Algorithm:

1. Init empty database ‘network’.
2. Init empty database ‘controllers’.
3. Init the database handler process.
    1. Boot the database handler process container.
    2. Connect to ‘controllers’ database and get a cursor.
    3. Start a listening TCP server on port 12346.
4. For each controller in network:
    1. Boot the controller container.
    2. Find the IP of own interface.
    3. Connect to Database handler.
    4. Send startup message to database handler.
    5. Wait for acknowledgement.
5. For each controller connected to database handler as client:
    1. Receive startup message
    2. Try to update ‘controllers.controllers’ table.
    3. On success return acknowledgement to respective controller.
    4. On even one failure, return fail code, exit.
    5. Exit program after all controllers exit.
6. For each controller:
    1. Receive acknowledgement or failure code.
    2. Disconnect database handler.
    3. Exit if failure.
    4. Else Initialize the controller code.
7. Init the mininet topology script.
    1. Get the number of hosts and switches required.
    2. Connect to databases ‘controllers’ and ‘network’.
    3. Fetch the list of available controllers from ‘controllers.controllers’.
    4. Form randomized pairs of available controllers and switches.
    5. Connect random number of random hosts to each switch.
    6. Populate ‘network.(controller_ip)’ for each controller.
    7. Start mininet CLI.
8. Init the relay process, listen on port 12345.
9. For each already running and registered controller:
    1. Connect to database ‘network’
    2. Gain insight to available hosts using table ‘netowrk.(self_ip)’
    3. Connect to relay at 12345 port.
    4. Wait for a connection from relay at port 6666.
    5. Start listing for openflow packets from connected switch(es) at port 6633.
    6. For each packet received from switch:
        1. Check if destination host is available in ‘network.(self_ip)’.
        2. If available, add a flow in switch flow table and send back flow packet.
        3. If unavailable, fork and send query to relay on personal link and wait for reply from relay while simultaneously serving other packets.
10. For each query packet received from any controller on personal link by the relay:

    1. Check the packet header:
        1. If it is ‘broadcast’, then broadcast that query to all connected controllers on broadcast link, except the sender, and add to broadcast packet list.
        2. If it is ‘found’, the match the packet ID to previously broadcasted packet list and find the controller that queried it, and send the now received reply.
        3. If it is ‘end’ exit main loop and end connection with that controller.

11. For each packet received from relay by any controller on personal link:
    1. Check the receiving link:
        1. If it is broadcast link, check if the host queried exists in network. (self_ip):
            1. If yes, then send acknowledgement to relay using personal link.
            2. If no, then ignore.
        2. If it is personal link:
            1. Switch to a previously waiting process querying the host and add new flow.