New Orleans JAZZED about Engineering Education

Paper ID #14425

Developing an Android-Based Layer 3 Switch as a Senior Project

Mr. Pat Smith, Oklahoma Christian University of Science and Arts

Mr. Pat Smith lives in Edmond Oklahoma. He has been married for 32 years to Dr. Virginia Smith, a leading expert and author in the field of higher education. He has a son who is a physician and daughter who teaches physics and chemistry. Mr. Smith enjoys long distance road cycling and board games. He gained his undergraduate degree at the University of Oklahoma in Computer Engineering in 1984. Following this he worked in industry for Conoco Inc, Du Pont, and Cisco Systems. At Cisco Systems he worked with Cisco customers designing and deploying core Internet designs and technologies. In 2005 Mr. Smith earned a master's degree in Computer Science from Colorado State and 1 year later left industry to teach engineering at Oklahoma Christian University. Mr. Smith's emphasis is in first year student success, mentoring young engineers, and data communications. He consistently scores well in student feedback and enjoys regular strong relationships with his students and classes.

Developing an Android-based Layer 3 Switch as a senior project

Abstract

A project-based course has been designed with a goal of developing an in-depth understanding of network systems and communication protocols. The course requires students to implement a full layer 1-3 router using protocols developed for the course which are based on Ethernet, IP, and RIP protocols. The router is deployed on an Android platform using Java. By implementing the software to support these protocols (along with support tables such as an ARP table) the student develops a strong understanding of the protocols, the interfaces between the various layers on the router, and how routing and forwarding work. The course is a senior elective and is also available as a dual listed graduate course.

Educational Objectives

A project-based course in network systems has been developed which requires students to develop a full layer 1 through layer 3 router. The instructor originally designed this course without a hands-on lab. The course attempted to teach the content through lecture, homework and a research paper. Based on both the instructor's personal experience, but also considerable research indicating that students learn better through active learning such as lab experiences [1], the instructor desired to add a lab experience. A hardware-based network system lab was investigated similar to the approaches suggested by both Comer [2] and Giladi [3], but the cost for implementing a hardware lab was prohibitive. It was at this time that a software project was designed to develop a layer 3 network system (a router) on an independent platform such as an Android tablet was conceived.

The resulting course has multiple educational goals including understanding:

- Responsibilities for layer 2 and 3 in an IP protocol stack;
- How protocol layers interact, especially between layers;
- How to implement a simple distance-vector protocol;
- How support protocols function in an Ethernet/IP protocol stack;
- Understanding how protocols drive requirements for hardware design in network systems.

Each of these goals is described in more detail in the following paragraphs.

Responsibilities for layer 2 and layer 3 protocols in an IP protocol stack

The students should understand the layer 2 and layer protocol responsibilities. Requirements for each layer are provided and students must convert these requirements into specific implementations. The responsibilities of each layer within the protocol stack are understood more clearly when the student must work through a requirements process that clearly defines specifically which tasks will be handled by each layer. Implementation of the requirements in the project further cements this understanding in the student's mind.

How the layers interact, especially interfaces between the layers.

The focus for this goal is on the interface between the layers. How do upper layers request services of lower layers? What information do upper layers require before requesting such services? How does a lower layer determine which upper layer protocol service to pass data up

to and what information must be passed along with this data? These topics can be covered in depth when developing and defining the interfaces between the layers.

Implements a simple distance-vector protocol
Most students who have taken a course in data communications have"book knowledge" of how a routing protocol works. In this course the student will deal with problems caused by timing of routing update generation and delivery, lost updates, maintenance of routing tables and how to build forwarding tables from routing tables. Finally, in the Lab Layer 3 Protocol the students must write the code to examine individual packets for destination addresses, search forwarding tables, and properly forward the packet – updating all appropriate fields in the packet such as Time To Live and Checksum fields.

Understand support "protocols" and data structures necessary to enable interaction.
There are several important support protocols that enable the smooth interaction between layers in an IP protocol stack. These include an ARP table along with routing and forwarding tables. The implementation of an Ethernet based protocol, an IP based protocol, and a switching engine cannot be accomplished without certain support protocols. A final goal in the course is for students to understand clearly how these support protocols work to enable the "primary" protocols to do their jobs.

