with link cost changes, shorter intervals lead to more signaling traffic and likely more routing traffic as well, particularly for short, intermittent link failures.

d) (6 pts) Broadcasting

Reverse path forwarding is a multicast or broadcast forwarding algorithm that forwards packets if and only if they arrived on the interface that leads to the shortest path to the source.

- i. (3 pts) If a network has $|V|$ vertices and $|E|$ edges, how many packets are being sent for each packet being broadcast?

Since each packet is sent along each edge twice, except for edges lying on the shortest path tree from the source, $2 * |E| - (|V| - 1)$ packets are sent.

- ii. (3 pts) By contrast, how many packets would be sent if a spanning tree based algorithm were used?

It's a tree, so $|V| - 1$. No surprise here.

5 Link Layer (12 pts)

a) (8 pts) Self-configuration

Most layer 2 switches deployed today include firmware that allows users to patch them together in an arbitrary topology, even one including redundant links. A spanning-tree algorithm sets forwarding policies that prevent loops. However, the resulting forwarding may not be optimal. An alternative would be to implement a traditional link-state or distance vector routing algorithm in each switch's firmware.

Discuss the merits of this idea! Assume that no changes to the Ethernet frame format, and no changes to the NIC controllers in end hosts are allowed!

It would certainly be possible, and lead to optimal routes. Lacking any operator input, switches could assume a cost metric of 1, e.g., number of hops. Since the end hosts would not participate, self-learning would have to be preserved, and frames for which no routing entries exist would have to be broadcasted. Since the Ethernet frames can't be changed - so no TTL field can be added - an algorithm - or enhancements - that avoids or minimizes routing loops should be used.

In practice, this approach probably wasn't used because many interconnected Ethernets either have little redundancy, e.g., they are already a tree, or the use of backup links occurs infrequently, or traffic engineering goals determine which links should be used, thus requiring the (often static) set up of forwarding policies.

For this question, I gave variable amounts of credit depending on how well I thought your argument discussed the idea. Many of your brought up issues that are likely non-factors – such as processing overhead, needing to send HELLO packets, etc. – all of these must already be handled in the existing spanning tree protocol in 802.1D.

Many confused optimal routing with minimum spanning tree. A minimum spanning tree does not provide optimal routing.

b) (4 pts) Switch Poisoning

What is switch poisoning, and how might an attacker benefit from it?

Switch poisoning attempts to force a switch to broadcast an incoming packet across all outgoing ports because the switch lacks the specific forwarding information for the destination MAC address. Since entries are added to the forwarding table based on learned source MAC addresses, an attacker could force the eviction of valid entries by flooding the switch with spoofed MAC addresses. The benefit would be the ability to sniff on such broadcasted packets, something that is ordinarily not possible when switched Ethernet is being used.

6 Optimizing RPC (12 pts)

RPC, including the simple implementation you created in project 2A, suffers from the same performance limitations as other stop-and-wait protocols. As we discussed in lecture, sliding window protocols provide better performance because they use pipelining. In this problem, discuss how to adapt your RPC implementation to allow for pipelining.

(8 pts) Can you accomplish this goal without changing the semantics of the program that is using your RPC middleware layer? If so, what changes, if any, would be required to your language binding? State your assumptions if necessary!

Pipelining will send multiple requests before the response to the first request has been received. This is possible only if the second request does not depend on the first response. Such lack of dependency can be identified if the RPC

middleware layer can identify when (and if!) the caller making the first request accesses the return value (or OUT parameters) of a call. Such identification can be done by changing the language binding and returning a handle rather than the actual value. When the caller accesses the data via the handle, the RPC system detects the dependency and stops pipelining.

For more details, see Ali Ibrahim, Yang Jiao, Eli Tilevich, and William R. Cook, "[Remote Batch Invocation for Compositional Object Services.](#)" Proceedings of the 23rd European Conference on Object-Oriented Programming (ECOOP 2009), July 2009.

(4 pts) Provide a concrete example from project 2B in which your approach would result in performance improvements!

A trivial example may be the forwarding of link-state advertisements or the sending of updated distance vectors. Most likely, you will have written a loop such as:

```
for (node n in neighbors) {
    write_msg(n, msg);
}
```

Here, the return value of 'write_msg' is never accessed, and it doesn't have out parameter. A more complex example may be when polling for cost changes:

```
for (port p in ports) {
    newlinkcost[p] = get_cost(p);
}

for (port p in ports) {
    if (linkcost[p] != newlinkcost[p].valueOf()) { // access to handle
        ... handle cost change ...
    }
}
```

Here, all calls to 'get_cost()' can be batched if get_cost() is changed to return a handle.

Many of you proposed a multi-threaded architecture for the simulator, which however doesn't address the question, which states that "the semantics of the program using the RPC layer" should not change. Here, the program is a discrete event simulator. Many of you didn't realize that in order to preserve the semantics, yet achieve overlapping, it's crucial to consider how and when the results of an RPC call are being used.