UiO : University of Oslo

IN4230 home exam 2

Autumn 2024

Course code: IN4230

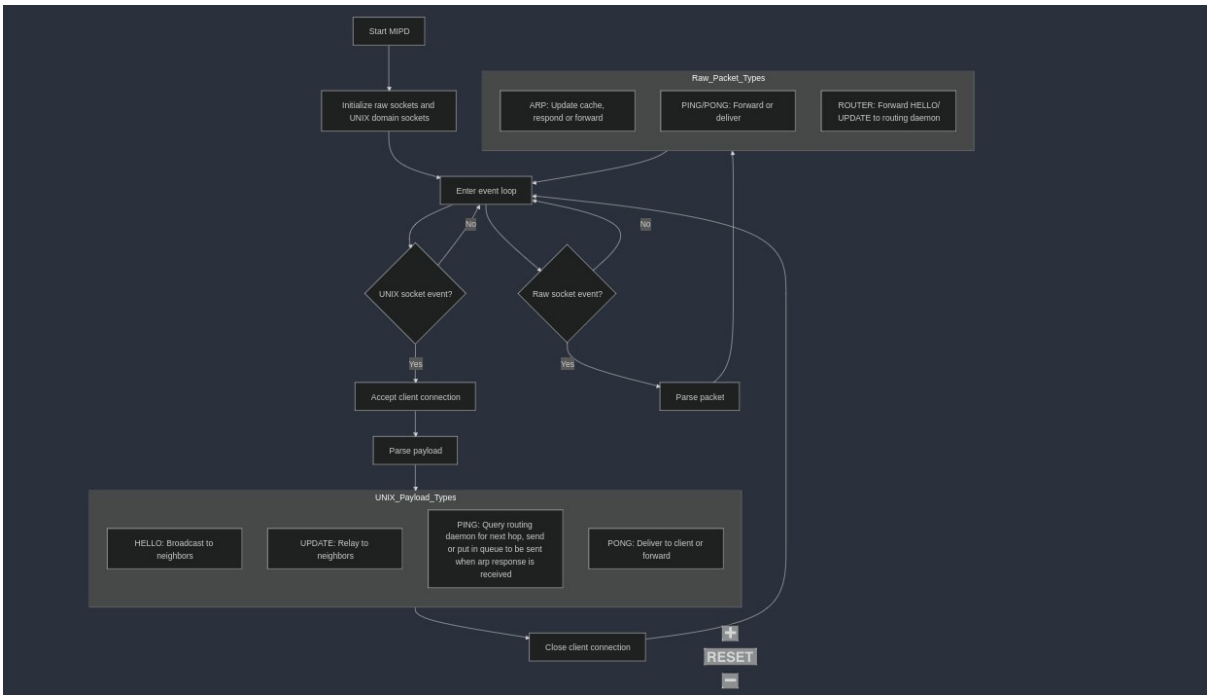
Candidate number: 15617

Word count: [Delete this and enter the word count]