Understand communication protocols implementation and hardware implications
Finally, the overarching goal for the course is to provide students with a clear understanding of how packet processing in network systems takes place and what the implications are for design of underlying digital hardware and systems. This is accomplished when the student develops an understanding of the common functions associated with packet processing such as error checking, list searching, data movement, and the common need for timers. At the same time it will become clear that network systems have little need for some features common to processor architectures such as floating point accelerators. An appreciation of network processors, switch fabrics, and the impact of choice of memory for various subsystems in a network system is gained by students in the course.

The remainder of this paper will discuss the simple protocols which are designed for use in the router, the design of the router's architecture and subsystems, and the development steps taken by students through the labs.

Lab Protocols used in the router

The protocols used in the router are lighter versions of the actual protocols they emulate. There is a simple layer 1 emulator called the Lab Layer 1 Protocol (LL1), an Ethernet based protocol called the Lab Layer 2 Protocol (LL2P), an IP based protocol called Lab Layer 3 Protocol (LL3P), and a simple distance vector protocol called the Lab Routing Protocol (LRP).

LL1 Protocol
The entire protocol stack for the lab routers is implemented on top of UDP/IP through the LL1 protocol. The job of the Lab Layer 1 protocol is to emulate a layer 1 medium delivery mechanism. In the router's layer 1 "medium" the transmission is achieved by mapping the LL2P

MAC address (covered below) to an IP address. This address is the IP address of the router or test device that should receive the Lab's transmitted protocol frame. A layer 2 frame is received by the LL1 daemon which then encapsulates this frame a UDP packet and transmits it to the target router.

The layer 1 function in the router is the only component of the software that interacts with the "real world" UDP/IP protocols. Thus the layer 1 protocol also is responsible for forking off a thread which monitors the UDP port for received "frames" from other student's routers. When a frame is received it is passed to the LL2P protocol daemon without modification.

LL2P Protocol

The Lab Layer 2 Protocol is very similar to Ethernet in the organization of the frame and in its delivery method. The frame (shown below in figure 1) contains a 3 byte destination MAC address, a 3 byte source address, and a two byte type field in the header. This is followed by a variable length payload and a 2 byte CRC field.

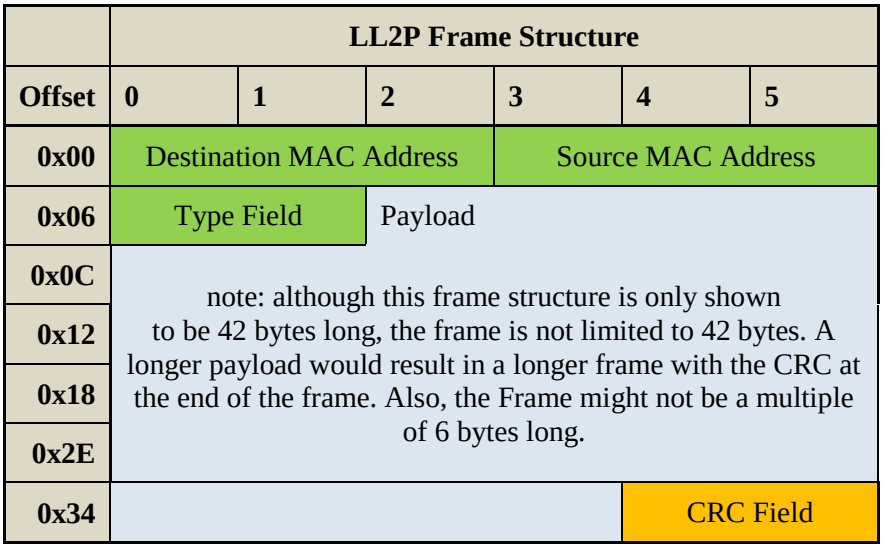

Figure 1 – Lab Layer 2 Protocol Frame Format

MAC Addresses are selected by or assigned to students at the start of the project. Students are asked to select a series of Hexadecimal codes that will be easy to remember, are non-offensive and are recognizable by the rest of the class (it's always interesting to see what they make up). The type field values are predefined well known numbers beginning with 0x8001 for LL3P and ascending for other upper layer protocols such as ARP and LRP. Students implement a CRC generator and validator for the layer 2 protocol transmissions and receipts. The CRC is created using a CCITT 16 bit polynomial applied to entire frame. The CRC is appending to the frame prior to transmission. Students who create working CRC generators in their routers for both creating and checking CRC codes they come to a much better understanding of how these work. A discussion is held in class of how CRC's can be calculated in hardware using simple feedback shift registers.

