Projects 2 and 3 have been graded. Use CMS for regrade requests.

Project 5 will be released this evening. Due date: November 20 at 11:59PM. **There will be no extension for this project.**

No supplementary lecture for project 5.

Next week: Project 6 lecture.
There are three kinds of death in this world. There’s heart death, there’s brain death, and there’s being off the network.

Guy Almes
1. Overview
2. Project Scope
3. Implementation Details
4. Testing your project
5. Concluding Thoughts
What is an “ad-hoc networking layer”?
What is an “ad-hoc networking layer”?

Ad-hoc networking enables wireless communication without the need for infrastructure.

What is it useful for?

- Removes infrastructure costs.
- Allows quick deployment.
- Potentially more reliable (no single point of failure).

Based on Dynamic Source Routing.*

*http://www.cs.cornell.edu/People/egs/615/johnson-dsr.pdf
What do you mean by routing?

- Packets that arrive at your machine may not be meant for you.
- Packets not meant for you should be routed to their destination.
- Insert a routing layer between the network and transport layer.
- Both minimsg and minisocket implementations should work on top of miniroute.
Our networking stack until now...

User Application

Minimsg | Minisocket

Network
Our networking stack after P5

- User Application
- Minimsg
- Minisocket
- Miniroute
- Network
What does this mean for the network headers?

- miniroute header
- minimsg/minisocket header
- data
DSR is a reactive protocol.

If a host does not know how to route a packet, it must discover the route.

- It does so by sending a route discovery packet.

A route discovery packet is broadcast to all hosts within proximity of a wireless signal.

Hosts will re-broadcast discovery packets if they are not the destination.

- The host will add itself to the constructed route.

The destination will send a unicast route reply packet along the reverse route.
Route replies

- Upon receiving a reply, the route will be updated into the cache.
- If the received route contains $A \rightarrow B \rightarrow C$ where $A$ is the source, $C$ is the destination, $C$ will flip the route to $C \rightarrow B \rightarrow A$ to send a reply to host $A$.
- Route cache entries expire in 3 seconds to prevent stale cache entries.
- Route discovery must be performed again upon route expiration.
  - Is there a better way to do this?
How does the protocol terminate?
- By tracking a time-to-live (\(ttl\)) value.
- \(ttl\) decrements by 1 each time a message is forwarded.

Set the initial \(ttl\) to \texttt{MAX_ROUTE_LENGTH}.

What happens when \(ttl = 0\)?

How do we prevent loops while broadcasting?
What do you need to implement?

- Convert calls in higher network layers to `network_send_pkt` to `miniroute_send_pkt`.
  - But your `miniroute_send_pkt` function may still need to depend on `network_send_pkt` for unicasts.

- Update the network handler.
  - Interpret the miniroute header.
  - Handle routing control packets.
  - Deliver packet as usual if the destination has been reached.†

†Strip off miniroute header before delivering packet up the network stack.
Routing Cache

- The cache must be able to hold `SIZE_OF_ROUTE_CACHE` entries.
- Routes are invalidated after a 3 second timeout.
  - Alarms may be used, but it can be done without.
- Access should be somewhat efficient, as you may increase `SIZE_OF_ROUTE_CACHE` to be large.
  - Aim for $O(1)$ access speed on operations.
  - Use a hash table with linear probing.
  - We have provided a hash function for network addresses.
**miniroute_send_pkt semantics**

- Allow only one routing discover request per destination on the network at any one time.
- Block threads if `miniroute_send_pkt()` was called and route isn’t in the cache.
- Discoveries for multiple destinations can be done concurrently, but...
- Multiple threads should not trigger multiple routing discovery requests for the same destination.
- Unblock all threads waiting on a route when that reply arrives.
Miniroute packets

Use the header format provided in `miniroute.h`:

- Pack fields into the structure.
- Append `minimsg` or `minisocket` messages (if necessary).
- Maximum overall network packet size is still 8192, so you may have to make adjustments in your P3/P4 code.
Write an Instant Messenger application that runs on miniroute.
- Read input from the user (look at `read.[ch]`).
- Add `miniterm_initialize` to your system initialize functions.
- `miniterm_read` will let you read from the keyboard.

Try running `network6.c` over miniroute.
- Test interoperability with your friends.
Trying out your implementation

- Open `network.h`.
- Set `BCAST_ENABLED` to 1.
- Set `BCAST_ADDRESS`:
  - 192.168.1.255 for ad-hoc network (see instructions for setting up an ad-hoc network).
  - x.y.z.255 for CSUGLAB.
- For debugging/testing in the CSUGLAB:
  - Set `BCAST_TOPOLOGY_FILE`.
  - Provide a topology file (see project description).
    - Test without wireless.
  - Use only in CSUGLAB.
If you are really interested in this stuff...

- Hybrid proactive/reactive routing protocols
- See Professor Sirer’s SHARP‡

Concluding thoughts

- Have some fun with this project.
- It’s much less work than P4.
- Computers in the CSUGLAB may get snapped up by other students near the deadline, so start early.
- Come see the TAs in office hours§.

§ some of the TAs get lonely