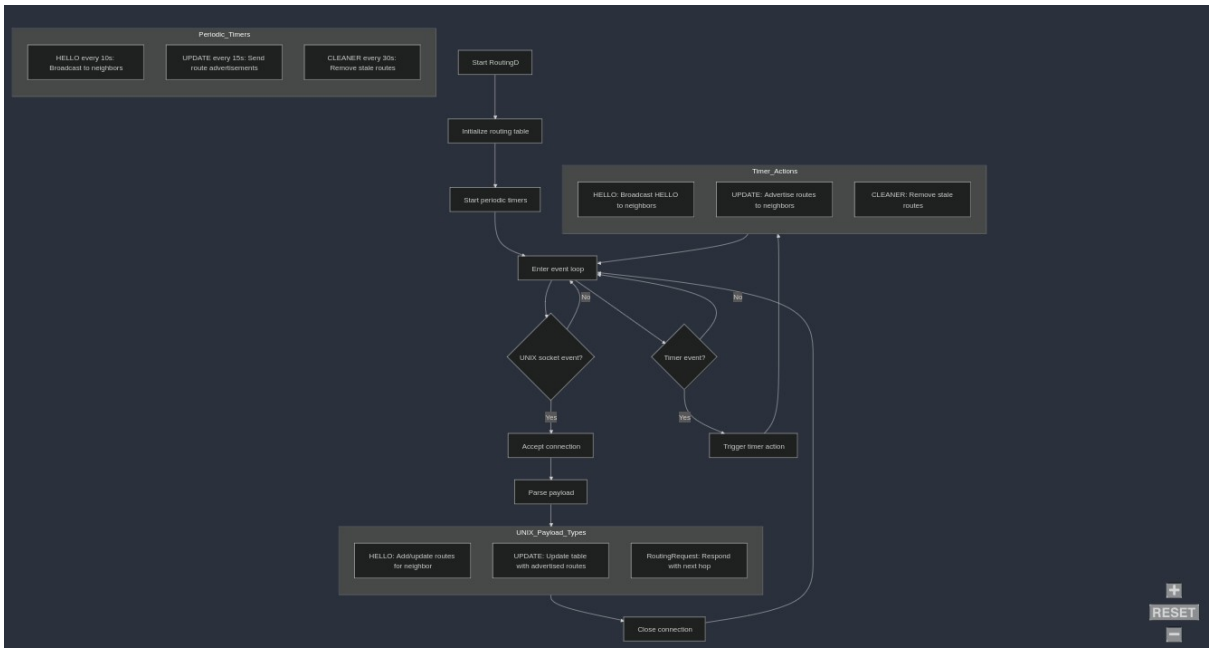
Date: 19.11.24

Flow charts

MIPD flow chart



Routing daemon flow chart



Routing protocoll specification

The routing protocol implemented is based on the “Distance Vector Routing Protocol”. It has been modified for the MIP protocoll. The router daemons decides the route, one hop at a time. Initially, all the routing tables are empty. The router has timers for sending HELLO and UPDATE messages. It also has a timer for cleaning up routes which has not been received in any HELLO or UPDATE messages after a specific amount of time. On startup, the router sends a HELLO request to the MIP daemon. The MIP daemon then sends HELLO messages as a broadcast to all neighbours. If a MIP daemon receives a HELLO message, it stores a route in the routing table to the src mip address of the HELLO message. When the router wants to advertise it’s routes, it sends an UPDATE request to the MIP daemon. The MIP daemon then broadcasts its UPDATE packet to all its direct neighbours.

The UPDATE packet sendt over the raw socket is a mip pdu with a ROUTER sdu type. The payload/sdu is as follows:

| 1 byte: packet type | + | 1 byte:src mip address | + | 1 byte: amount of routes to be advertized| + | route entries which each contains: 1 byte: destination MIP address + 1 byte cost| + …… until we have n routes|

When another MIP daemon receives the UPDATE message it sends it to the router daemon. The router daemon then loops through its own routing table looking for a better route. If a better route is found it is updated with a new cost and perhaps new next hop if it is from a different host. As all the routes in the table are checked, the clean timer on the route is updated so that it is not removed during the next CLEAN routine. If a HOST disconnects, it will get removed after some time thanks to the cleaning routine, however this might take som time and can be tweaked and optimized by fine tuning the parameters in the code. For now this cleaning routine is not fine tuned and may lead to some packet loss if routes are removed and messages sent to them to fast.

Describe the "Count-to-Infinity" problem in DVR protocols. How do you handle this problem in your own implementation?

Explanation: The "Count-to-Infinity" problem occurs in Distance Vector Routing when a route fails, and nodes propagate outdated or incorrect information indefinitely. For example:

- Node A loses a route to D.
- A relies on B's route to D (assuming B knows the path), causing a loop where the cost increments until it hits a maximum threshold.

Solution: In this implementation:

1. **Hop Limit (TTL)**: The protocol enforces a TTL for routes. If the cost exceeds the TTL, the route is invalidated.
2. **Poison Reverse**: When a route fails, the node advertises the failed route to its neighbors with an infinite cost, preventing loops.

What is the limitation of Split Horizon? How does Poison Reverse solve this issue?

Limitation of Split Horizon: Split Horizon prevents a node from advertising a route back to the neighbor it learned it from. However, this mechanism does not handle cases with indirect loops (e.g., three or more nodes involved in a loop).

How Poison Reverse Solves the Issue: Poison Reverse enhances Split Horizon by explicitly advertising failed routes with an infinite cost to all neighbors, ensuring that even indirect loops are broken.

For example:

- Node A learns a route to D via B.
- When the route fails, A advertises the route to B with a cost of infinity.
- This explicit "failure notification" ensures that all loops are immediately broken.