What you really want to know

Those concerned about the Comp 243 workload (and grades) should find this handout interesting reading. It should give you a pretty idea about the instructor's expectations.

First, you will find copies of last year's midterm† and final exam. The average grade for the midterm was about 85%; the average grade for the final, about 75%. Both exams were a bit too long. I'm planning on having shorter ones this year.

Next, you will find a list of all the team projects. This should give you a good idea of the size of a project. Incidently, almost every project had a few rough ends. If one of last year's project intrigues you, you might consider beginning where that project ended.

By the way, the course grades for graduate students were about one-half H's and one-half P's.

† Except for the figure for Problem 2 of the open part. I've lost it.

Old Midterm – October 24, 1988

Closed book section (50 points)

The exam is to be turned in at 2:50 pm. Work the closed book section first and turn it in before you consult your books and notes to work on the open book section. For the closed book section, write your answers on the exam itself. For the open book section, write your answers on separate pieces of paper.

University regulations require that you sign the following pledge on the first page of your turned-in exam.

I have neither received nor given any unauthorized aid on this exam. ____________________

Problem 1. (21 points–3 points each)

Give short definitions (one or two phrases or sentences) of the following terms.

virtual circuit

flooding (as in bridges)

checksum

well-known ports

baseband

maximum transmission unit (MTU)

duplex connection

Problem 2. (5 points)

What is the difference between circuit and packet switched networks?

Problem 3. (9 points–3 points each)

Give a brief description of what the following BSD system calls do at the user level.

`socket`(af, type, protocol)

`bind`(s, name, namelen)

`fork`()

Problem 4. (5 points)

What is the difference between a bridge and a repeater?

Problem 5. (5 points)

Give an example of a high speed network. What (within an order of magnitude) is its bandwidth (in bits per second)?

Problem 6. (5 points)

Why might one machine have two IP addresses?

Old Midterm – October 24, 1988

Open book section (50 points)

The exam is to be turned in at 2:50 pm. The closed book section should be turned in before you open your books and notes to work the open book section. For the open book section, write your answers on separate pieces of paper.

Problem 1. (12 points)

Consider the following modification to the Ethernet RARP protocol: A machine trying to obtain its IP address uses as the Ethernet destination address of its RARP request the multicast address `0x777777777777` rather than the broadcast address `0xffffffffffff`.

What changes would have to be made to RARP servers and clients to use this new destination address for RARP requests? (The ARP and RARP protocol packet formats are shown on page 55 of Comer. The Ethernet frame is shown on page 19.)

How would this change effect the efficiency of a network of machines?

Problem 2. (6 points)

What is the spanning tree embedded in the mesh of bridged Ethernets shown on the next page by the IEEE 802.1 algorithm? (You may just draw the tree in heavy lines on the figure and turn that in as your answer.)

Problem 3. (13 points)

ICMP, UDP, and TCP each contain their own form of an "echo" protocol.

Can a user process (under BSD) access all three different echoes? If so, how? Compare the difficulties of using each to discover if a remote machine is reachable?

Problem 4. (19 points)

Presently, if an IP gateway receives an IP packet that is too large for the next hop, the packet is fragmented. Let's consider the following alternative to fragmentation when the IP packet is a TCP segment.

If a gateway receives an IP packet too large for the next hop *and* that IP packet contains a TCP segment, the gateways splits the segment at the TCP level instead of the IP level; that is, the gateway generates several small TCP segments (each encapsulated in its own IP packet) rather than fragmenting the IP datagram.

Describe how the fields of the headers of the new, smaller TCP segments could be derived from those of the original longer segment. Ignore the problems of dealing with URGENT data. (The TCP header is shown on page 140 of Comer.)

Compare the efficiency and robustness of TCP segment splitting (as proposed by you) versus old-fashion IP packet fragmentation.

Final exam – December 14, 1988

Closed book section (33 points)

The exam is to be turned in at 12:00 NOON. Work the closed book section first and turn it in before you consult your books and notes to work on the open book section. For the closed book section, write your answers on the exam itself. For the open book section, write your answers on separate pieces of paper.

University regulations require that you sign the following pledge on the first page of your turned-in exam.

