Local Social Networking Protocol (LSNP)

CSNETWK Machine Problem 3T2425

Objective

Implement a Local Social Networking Protocol (LSNP) over UDP with core features defined in the RFC: peer discovery, messaging, file sharing, groups, and optional gameplay.

Deliverables

Source code + README

Deadline and Demo

13th and 14th week (to be announced by your instructor)

AI Usage Policy

Students may use AI tools (e.g., ChatGPT, Copilot) to help generate parts of the codebase or understand protocol requirements. However, all AI-generated code must be reviewed, tested, and verified by the student. The final submission should reflect the student's own understanding and ability to explain how the code works. Blindly copying output without testing or comprehension is not allowed. Submissions that show a lack of comprehension or appear to be largely untested AI output may receive reduced credit up to a grade of zero.

Groupings

Each group should have up to four members. All members of a group are expected to contribute meaningfully to the design, implementation, and testing of the project. Each participant must be capable of explaining all parts of the submission, even components they did not directly code. A detailed report of the tasks implemented by each team member should be recorded in the README or documentation. A sample of a Task Matrix is shown below. Additional tasks can be added or removed. In cases where uneven participation is suspected, instructors may request individual explanations. The group can drop any non-participating member in the submission.

Code sharing between different groups is strictly prohibited. While discussion of ideas, clarification, and protocol testing is encouraged, all source code must be the original work of the group submitting it. Any third-party libraries or tools used must be cited appropriately.

Violations of the collaboration and academic integrity policies—such as unauthorized sharing of code, plagiarism, or misrepresenting AI-generated work as entirely original—may result in a grade of zero for the project. In severe cases, such violations may also be referred to the appropriate academic disciplinary body for formal investigation. All students are expected to follow ethical guidelines and work within the permitted bounds of group work and tool usage.

Sample Equal Distribution of Tasks

Each member has:

- **5 Primary tasks**
- **5 Secondary tasks**
- **5 Reviewer tasks**

Task / Role	Member 1	Member 2	Member 3	Member 4
Network Communication				
UDP Socket Setup	Primary	Reviewer	Secondary	
mDNS Discovery Integration	Secondary	Primary	Reviewer	
IP Address Logging	Reviewer	Secondary	Primary	
Core Feature Implementation				
Core Messaging (POST, DM, LIKE, FOLLOW)	Primary	Reviewer		Secondary
File Transfer (Offer, Chunk, ACK)	Secondary	Primary		Reviewer
Tic Tac Toe Game (with recovery)	Reviewer	Secondary	Primary	
Group Creation / Messaging		Reviewer	Secondary	Primary
Induced Packet Loss (Game & File)	Primary		Reviewer	Secondary
Acknowledgement / Retry	Secondary	Reviewer		Primary
UI & Logging				
Verbose Mode Support	Reviewer		Primary	Secondary
Terminal Grid Display	Primary	Secondary		Reviewer

Message Parsing & Debug Output	Secondary	Primary	Reviewer	
Testing and Validation				
Inter-group Testing	Reviewer	Primary	Secondary	
Correct Parsing Validation	Primary	Reviewer		Secondary
Token Expiry & IP Match	Secondary		Reviewer	Primary
Documentation & Coordination				
RFC & Project Report	Primary	Reviewer	Secondary	
Milestone Tracking & Deliverables	Secondary		Reviewer	Primary

To ensure fair contribution and collaborative integrity within groups, all members are expected to fulfill their assigned roles as outlined in the Work Distribution Matrix. If a member consistently fails to complete their assigned tasks, the group must first attempt to resolve the issue through clear, documented internal communication.

If the issue persists, the group is required to notify the course instructor or facilitator with supporting evidence (e.g., chat logs, Git commits, progress summaries). The instructor may then take one or more of the following actions:

- Individual grade adjustment based on documented contribution.
- Exclusion from the group if no remediation is possible

The remaining members must absorb the remaining work or redistribute with instructor approval. **False reporting of the distribution of work may be treated as academic misconduct**, resulting in a grade of zero or a disciplinary referral.

