Review Secure Programming: By A1851002

I reviewed the code both manually and using a tool, the tool I used was bandit, bandit is a tool designed to find common security issues in Python code. I was able to eventually run the code but it did take a bit of trial and error as the instructions in the readme weren't particularly helpful or detailed on how to run it. However, the readme did do a pretty good job of describing their implementation and how it works. Here are some of the things I found while reviewing the code:

- The pyCrypto library and its modules AES, PKCS1_OAEP, RSA, and get_random_bytes are no longer actively maintained and have been deprecated. Consider using pyca/cryptography library.

- Insecure WebSocket Connection: The code uses "ws://" instead of "wss://", which means the connection is not encrypted. This could lead to man-in-the-middle attacks.

- Hardcoded Server Addresses: The client_to_server dictionary contains a hardcoded server address. This could be a potential backdoor if it's pointing to an attacker-controlled server.

- Weak Key Generation: The code uses get_random_bytes(32) for AES key generation. While not inherently insecure, it's better to use a key derivation function.

- Potential Backdoor in Message Handling: The unpack_message function doesn't properly verify the integrity of received messages before processing them.

- Lack of Error Handling: There's minimal error handling throughout the code, which could lead to unexpected behavior or crashes that an attacker could exploit.

- Very basic instructions for running the code.