I have neither received nor given any unauthorized aid on this exam. ____________________

Problem 1. (12 points–1.5 point each)

Give short definitions (one or two phrases or sentences) of the following terms.

transceiver

stable storage

stateless server

network worm (or virus)

autonomous system

atomic object

code violation (as used in token ring specification)

free token

Problem 2. (5 points–1 point each)

In at most five words, on what did each of the following lecture in Comp 243.

Dan Pitt

Diane Pozefsky

Don Smith

Bobby Stam

Bob Warren

Problem 3. (2 points)

Why is the purpose of Sun's XDR (eXternal Data Representation)?

Problem 4. (3 points)

What BSD communication-related system calls would be required in programming a single server process that implements both the UDP and TCP `echo` service.

Problem 5. (2 points)

If an IP packet is routed into a cycle, how is this detected?

Problem 6. (3 points)

For the IEEE 802.3 (CMSA/CD) specification, describe one feature that relates to the physical hardware layer of the ISO reference model and another feature that relates to the data link layer.

Problem 7. (3 points)

For the IEEE 802.5 (token ring) specification, describe one feature that relates to the physical hardware layer of the ISO reference model and another feature that relates to the data link layer.

Problem 8. (3 points)

What happens if "ordinary" learning bridges are connected in a cycle?

What happens if "spanning tree" bridges are connected in a cycle?

Old Final exam – December 14, 1988

Open book section (67 points)

The exam is to be turned in at 12:00 NOON. The closed book section should be turned in before you open your books and notes to work the open book section. For the open book section, write your answers on separate pieces of paper.

Problem 1. (4 points)

How many possible Internet "class B" logical networks are there?

According to Internet addressing standards, what is the largest number of hosts a class B network can contain?

Problem 2. (4 points)

Name two different kinds of distance metrics used in routing algorithms. Give examples of routing protocols using these metrics.

Problem 3. (13 points)

Suppose you are given three (3) file servers, thirty-three (33) diskless clients, and a spool of Ethernet cable of infinite length. Further assume that each diskless client is equipped with one Ethernet interface card and transceiver while each file server is equipped with two Ethernet interface cards and two transceivers. Draw a *good* interconnection for these thirty-six machines assuming eleven diskless clients are assigned to each file server. [Note: You have *no* bridges.]

Further suppose you are assigned the IP Class C address 195.33.55.0 to use in your network. Describe how you would assign *subnets* within your network? Show the network number and subnet mask of each physical network.

Randomly choose a file server *and* a diskless client in your network. Show the routing tables in both of these machines.

Suppose `mercury.cs.unc.edu` and one of your file servers are interconnected into a two-machine physical network. How does that change your assignment of subnets? How does that change the routing tables in the diskless clients?

Problem 4. (4 points)

Typically, name server mappings of computer name to IP address timeout in about a week while ARP table mappings of IP address to hardware address timeout in about five minutes. Is the extreme difference between the length of the two timeout periods reasonable? Why?

Problem 5. (4 points)

Discuss the tradeoffs between having a caching name server on each workstation versus having one caching name server for an entire department of one hundred (100) workstations.

Problem 6. (10 points)

We looked at three different network file systems: Sun's NFS, RTI's FreedomNet, and ITC's Andrew File System. Describe how the differences among these file systems must influence the actions taken to implement the `close` system call.

Problem 7. (10 points)

Flooding algorithms are used both in the 802.1 standard for bridged LANs [Backes] and in the directory search function of SNA [Baratz *et al.*, pp. 421-422]. However, the flooding algorithm of LANs can only be applied to LANs in which a *spanning tree* has been embedded while the SNA algorithm can be applied to arbitrary topologies.

Explain why the SNA algorithm can be applied in these more general cases. Either outline an SNA-like algorithm to set up bridge forwarding tables in an arbitrary interconnection, via bridges, of Ethernets or explain why such an algorithm cannot be constructed.

Problem 8. (4 points)

In Gusella and Zatti's analysis of TEMPO, it is assumed that the interval between clock ticks is the same on all systems being synchronized. However, this is often not the case; for example, VAX ticks are 10 milliseconds apart while Sun ticks are 20 milliseconds apart.