The LL2P protocol is a connectionless protocol. When a frame is to be transmitted it is passed to the LL1 daemon, which takes care of transmitting the frame. At this time a broadcast address is not implemented due to time constraints. The original design was to use a multicast IP address for broadcasts. During the first year several desired features had to be dropped and this was one of the ones that was trimmed from the router in order to leave time for other implementations.

LL3P Protocol

The LL3P protocol is similar to the IP protocol. It contains a 2 byte source and destination address. The first byte is the network number and the second byte is the host number within that network. Forwarding of packets is performed on the network number only and the host number is used for delivery to attached devices. The type field provides the ability to implement upper layer protocols, but at this time these are not used. The type field can have only one valid value, that of the application layer's Instant Messenger application (used to create LL3P packets). Each packet has a unique identifier (2 bytes long), and a 1 byte Time To Live field which must be decremented as the packet is forwarded through the network. The payload is variable sized and a 1's complement checksum is calculated over the packet using a method similar to IP's method of zeroing out the checksum field and placing the resulting checksum in the checksum field. Accordingly, when the TTL field changes the checksum must also change.

<table>
<tr><td></td><td colspan="6">LL3P Frame Structure</td></tr>
<tr><td>Offset</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
<tr><td>0x00</td><td colspan="2">Source LL3P Addr</td><td colspan="2">Dest LL3P Addr.</td><td colspan="2">Type</td></tr>
<tr><td>0x06</td><td colspan="2">Identifier</td><td>TTL</td><td colspan="3">Payload</td></tr>
<tr><td>0x0C</td><td colspan="6" rowspan="3">note: although this frame structure is only shown to be 42 bytes long, the frame is not limited to 42 bytes. A longer payload would result in a longer frame with the CRC at the end of the frame. Also, the Frame might not be a multiple of 6 bytes long.</td></tr>
<tr><td>0x12</td></tr>
<tr><td>0x18</td></tr>
<tr><td>0x2E</td><td colspan="6" rowspan="2"></td></tr>
<tr><td>0x34</td><td colspan="2">Checksum</td></tr>
</table>

When a LL3P packet is passed to the LL3P daemon the LL3P daemon must perform the normal error checking, destination checking and forwarding. These functions require the use of underlying processes and data structures including the checksum calculator, forwarding table and ARP table.

LRP Protocol

The Lab Routing Protocol (LRP) is a simple distance-vector protocol. It is responsible for building a routing table which contains all known remote networks, the number of hops to those networks, and the next hops to those networks. It also maintains a separate forwarding table (used by the LL3P forwarding engine) which contains only the current best routes to remote networks.

The LRP packet, shown in figure 2 below, contains a 2 byte Source LL3P address of the sending router, a sequence number to provide duplicate packet detection, a "count" of the number of network-distance pairs in the packet, and a list of network-distance pairs in the routing update packet.

Each record in the routing table contains fields for the source of the route update, the destination network, the distance to that network, and an age value which records how "old" the update is in seconds.

	LRP Packet Structure			
Offset	**0**	**1**	**2**	**3**
0x00	Source LL3P		Seq # / Count	(next row)
0x03	Net #1	Dist #1	Net #2	Dist #2
0x07	Net #3	Dist #3	Net #4	Dist #4
0x0B	Net #5	Dist #5	Net #6	Dist #6
0x0F	Net #7	Dist #7	Net #8	Dist #8
0x13	Net #9	Dist #9	Net #10	Dist #10
0x17	Net #11	Dist #11	Net #12	Dist #12
0x2B	Net #13	Dist #13	Net #14	Dist #14

Figure 2 – Lab Routing Protocol Packet Format

When the LRP protocol is introduced two timers are discussed. These are an update_value and a timeout_value. Each router must send an update to every neighbor every update_value seconds. When updates are received the routing table is updated and the age of the entry is reset to zero. The class discussion is around the relationship between these values. Questions such as what is a good update_value, what should be the relationship between these two values be are discussed. The students make a group determination about what the standard will be for all routers being implemented, and these values become the standard in the LRP protocols.

