Code report

Every node share the same algorithm and many functions with the common libraries.
The steps of the algorithm, as well as the functions (for example char* iptab_getaddr(node_id n)) are explained in the folder "/Main/Arp_headers/docs" (available only after the library download). The algorithm has been converted to code, in the main, using both the functions shared by everyone and those of my sub-group

void find_next(node_id my_id, handshake_t hsm,int first)

This function is used to find the next node and write to it in the initial phase of the algorithm, in the handshake step. It also contains the part of code needed to connect the client (int net_client_connection(char *IPaddr);).

handshake_t read_hsm(int sockfd)

This function, always used in the handshake step, accepts the client of the previous node and reads the message. The read struct will update your table.

node_id init_master(node_id my_id, int sockfd)

It is used by the zero nodes (which makes the turn leader vote for the first time) and by the turn leader. Make the vote, write the message to the next available node and at the end of the round read the message. Save the value of the winner that comes from the voting algorithm, reading it from msg via the functions vote_getWinner(&msg) and save it in the variable “winner from the functions”.

void init_general(node_id my_id,int sockfd)

It is used by nodes that are not zero or turn leader, in this case, the order is inverted: it reads first from the previous node and then writes to the next node.

read_msg and write_standard_msg are used instead for reading and writing messages (msg1, msg2 and msg3), in the first case there is an acceptance of the client and then a read, in the second case a request for client connection and then a write().
The deb() function, on the other hand, only prints the available table nodes, which is very useful to see who is available after updating the values at the end of the handshake.