Will the TEMPO algorithm work well when two systems have different clock tick intervals? If not, how should the algorithm be modified to work under these circumstances. Describe how Gusella and Zatti's analysis can be modified to justify your answer.

Problem 9. (4 points)

Briefly explain the tradeoff between increased concurrency and the increased chance of deadlock with which the ARGUS programmer must contend.

Problem 10. (10 points)

Suppose hosts X and Y want secure access to a few of each other's files using NFS based on the secure RPC authentication protocol described by Taylor. If a "spy" installs a machine S on the same LAN segment as X, what mischief can S perform without being detected? (Yes, S could "jam" the LAN, but not for long.) Can S impersonate X when talking to Y?

Now suppose that the spy has discovered that X and Y are not on the same LAN segment but are connected via a gateway G. If the spy replaces G with a spy gateway G_S, can G_S be used to impersonate X when talking to Y?

Previous Class presentations – December 5, 1988

Azuma and Low 2:03-2:12

VIEW interface

Our project is to develop a server/daemon that provides a high level interface for an application that wants to run programs on other machines across a network. It will enable an application to start programs on other machines, asynchronously receive messages and results from them, and abort them if desired. The interface hides most of the implementation details from the application, and although it has been written with the VIEW system in mind, it should be generic enough for other applications to use.

Luca and Rhoades 2:12-2:21

Evening at the chess club

A system that uses networked Sun workstations to simulate an evening at the chess club. Any number of players can play, watch other games or kibitz over the net. A multiple client/central server model is used with datagrams to exchange messages.

Bajura 2:21-2:26

Distributed Tree Evaluation

I have been working on a program which distributes the computation of a game-type tree among an arbitrary number of computers. The network game distribution is kept separate from the functions which evaluate each position in the tree, which is a function of itself and its direct children. A "primary" image accepts user input and starts working on solving the tree. Secondary images connect to the primary image and each other and poll for work to do when they don't have any. Work is distributed throughout the network with specific return addresses and returned to its origin when completed. The primary image reports the tree evaluation to the user and asks for a new problem.

Cheung, Poirier, and Uichanco 2:26-2:38

A Distributed Blackjack Game

This is a distributed version of the common casino game. It allows up to six players on different machines to play blackjack with a computer dealer. Each player is given a display showing the cards and bets on the imaginary card table. Each player starts with a set amount of money and the server keeps track of this amount between plays.

Menges, Singh, and Whitfield 2:38-2:50

An SGMP implementation for the North Carolina Data Network

The objective of this project is to implement SGMP to address the network management concerns of Microelectronic Center of North Carolina which monitors the internetwork gateways in the North Carolina Data Network. Particularly, the internetwork gateways in the Triangle universities were considered.

The Simple Gateway Monitoring Protocol (SGMP) is an application-layer protocol that allows logically remote users to inspect or alter management information for a gateway (or a bridge).

The number of sites on the internet has escalated dramatically, and as it continues to grow more complex the issue of network management becomes more serious. A typical function of network management is network problem determination, which includes problem detection and diagnosis. This information can be used by the person responsible for network management to make performance assessment, and planning and design decisions about the network.

Our implementation consisted of three components:

1 A bridge server that communicates with bridges using XNS.
2 A proxy SGMP server that translates between SGMP and individual bridge command languages. The proxy server communicates with bridges via the bridge server.
3 A client program that monitors multiple bridges concurrently, via SGMP messages between the client and the proxy server.

Previous class presentations – December 7, 1988

Jacobsen and Taylor 2:03-2:12

Network Monitor

This project will provide a graphic window-based monitor for the ethernet. It shows graphs of the total number of packets per second, as well as splitting the packets into groups. The groups can be by length, source, destination, or packet protocol.

Ross and Steele 2:12-2:21

Mac news reader

An NNTP based news reader for the Macintosh. Or MacRn. Or WindowReader. Or NewsCruise. Or NewsGroup-Soup (like the Mac “talk” program, AppleJam) by the Punchy and Judy troupe. [Ed: Heavily edited version of description submitted by Steele.]