A separate threaded process is created to check the routing table at a regular interval which is related to the timeout_value and also determined by group consensus in the class. When routes "expire" they are removed from the routing table. This may result in a need to update the forwarding table. When this happens the forwarding table is updating using the new routing table.

From an educational perspective one of the more valuable processes that the LRP daemon must manage is the steps involved in the creation of the forwarding table from the data in the routing table. Whenever the routing table is updated with a new route the forwarding table might need to be updated.

At the same time students are implementing the LRP responsibilities in their routers the lecture material is focused on how the hardware (chip sets, memory, and processors) in the network system would support the routing protocols. By working through requirements created by the routing protocols the underlying hardware design choices are better understood. The need for efficient search algorithms, how different types of memory can help improve network system performance, timers, and timeout values now makes it clear why network system chips make the design choices that they make.

The work required to maintain coherent routing and forwarding tables students provide valuable insights into the workings of the routing and forwarding engines within network systems. It is during this portion of the course that discussions about the need for built-in timers and considerations of memory types (SRAM, DRAM, CAM, etc.) can be held with the class. Within the context of implementing these features in software the students are better able to grasp how these processes inform the requirements of network system hardware and network system silicon.

Router Architecture and Subsystems.

This paper now turns to the overall software architecture of the router application as implemented on the Android. Android development is accomplished primarily with the Java programming language coupled with a collection of XML files for describing external or environmental aspects of the application such as screen views and application meta-information (e.g. Permissions for example).

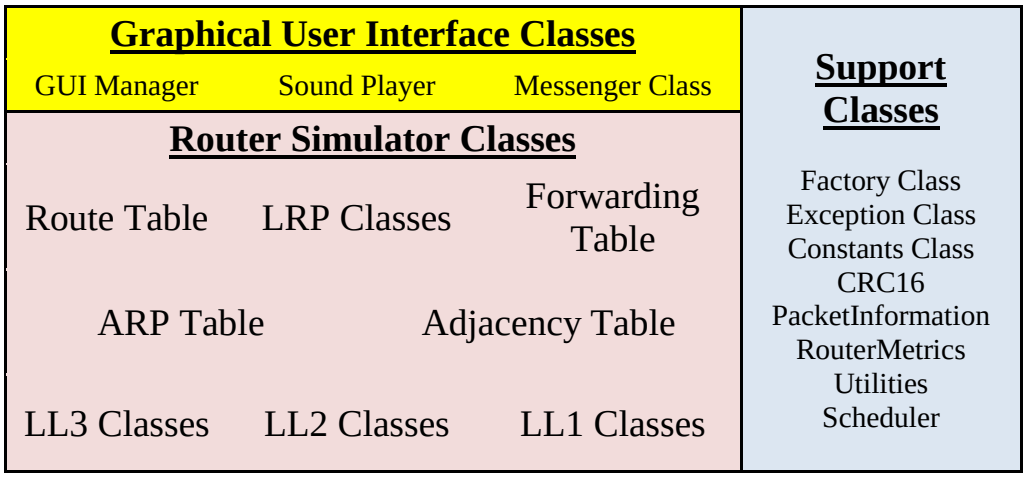

Figure 3: Organization of Packages in Lab Router System

The router consists of three packages of classes which the students must create over the course of the semester. These are shown above in figure 3. They are: (1) The View and Controller class in the Model-View-Controller pattern implemented in the router; (2) Support classes which are broadly reused across the router; and (3) the network classes which make up the data structures and daemons used to actually perform the routers network processes.

The first set of classes is a java package containing the user interface classes (this also contains the instant messenger class which is used later to "prove" the routers work). This class contains a Graphical user interface class which provides the view-controller component of the MVC architecture. This is strictly enforced throughout the semester. Students quickly appreciate the value of this approach to application development as the router's underlying complexity is separated from the user interface. The overall design means the application can also be easily ported to other platforms. As a side-note, the author has frequently considered implementing the router on a laptop since porting the application would be a simple exercise. However, in discussions with students one frequent comment is that developing the router on a standalone device such as an android device reinforces in the student's mind the idea that they are developing a separate device. Secondarily students often comment that they enjoy learning how to develop applications on the android platform. Because of these two anecdotal observations the course continues the practice of using the Android platform.

