


CS 415: Operating Systems Practicum

Project 5: Ad-Hoc Routing and Applications

Oliver Kennedy
okennedy@cs.cornell.edu


Project 5

- We've implemented unreliable networking.
  - Now 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.


Types of Routing


Types of Routing

- IP: Each network runs a router that discusses routes with other routers.
  - Expensive infrastructure
  - Cant move between networks easily
- 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.
- Gnutella: Flood your peers looking for a particular service.
  - Inefficient.
  - Services must be highly replicated.


(mini)BGP style routing

- How IP does things (simplified).
- Maintain a table of all known peers and yourself (and the path to them).
- Periodically flood your presence over the network.
- When you receive such an ad, update your own table with any better routes.
- Discard routes you haven't seen recently.

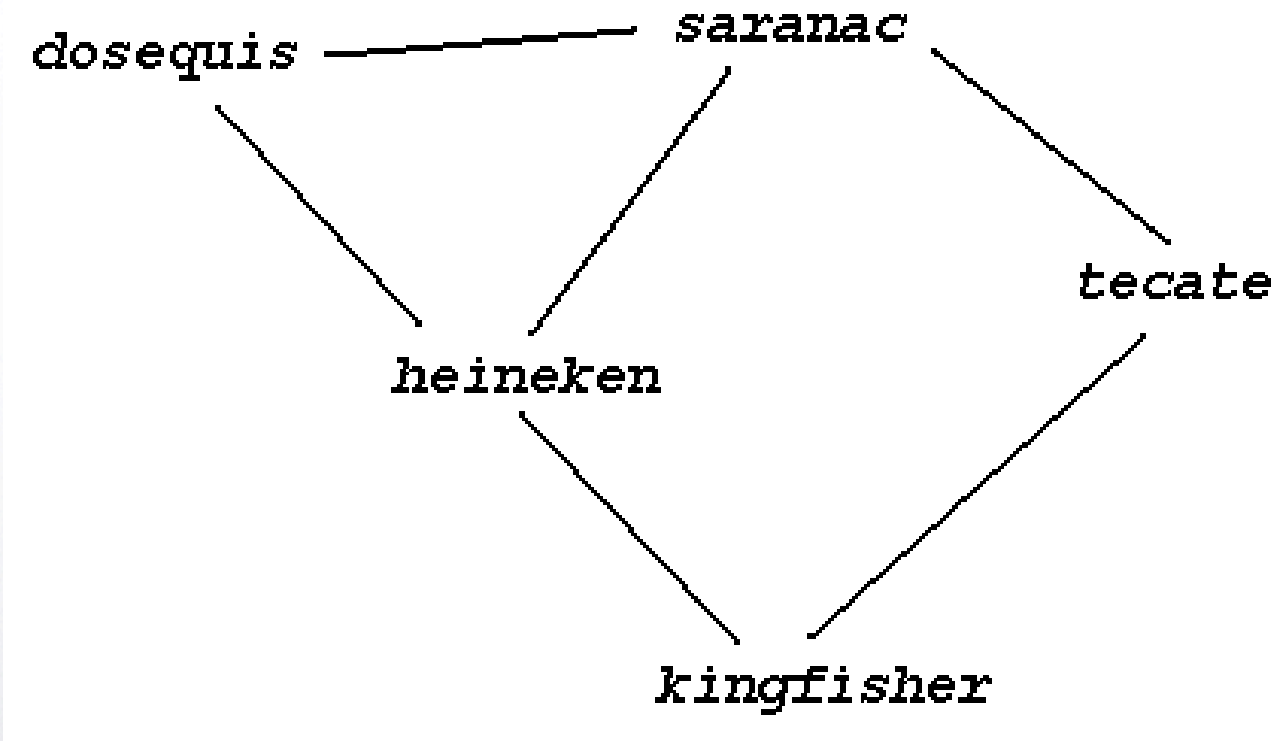

Setup

- 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 nexthop to the destination cached?
    - Yes: Send the packet to the next computer in the path
    - No: Error!
- New system thread: Flood your presence over the network every 5 seconds.
- New datastructure: routingtable.c


New Interrupt Handler


New Interrupt Handler

- What kind of message is this:
  - Unreliable data?
    - Am I the destination?
      - If so, pass to old handler.
    - Else forward to next peer in route.
  - Routing advertisement?
    - Am I in the path?
      - If so, stop processing.
    - Update the routing table.
    - Add me to the path and forward the message to my peers.


The Routing Table


The Routing Table

- routingtable.c


The Routing Table

- routingtable.c
- Routes time out after 30 seconds. (use alarms)


The Routing Table

- routingtable.c
- Routes time out after 30 seconds. (use alarms)
- Update it whenever you receive an advertisement.
  - Ignore paths with you in them.
  - Install shorter paths.
  - Reset the timeout if you re-receive a path.


Code Changes

- struct minimsg_hdr { ... int **msg_type**, ... }
  - We have message type 0: unreliable data.
  - We introduce message type 2: routing advertisement.
- Network Interrupt Handler
  - Start processing packets with the new handler.
  - Should now receive and process routing table messages.
- Change minimsg_send() to use miniroute_send_pkt()


The Spec

- The new layer MUST use the message types defined in miniroute.h
- MAX_ROUTE_LENGTH must be 20.
- As before, unreliable messages should have type 0.
- Routing table messages should have type 2.


Overview

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


Testing

- Write an instant messenger
  - 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