Markas 2:21-2:26

Distributed Fault Simulation

Fault Simulation is used to determine the testability of given digital circuits. The fault simulators detect the faults that are detected from a particular test pattern in a given circuit. In the most commercially available simulators the above procedure is accomplished by injecting faults in the circuit and comparing the fault-free response of the circuit with the response under the presence of a fault. If the responses differ, then the particular fault is detected by the simulated pattern. In practical circuits the number of the faults can be extremely large making the fault simulation a time consuming process.

The purpose of this project will be to execute fault simulation in a homogeneous distributed environment. A number of server workstations, running in 4.3 BSD UNIX will be responsible of simulating disjoint sets of faults using identical test patterns, and reporting the results back to master process. Issues for balancing uniformly the workload in the available workstations, as well as reliability issues will be addressed.

Kelley, Owen, and Scheviak 2:26-2:38

A Sample Implementation of UDP/IP in User Space

We have implemented the UDP and IP protocols as a user process in order to further our understanding of the protocols and to experiment with the performance implications of changes to the various levels of the code.

Good, Rheingans, and Tebbs 2:38-2:50

RUSH - an enhanced RSH

Rush is an enhanced version of the rsh unix system utility. It sets up the user's environment and working directory on the remote machine to match the current environment and directory on the host. It also creates a psuedo-tty driver for the remote process so that it will work correctly with full screen programs such as emacs or vi. Another feature of rush is to select the best machine to execute the remote process on. Rush can respect the rights of private workstation owners. Using information from a configuration file it will restrict access to certain machines during certain hours or when a certain user is logged on. Rush then chooses the available machine with the lowest load-average.

Previous Class presentations – December 9, 1988

Neumann and Thakur 2:02-2:11

Distributed Mandelbrot Computation

The basic idea is that since each point of calculation of the Mand. set is independent, let's let a group of processors do a portion each of the entire area to be calculated. Thus each machine is assigned a region by a host and the results are sent to the host where all the regions are assembled into one glorious ensemble image (yeah!).

The host has a user interface which allows the user to specify the region of interest and the calculation threshold. The goal is to allow a fairly good sized image (256×256) to be manipulated in "real-time," which for us is within a few seconds, *i.e.,* less than hours.

Of interest also is at what point does the network congestion and overhead make adding additional processors pointless? What does the efficiency curve look like? What's the penalty for going "off sub-lan"? Is 42 really the optimal machine number? Does running this at 3:00 AM help?

Barkley and Shan 2:11-2:20

Comms

Comms is a tool that allows users on multiple machines to share a graphics window. From the users' viewpoint, the program operates as a master controlling one or more slave screens. A user passes control explicitly by making a menu selection. The system prevents lockout by passing control round-robin, so that each user gets a chance to control the window. This version allows any participant to terminate the session.

Brown and Hill 2:20-2:29

Located: A Daemon to locate users in the department

Located' is a locate daemon that polls the machines in the department using the RPC command *rusers* to gather information, and stores this information in a database for easy retrieval. Located, combined with it's associated client program *locate,* provides a mechanism for any user to query the location of a user on any Sun, Vax, or uVax in the department.

By typing `locate username,` you will receive a response in about one second, without unnecessary delays or network traffic from rusers packets. Since the polling process queries the machines on a regular basis, the only network load is from the rusers responses. This should obviate the need of any user to use the rusers or rwho commands to find people, thereby potentially reducing total network load.

Fitzgibbon and Rohlf 2:29-2:38

A Fast Fourier Transform Server for the Convex C220

We are implementing a server which will allow users on campus to send two dimensional data to the Convex C220 Supercomputer in Phillips Hall. The server will store this data on disk, perform an FFT on it, multiply it by a filter, or perform an inverse FFT on it. On the client side, the user will see only a set of C++ classes which interface with the COOL library and hide communications details. The server is being designed so that it can later be extended to three dimensional data, for the use of the biochemistry department.

Dratch, Haskell, Lee, and Skinner 2:38-2:50

Finding users on the UNC Internet

We are working on a system to assist users in finding the electronic address of a specified user within the UNC Internet system. The user gives us a login name/last name and an optional first name/initial and we generate appropriate ping/finger commands to the user-specified and/or program-generated possible computer systems. We receive the results of the ping/finger and report back to the user (either interactively or through a file) the success and/or failure of our attempts in an easily understood manner.