A second package is a support package containing classes which are used by multiple layers or are considered "utility" classes such as schedulers, checksum calculators, or table managers. These classes are "universal". For example there are several tables in the router (ARP, Routing and forwarding). These all have similar structures and features so support classes are implemented which can be reused by the protocol daemons. Static classes containing constants (timer values, field sizes, locally assigned addresses) are also placed here.

Finally the "meat" of the router is in the network package. All the protocol classes and daemons are in this class. Most protocols have a class to support the storage of a packet and a class to support processing the packet. Thus the network package contains classes which can represent individual frames and packets. This network package also contains the daemon classes which do the actual work of transmitting, receiving, and processing frames and packets. Thus there are classes which perform layer 2 processing, layer 3 processing, and respond to router updates, to name a few.

Course Structure

This course is a three hour course. The three hours are allocated to two hours per week of lecture and three hours per week in lab. There is one primary goal in the course. Students will develop a detailed and practical understanding of network system and protocol implementations. An ancillary goal is to reinforce certain software engineering principles in the area of code generation (as highlighted section 3 of the Software Engineering Body of Knowledge [4].

Lectures provide instruction in 3 main topic areas including (1) A model for Large-scale Network Architecture Design [5]; (2) Network System Architectures and Design [2] [3]; and (3) Software Engineering Topics as they impact the development of the router.

The primary goal of the labs is to provide students with an opportunity to begin the week's work with the instructor present to answer questions and clarify the week's development requirements. The three hours each week is typically insufficient for students to finish the development work assigned. Students typically finish the weekly lab work on their own time.

Lab Development Progression
There are fourteen labs sessions, each is 3 hours in length. Some of the sections are large and complex enough that the students require 2 weeks to complete the work.

For most students this is the first experience with Android application development. The first lab is extremely simple in its objective. In this lab students simply install the android studio IDE, locate and install drivers for their Android, and demonstrate they can build a simple "hello World" application on their Windows-based computer and then install and execute this application on the Android tablet they have purchased. In this class the students are permitted to purchase any Android-based tablet provided it has at least a 7" display and uses capacitive touch screen technology (resistive touch screens have proven troublesome). In a few cases some students encountered some trouble locating necessary drivers to enable their Windows-based laptop to communicate with their Android tablet. With one exception, every student has successfully enabled laptop-to-tablet communication within a couple days of the first lab. In the one experience where the student was unable to locate the proper drivers for their Windows-based laptop the Android tablet was returned and a new one was purchased.

In the second lab the software architecture is put in place. The support classes as well as the UI classes are created. Students demonstrate that they can update UI elements and respond to a single menu item selection.

The remaining 9 project assignments build the protocol stack from layer 1 through the application layer. Often the process of implementing a layer is divided into two separate assignments. In the first assignment the classes are put in place to support data structures such as frames, packets, or tables. In the second week the daemon is implemented to use the newly completed classes. What follows is a brief description of each week's primary goal:

Lab #	Goal
3	Create LL2P frame class, CRC calculator, and screen display elements for inspecting LL2P frames.
4	Develop the Layer 1 daemon and the tables necessary to map LL2P addresses to IP addresses, UI elements to display adjacencies. This is a two week lab.
5	Develop the LL2P daemon process. This includes an "echo" feature implemented in layer 2 (similar to the ICMP echo request feature).
6	Develop the ARP function.
7	Create table classes that will be used by routing process for maintaining routing and forwarding table.
8	Lab Routing Protocol part 1 – Get the LRP Daemon able to add, remove, and maintain the tables.
9	Lab Routing Protocol part 2 – Get the LRP Daemon to send and receive routing updates with peer devices. A software tool is provided to the students, implemented in Java, which can be used as a peer to verify proper functionality of the routing function.
10	The Lab Layer 3 packets and daemon is implemented. This is the last protocol daemon implemented because it requires all other protocols, including routing, to be in place before it can be put in place. This is a two week lab.

11	One week is taken where no new work is assigned. Students who have fallen behind have this time to attempt to catch up prior to the final "demonstration" in lab 12.
12	In this lab the instant messenger is installed which uses the router to forward and receive packets. All students' routers are connected in a ring topology and each student must demonstrate a working router. Routers which are not working properly are removed from the ring.

