CS 415
Operating Systems Practicum

Ad-Hoc Routing and Applications

Oliver Kennedy
okennedy@cs.cornell.edu

Project 4 Comments

- Project 4 due date pushed back to the 19th.
- Project 4 Questions:
  - minisocket_send() behavior:
    - Block until the entire message is sent and acknowledged.
  - minisocket_receive() behavior:
    - Do not block; If there are not enough bytes in the queue to fill the buffer, return.
    - This means you need to implement a non-blocking read, or a will_recv_block() function.
    - Code for ACKs should NOT be located in minisocket_receive().

Project 5

- We've implemented unreliable networking.
  - We've implemented reliable networking on top of it.
  - But we want to scale up.
- Messages can only be sent to the local network.
  - Different entities have their own networks.
  - Can we join two networks together?
  - What if computers on different networks need to talk to each other?
    - A computer needs to know which route to take to another computer.

Routing

- IP: Each entity runs a router that discusses routes with other routers
  - Expensive infrastructure
  - Cant move between networks easilly
- 802.11: Each base station keeps track of which computers are connected and communicates with all the other routers.
  - Still a lot of expensive infrastructure
- IM Clients: Central agent knows routes to all computers, and all computers know a route to the central agent.
  - Same problems as before

Ad-Hoc routing

- Each computer is in communications range of N other computers.
- If a computer has a message for a computer not in range, it asks around for a path to the target.
- Flood the network looking for our target (like Gnutella)
  - Doesn’t scale well, but works well for small networks.
- We emulate a set of wireless nodes talking to each other in ad-hoc mode.

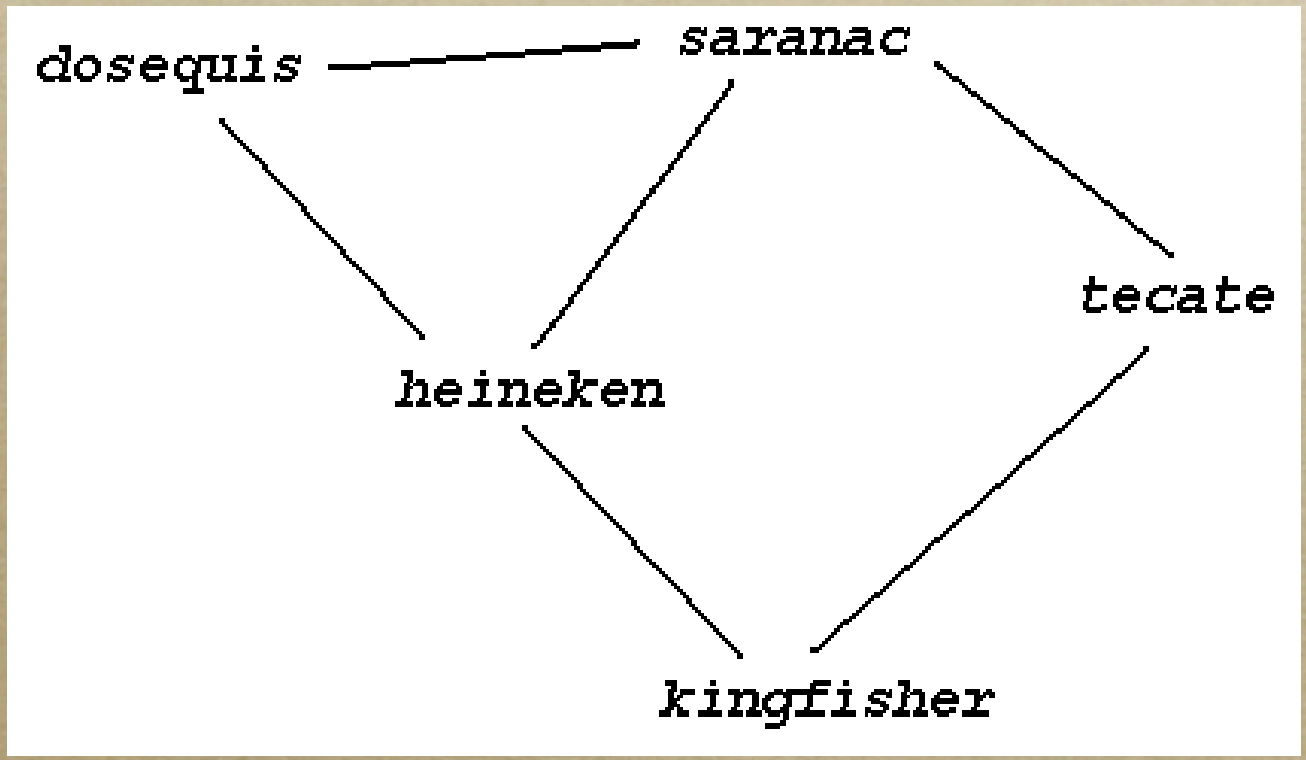
Setup

- In network.h
  - #define BCAST_ENABLED 1
  - #define BCAST_USE_TOPOLOGY_FILE 1
- Create a topology file

Topology File

```
saranac
heineken
dosequis
kingfisher
tecate

.xx.x
x.xx.
xx...
.x..x
x..x.
```



Implementation

- miniroute_send_pkt() (in miniroute.c)
  - Same interface as network_send_pkt()
  - Do we have a route to the target cached?
    - Yes: Send the packet to the next computer in the path
    - No: Find a route
- Finding a route
  - Use network_bcast_packet() to send a request to the network
  - Wait 15s for a response before timing out
  - Make 3 attempts before failing

The Route Cache

- Keep a data structure containing all known routes.
  - Linked List.
  - Fixed Size Array (with LRU replacement).
- When a route is discovered, add it to the cache.
  - Set an alarm to discard the route after 3 seconds.
- miniroute_send_pkt() should check the cache first.
- The routing protocol should ignore the cache.
  - This avoids cyclical and stale paths.

Changing your existing code

- Network Interrupt Handler
  - Should now forward route requests, responses and routing packets.
- Route Requests
  - If I'm the destination, send a response.
  - Have I seen this request before (sequence/host pair)?
    - If so, ignore it
  - If I'm not the destination, decrement TTL and rebroadcast to all peers.

Changing your existing code

- Route Responses
  - Am I the destination?
    - If so, add the route to the cache and wake up the sender thread.
  - Else forward to next peer in route.
- Packet Routing
  - Am I the destination?
    - If so, pass to UDP handler.
  - Else forward to next peer in route.
- Replace all calls to network_send_pkt() with miniroute_send_pkt().

The Spec

- The new layer MUST use the headers defined in miniroute.h
  - This way, you can route messages for other computers, even if those computers are using a different implementation of UDP.
- MAX_ROUTE_LENGTH must be 20.
- ROUTING_ROUTE_DISCOVERY
  - Each host should decrement ttl by 1 and put themselves in route entry M_R_L - ttl before forwarding.
- ROUTING_ROUTE_REPLY
  - The sender of the reply should just invert the first M_R_L- ttl entries of the discovery packet.
  - The recipient should do the same for the reply.

Overview

- Expanded Network Interrupt Handler
- New Send Packet
  - Change all code to use new send packet.
- Route Cache
  - ... with timeouts.
- And a test case...

Testing

- Write an instant messenger
  - Use reliable or unreliable connections.
  - Instead of using <stdio> primitives, use miniterm_read() defined in read.h
    - You'll need to call miniterm_initialize() in read_private.h
  - You should be able to communicate with any other computer running your software, even if direct communication is not possible.

fin