Peer Review Report

Reviewer: Abidul Kabir
ID: a1974976
Date: 16 October 2025 (Revised 18 October)

Summary

The project shows a good understanding of how to make secure systems and use cryptography the right way. The system works well; the code runs fine, and it performs as intended.

The review found a few small problems with security and reliability, mostly about how errors and exceptions are handled. These are not significant security risks, but fixing them would make the code easier to maintain and safer.

Tools and Testing Approach

Method	Tools / Process
Static Security Analysis	bandit -r .
Manual Code Review	Inspected node.py, crypto.py, and helper modules
Functional Testing	Ran node start-up and message exchange
Code Quality	Pylint

High-Level Findings

1. [Low] Silent Exception Handling.
2. [Low] Limited Input Validation.
3. [Low] No Central Logging / Monitoring.

Bandit Result

```
--------------------------------------------------
>> Issue: [B110:try_except_pass] Try, Except, Pass detected.
   Severity: Low   Confidence: High
   CWE: CWE-703 (https://cwe.mitre.org/data/definitions/703.html)
   More Info: https://bandit.readthedocs.io/en/1.8.6/plugins/b110_try_except_pass.html
   Location: .\node.py:628:8
627                     await self.inbox.put(frame)
628             except Exception:
629                 pass

--------------------------------------------------

Code scanned:
        Total lines of code: 1133
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0
                Low: 6
                Medium: 0
                High: 0
        Total issues (by confidence):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 6
Files skipped (0):
```

Finding Details and Impacts

Severity	Finding	Description	Impact
Low	**Silent Exception Handling**	Repeated use of try/except/pass may suppress important runtime events, including connection drops or malformed frame errors.	Implement structured logging using the logging module; record exceptions instead of passing silently.
Low	**Limited Input Validation**	Incoming frames are assumed to be valid JSON without deep type or length checking.	Add lightweight validation and sanity checks to prevent malformed input from causing instability.
Low	**No Central Logging / Monitoring**	The system lacks centralized error or event logging.	Integrate a simple logger with severity levels (info, warning, error).

Recommendations

1. Replace all silent exception handlers with explicit logging or targeted exception catching.
2. Implement structured input validation for message frames.
3. Add centralized logging for system events and errors.
4. Add docstrings and comments for crypto and networking functions.

Code Quality Review (Pylint Summary)

Overall Score: 8.81 / 10

Strengths

- Project structure modularised under the socp/ package (crypto, framing, messages, node, run_node).
- High readability and consistent coding conventions.
- Code compiles and runs cleanly with no critical or fatal errors.
- Logical separation of cryptographic, framing, and routing components matches SOCP architecture.

Issues Identified

- **Missing docstrings** in most modules and functions (C0114, C0116).
- **Broad exception handling** (W0718) in multiple files; hides underlying errors.
- **Excessive complexity** in core functions — too many local variables or branches (R0912, R0915).
- **Minor style warnings:** long lines, trailing whitespace, missing final newline, and import-order inconsistencies.
- **Unused / reimported modules** (W0611, W0404).

Functional / SOCP Gaps

1. Server bootstrap process – Missing dynamic handshake (SERVER_HELLO_JOIN, SERVER_WELCOME, SERVER_ANNOUNCE).
2. Presence gossip cleanup – USER_REMOVE broadcast absent on user disconnect.
3. Duplicate-message suppression – seen_ids cache not implemented
4. Heartbeat and timeout monitoring – No HEARTBEAT frames or 45-second connection timeout handling
5. Standardized acknowledgements – No ACK or ERROR message handling implemented.
6. Direct message verification – Lacks end-to-end signature validation of encrypted payloads