This is a complicated and challenging project for the students. Those who complete it and are able to demonstrate functionality in the final lab period are usually extremely excited about their work and their accomplishment. Even those who are "close" usually end the course with an expression of satisfaction about what they'd done and what they've learned.

Course Feedback

Each time the course is offered students are surveyed regarding the course. The survey provides students with the opportunity to rate the effectiveness of the course on a Likert scale and also provide feedback using open-ended short answers.

The survey has 12 responses, most of which focus on the instructor. Three responses are particularly germane to the question of course content. These questions are discussed below. The average Likert score is compared against the score in other courses within the University as a whole. The responses are scored on a rating of 1 to 5 with 1 being "Strongly Disagree" and a score of 5 representing a response of "Strongly Agree".

The first survey response of note is "I am motivated to learn". The average response for this question was 4.76 against a university-wide average score of 4.09 on the same question. This is a strongly positive response to the course as a whole, indicating students are engaged and interested in the work.

The second question of note is "This course encourages independent thought." The average response was 4.75 against a university-wide average score of 4.38 on the same question. This also is a very strong response to the content of the course. Exactly what kind of independent thought is encouraged is a matter of further investigation.

The third question deals with a concern on the instructor's part regarding whether or not the workload is too much. The lab frequently takes 3 hours in lab and many students anecdotally report spending a significant amount of time working to finish each week's lab work. This is supported by the number of students seeking help during office hours throughout the week following the lab session. The score on this question was a score of 4.63 against a university-wide score of 4.30. This final response is lower than the other two responses highlighted above. It is still relatively strong and the instructor's response has been that the workload, while challenging, is appropriate.

Finally, the students are asked to provide open-ended responses to these questions related to the course: First, "What change could be made to improve the course?" The second question is "What do you like best about this course?"

For the first question the number one improvement requested is to "Smooth out the workload between the labs". Some students have specifically suggested that week 1 is too short and week 2 is too much work for one week. The instructor has carefully weighed this input but remains persuaded that although week 1 is an easy lab for many students, it is better to get everything working and do no development work in the first lab rather than press on with new content.

A second recommendation made by more than one student was to request that the instructor increase the weight of the lab in determining the final grade so the amount of time invested in the project is more appropriately represented in the final grade. In the first two years the lab was worth 20% of the final grade. This suggestion was appropriate and the project is now worth 40% of the final grade (the rest is distributed among a research paper, exams, and homework).

The second question, "What do you like best about this course" has been a source of encouragement that the course is meeting its goals of engaging students in active learning which most students are finding rewarding (even if the workload is high). Most students indicate in the short responses here that they enjoyed the labs even though the work was challenging. One student wrote, "It was challenging, but it pushed me to learn about network systems and become a much better programmer. It reinforced the skills I learned in Object Oriented Programming."

Overall the feedback from course surveys indicate that the approach taken is well received, provides an engaging challenge, and is providing a strong hands-on experience for active learning.

Summary

We have described how a course in network systems uses the implementation of a router on an Android platform to teach students a deep and detailed understanding of how protocols are implemented on network systems. While working through the details in the lab assignments corresponding lectures and readings apply this new understanding to the development of network system chips design.

At the end of the course the student has a clear understanding of the requirements, workings, and interactions of layer 2 and layer 3 protocols necessary to send, forward, and receive packets for upper layer applications and also how these can be implemented in hardware. The course has received strong ratings in feedback surveys by students during the course, and feedback after graduation (from graduates who went on to industry and graduate school) has indicated the course was extremely valuable in future endeavors.

The author of this paper is happy to provide further information upon request including samples of lab and lecture activities.

Bibliography

[1] – Feisel, L., Rosa, A., (2005) The Role of the Laboratory in Undergraduate Engineering Education, Journal of Engineering Education, January, 2005.
[3] – Comer, D, (2004), Network Systems Design Using Network Processors, Pearson Prentice Hall.
[4] - Giladi, R., (2008), Network Processors, Morgan Kaufmann
[5] – IEEE, Software Engineering Body of Knowledge (SWEBOK). https://www.computer.org/web/swebok/v3

[6] – Cisco Systems, Inc., Internetworking Design Basics, Chapter 2.
http://www.cisco.com/cpress/cc/td/cpress/ccie/ndcs798/nd2002.htm#33232