Checking

Project checking and demonstrations will be conducted with multiple groups simultaneously to assess interoperability, correctness, and compliance with the protocol specification. Each group must ensure their implementation functions correctly in a networked environment alongside others, as this setup reflects realistic use cases. Groups are expected to actively participate during checking and be able to explain and troubleshoot their contributions. Failure to perform during this stage may affect the final grade.

Activity	✅ Allowed	❌ Not Allowed
Working with a group (2–4 people)	Yes, group work is supported	Groups larger than 4
Using AI tools (e.g., ChatGPT, Copilot)	Yes, for generating, explaining, or debugging code	Submitting AI-generated code without review or testing
Discussing protocol ideas with others	Yes, general discussion of LSNP concepts is encouraged	Sharing source code between groups
Testing with another group	Yes, for interoperability testing and debugging	Submitting shared or merged code from different groups
Using open-source libraries/utilities	Yes, with proper citation	Using uncredited third-party code
Submitting identical code as another group	No	Plagiarism, even with minor changes
Individual understanding of group submission	Each member must understand the entire solution	Relying solely on one member or AI without team involvement
Submitting reused AI responses from others	No	Reusing AI content from another person’s session
Acknowledging AI usage	Must include a note in README or code comments	Omitting any mention of AI assistance used

Grading Rubric (Total: 125/100 points)

To ensure steady and verifiable progress, students may only proceed to the next milestone after successfully completing the previous one. Each milestone builds upon the functionality of the prior stage. Skipping or bypassing a milestone without approval is not permitted and would result in a grade of zero past the milestone. Make sure you save your code for each milestone. This structure is designed to reinforce foundational understanding and maintain project integrity.

Milestone #1: Basic Functionality (35)

Category	**Points**	**Criteria**
Clean Architecture & Logging	5 pts	Modular code, readable structure, debug/log output.
Protocol Compliance Test Suite	10 pts	CLI or tests for crafting, parsing, and simulating LSNP messages. Should be able to turn on verbose and non-verbose setting.
Message Sending and Receiving	10 pts	The peer can send and receive messages at the same time
Protocol Parsing and Message Format	10 pts	Parses all LSNP messages (PROFILE, POST, DM, etc.) in key-value format correctly. The peer should be able to show a list of names (i.e. known peers and their display names) and all the valid posts and DMs by the said peer.

Milestone #2: Basic User Discovery and Messaging (25)

Category	**Points**	**Criteria**
User Discovery and Presence	5 pts	Broadcast PING/PROFILE every 5 minutes; responds to presence announcements.
Messaging Functionality	15 pts	Sends/receives POST, DM, FOLLOW, UNFOLLOW .

Milestone #3: Advanced Functionality (65)

Category	**Points**	**Criteria**
Profile Picture and Likes	10 pts	Correctly includes AVATAR fields and LIKE actions for posts.
File Transfer	15 pts	Handles AVATAR fields and FILE_OFFER, FILE_CHUNK, FILE_RECEIVED; reconstructs full file.
Token Handling and Scope Validation	10 pts	Validates token structure, expiration, and appropriate scope. There should be a way to store all message with valid token structure.
Group Management	15 pts	Implements GROUP_CREATE, GROUP_UPDATE, GROUP_MESSAGE; tracks membership locally. There should be a mechanism to print all groups the user belongs to, the members of a group, and print only incoming group message.
Game Support (Tic Tac Toe)	15 pts	Implements basic game state, move tracking, result detection.

AI Disclaimer

The protocol is primarily designed by the author. AI tools, primarily ChatGPT and CoPilot, are leveraged for the writing of the RFC. AI was leveraged to help formulate message structures and RFC formatting. Additionally, it supported the writing of the grading rubric, collaboration policy, and testing guidelines. All AI-generated content was thoroughly reviewed, validated, and adapted to meet the functional and educational goals of the project.