CSC 458H1: Programming Assignment 1: Frequently Asked Questions (FAQ)

General

Which files in the stub code can I modify?

The only files in the provided source that you may modify are `sr_router.h`, `sr_router.c`, and `arpcache.c`. You may, however, add your own files and update the Makefile to support them.

What is the best methodology for debugging my router?

Debugging the source code for logic errors, crashes and memory leaks should not be any different than any other program. The use of gdb is encouraged.

In order to facilitate debugging of network traffic handled by your router, the sr stub code supports tcpdump compatible packets sent from and coming to the router. It is highly recommended that you get comfortable logging packets and viewing the logfiles in tcpdump as soon as possible.

To log packets, use the –l command line option for sr to specify a log file. All packets will then be written to this log file.

```
$ ./sr -t 17 -l logfile
```

To view the log file using tcpdump use the –r command. It is also recommended that you use the –e command to print out headers, the –vvv command for verbose output and the –x command to print out the hex values of the packets. Output should look as follows:

```
$ tcpdump -e -vvv -x -r logfile

reading from file logfile, link-type EN10MB (Ethernet)
13:45:00.855765 4a:d5:84:dd:c4:ff (oui Unknown) > c6:e5:8c:e6:48:6d (oui Unknown), ethertype IPv4 (0x0800), length 98: (tos
0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
   10.0.1.100 > 192.168.20.2: ICMP echo request, id 2991, seq 1, length 64
      0x0000:  4500 0054 0000 4000 4001 5a9b 0a00 0164
      0x0010:  c0a8 1402 0800 3fd9 0baf 0001 9c70 0a52
      0x0020:  0eb1 0c00 0809 0a0b 0c0d 0e0f 1011 1213
      0x0030:  1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
      0x0040:  2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
      0x0050:  3435 3637
```

What are good references for dealing with protocol headers?

- Unix Network Programming, Stevens
- Network Sorcery RFC source book
- RFC 826 (ARP)
- RFC 791 (IP)
- RFC 792 (ICMP)
- /usr/include/netinet/ip.h
- /usr/include/netinet/ip_icmp.h
- /usr/include/netinet/arp.h

What kind of sanity checks should I perform on the packet header?

Your code must be stable (e.g. not crash) with any packet it receives. You should also discard packets that are obviously corrupted, e.g. if the IP version is not IPv4, if the packet length is negative or above the Ethernet MTU, etc.

What sort of routing entries do we need to support in the routing table?

You have to support two cases:

- Routes where the next hop is explicitly stated.
- Routes where the next hop is '0.0.0.0' which denotes that all matches to this route lie in the collision domain of the interface associated with the route. In this case, the next hop is the destination address of the IP packet.

Ethernet

Do I have to deal with the Ethernet preamble/crc?

No, `sr_handlepacket(…)` delivers Ethernet packets without the preamble and without the CRC. Likewise, `sr_send_packet(…)` expects an Ethernet packet without the preamble and CRC. All packets to/from these functions have the following header format:

```
 DST      SRC      FRAME       DATA
 MAC      MAC      TYPE
+-------+-------+-------+------------------+
|6 bytes|6 bytes|2 bytes| 46 - 1500 bytes |
+-------+-------+-------+------------------+
```

Does the Ethernet MTU include the header?

No, you can expect packets of up to 1514 bytes

ARP

What ARP functionality do I have to handle?

You have to handle both ARP requests and replies.

How do I implement the ARP cache? Is a static table sufficient?

A static, hard-coded ARP cache is NOT acceptable. You should have a sufficiently sized (e.g. 100) entry table that is dynamically filled with values when ARP replies are received. You do NOT have to implement any sophisticated search algorithm such as hashing or RB Trees, a linear search is perfectly acceptable.

How do we implement the ARP queue timeout? Should we use multiple threads or a timer?

