Peer Review Report

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

Summary

When the system runs normally, it does encryption, signing, and message sending the right way. But when the vulnerable mode is turned on, it adds some built-in weak points and backdoors on purpose.

Automatic scans (Bandit) found 13 small problems and no big ones. However, upon manual inspection, numerous serious security risks were identified when the vulnerable mode was employed.

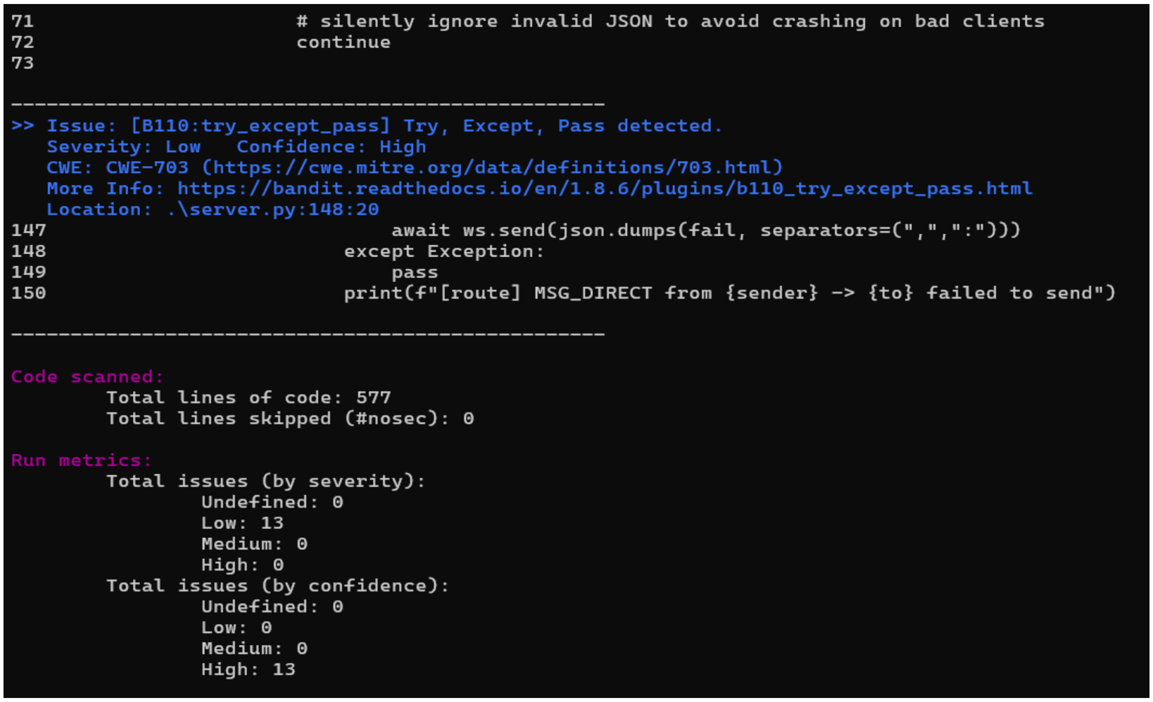
Tools and Testing Approach

Method	Tools / Process
Static Security Analysis	bandit -r .
Manual Code Review	Reviewed server.py, client.py, modules/crypto_rsa.py, config.py
Ethical Backdoor Testing	Ran in both normal and --vuln modes
Code Quality	Pylint

High-Level Findings

1. [Critical] Weak RSA Keys in Vulnerable Mode.
2. [Critical] Replay Guard Disabled in Vulnerable Mode.
3. [High] No TLS (Insecure Transport)
4. [High] Identity Registration Policy Too Permissive

Bandit Result



```
71                      # silently ignore invalid JSON to avoid crashing on bad clients
72                      continue
73

--------------------------------------------------
>> 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: .\server.py:148:20
147                         await ws.send(json.dumps(fail, separators=(",",":")))
148                     except Exception:
149                         pass
150                     print(f"[route] MSG_DIRECT from {sender} -> {to} failed to send")

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

Code scanned:
        Total lines of code: 577
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0
                Low: 13
                Medium: 0
                High: 0
        Total issues (by confidence):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 13
```

Finding Details and Impacts

Severity	Finding	Description	Impact
Critical	**Weak RSA Keys in Vulnerable Mode**	The --vuln flag sets MIN_RSA_BITS_VULN = 1024 in modules/crypto_rsa.py. This allows insecure 1024-bit RSA key generation.	Attackers can factor weak keys and decrypt all messages.
Critical	**Replay Guard Disabled in Vulnerable Mode**	client.py disables replay protection when config.IS_VULN=True.	Enables replay attacks — old ciphertexts can be resent and accepted as new.
High	**No TLS (Insecure Transport)**	Uses ws:// WebSockets without encryption.	Metadata and user identity leaks possible under MITM conditions.
High	**Identity Registration Policy Too Permissive**	server.py accepts arbitrary or duplicate user_id values without strict validation.	Impersonation or public key overwrite attack.

Recommendations

1. Remove or permanently disable the --vuln runtime flag for production.
2. Enforce minimum 2048-bit RSA keys and verify imported keys' sizes.
3. Implement UUIDv4 user identity checks and reject duplicate registrations.
4. Secure transport with wss:// using TLS certificates.
5. Replace assert statements with explicit conditional checks.
6. Add structured logging for all exception blocks.
7. Strengthen server-side message validation (timestamps, nonce tracking).

Code Quality Review (Pylint Summary)

Overall Score: 7.87 / 10

Strengths

- Code executes without major syntax or runtime errors.
- Logical structure between server.py and client.py is consistent and modular.
- Async handling and message parsing are clearly implemented.
- Overall readability is fair; indentation and flow are mostly clear.

Common Issues

- **Missing documentation:** No module/class/function docstrings (C0114, C0115, C0116).
- **Stylistic warnings:** Lines exceed 100 chars, trailing whitespace, and multiple imports on one line (C0301, C0303, C0410).

- **Code complexity:** server.py has large functions with too many branches/statements (R0912, R0915), which reduces maintainability.
- **Exception handling:** Broad Exception blocks in both files (W0718) may hide real errors or security faults.
- **Minor duplication:** Similar JSON-handling blocks in client and server (R0801).

Functional / SOCP Gaps

1. Bootstrap (Introducer Flow) not fully implemented; hardcoded links instead of dynamic SERVER_HELLO_JOIN / SERVER_WELCOME.
2. SERVER_ANNOUNCE broadcast absent — new servers are not formally advertised to others.
3. File transfer (FILE_START, FILE_CHUNK, FILE_END) functionality absent
4. Presence gossip incomplete — USER_REMOVE broadcast not triggered
5. Heartbeat (15 s) / Timeout (45 s) not enforced.
6. No transport-level signature verification between servers.