You can take the simple solution and only check the ARP queue timeout whenever another packet is received. This is not entirely correct behavior as a packet could be in the ARP queue for a very long time. But usually retransmits will arrive soon after the first packet, it performs pretty well in reality.

Why are the ARP packets I get from `sr_send_packet(…)` 60 bytes instead of 42?

The minimum size for the data segment of an Ethernet packet is 46 bytes, whereas an ARP packet is 28 bytes. The OS pads an extra 18 bytes on to the end of the ARP packets to meet this minimum length requirement. You don't have to worry about adding padding when generating ARP packets it will be added by the OS.

How many ARP requests must I send to a host without a response before I send an ICMP host unreacheable packet back to the sending host?

5.

IPv4

Do I have to handle Multicast in my router?

No, you don't.

How do I have to deal with fragmented packets?

Just ignore fragmentation. Today on the internet fragmentation is actually pretty rare. Treat fragmented packets as normal packets and you should be fine.

Do I have to be able to handle IP Options?

No, other than that your client may not crash, if it receives a packet with ip options, you don't have to do anything with them. You don't have to parse them and it is acceptable to drop suck packets or generate incorrect checksums for them.

Do I have to check/decrement TTL in the IP header if the packet is addressed to me?

No, you don't.

Section 4.2.2.9 of the router RFC (1812) states:

"Note in particular that a router MUST NOT check the TTL of a packet except when forwarding it."

Checksums

Which types of checksums (TCP/UDP/ICMP/IP) do we have to verify/calculate?

There are three (3) types of checksums:

- IP Checksums – All IP packets have these and for any forwarded/generated IP packet you have to verify/calculate it.
- TCP/UDP Checksum – You should verify/calculate his for ICMP messages that you received/send (But NOT for ICMP message that you forward, these should be treated as IP packets only). For info on how to do this see RFC 792.

How do I write and test a function that calculates the IP Checksum?

The algorithm as well as sample source code is in Peterson & Davie, page 95. Be careful what parts of the package you calculate the checksum over. TCP and ICMP for example need to be treated differently.

A great way to test if your function is working is to run it on arriving packets. If you get the same checksum that is already contained in the packet it looks like your algorithm is working. If you use this test, make sure not to include the old checksum (the one that is in the packet when it arrived) in your checksum calculation.

ICMP

If I generate an ICMP message, what is the right TTL to use?

There is no "right" value, however 64 is a good choice.

What types of ICMP messages do I have to generate?

You should support the following:

- Host unreachable (if no host replies to ARP requests on the local network)
- Net unreachable (if no route to the destination IP exists)
- Port unreachable (if router receives a TCP or UDP packet addressed to one of its interfaces.
- Timeout (TTL of the arriving packet is one or zero)
- ICMP Echo reply (on receipt of an ICMP Echo request to one of the router's interfaces)

All other ICMP message that you receive you can ignore, however, you should forward any ICMP message that is not for your router.

Does the client have to reply to ICMP Echo Requests on all three interfaces or just on the primary interface?

The client has to reply to all ICMP requests destined to IPs owned by the router's interfaces.

Do I have to generate ICMP Host Unreachable messages for ICMP packets?

For regular ICMP packets (such as PING packets destined for an application server), YES, you do. For ICMP error messages (such as Host Unreachable messages), NO, you do not. While RFC 792 states that ICMP messages should not be sent about ICMP messages, later RFCs (particularly RFC 1812 and 1122) have qualified that point, stating instead that ICMP messages should not be sent as a result of receiving ICMP error messages. This distinction allow hosts to be notified in the event of an actual error (such as a PING packet being ignored), but prevents the network from being spammed with error messages flying back and forth in response to other errors.

What source address should I use when responding with a port unreachable?

Use the destination address of the packet you are responding to.

What is the timeout of a host unreachable?

Host unreachable should be sent after 5 failed ARP attempts. Host unreachable should be sent back to the client after 5-7 seconds of sending the first ARP.