## Methodology/ Planning of work

Securing communication in a Body Area Network (BAN) using Partially Homomorphic Encryption (PHE) involves a series of steps to ensure data confidentiality, integrity, and authenticity while maintaining the capability for certain computations on encrypted data. Here is a detailed methodology:

**1. System Setup and Initialization**

BAN Components Identification:

Identify the BAN components, including wearable sensors, personal digital assistants (PDAs), and gateways.

Define roles and data exchange requirements among these components.

Key Management:

Key Generation: Generate public-private key pairs for each device in the BAN using a PHE scheme such as Paillier or ElGamal.

Key Distribution: Securely distribute the public keys to all devices and ensure private keys are securely stored in their respective devices.

Encryption Parameters:

Define encryption parameters such as key size and encryption modulus, ensuring they meet the security requirements for BAN.

**2. Data Encryption**

Data Collection:

Collect data from sensors attached to or implanted in the human body. The data may include physiological signals like ECG, temperature, glucose levels, etc.

Data Preprocessing:

Perform necessary preprocessing on the sensor data, such as filtering, normalization, and sampling.

Homomorphic Encryption:

Encrypt the preprocessed data using the public key of the intended receiver (e.g., the PDA or healthcare provider).

Utilize PHE to encrypt the data such that specific operations (e.g., addition or multiplication) can be performed on the ciphertext without decrypting it.

Example with Paillier Encryption:

𝐸(𝑚1)=𝑔 𝑚1⋅𝑟𝑛 mod 𝑛2

E(m1)=g m1⋅r n mod n2

Where

E(m1) is the encrypted message,

g and 𝑛 are encryption parameters,

m1 is the plaintext, and r is a random number.

**3. Data Transmission**

Secure Channel Establishment:

Establish secure communication channels between the BAN devices using protocols like TLS/SSL to protect against eavesdropping and man-in-the-middle attacks.

Data Transmission:

Transmit the encrypted data from the sensor nodes to the PDA or gateway. Use the established secure channel for this transmission.

**4. Data Processing and Aggregation**

Encrypted Data Aggregation:

Aggregate encrypted data from multiple sensors at the gateway or cloud server, performing operations directly on the ciphertexts if necessary.

Partially Homomorphic Operations:

Execute allowed operations (e.g., summation) on encrypted data without decrypting it.

**5: Decryption and Analysis**

Decryption:

The intended recipient uses their private key to decrypt the aggregated or processed ciphertext.

Data Analysis:

Analyze the decrypted data to extract meaningful health insights and take necessary actions.

**6: Security Measures and Protocols**

Authentication and Authorization:

Implement strong authentication mechanisms and role-based access control (RBAC) to ensure only authorized access.

Integrity Checks:

Use cryptographic hash functions to ensure data integrity and verify hash values before and after transmission to detect tampering.

Audit and Monitoring:

Continuously monitor the BAN for suspicious activities or anomalies and maintain an audit trail of data transmissions and access events.

**7: Performance Optimization**

Efficient Encryption Algorithms:

Choose PHE schemes and optimize parameters for computational efficiency to minimize resource consumption on wearable devices.

Resource Management:

Effectively manage computational and power resources, especially for battery-operated wearable sensors.

**ECG**

**EEG**

**Protocol**

**Virtual Actuator**

**Sensor**

+get\_data()

+use\_data()

+start\_session()

+end\_session()

**Interpreter**

**Actuator**

**Real Actuator**

**DB**

**VirtualHand**

+set\_protocol()

+send\_data()

**Receiver**

+set\_protocol()

+send\_data()

**Transmitter**

+add\_sensor()

+delete\_sensor()

+start\_acquisition()

+stop\_acquisition()

+make\_data()

+on\_recieve\_data()

+\*Sensor

**Reader\_**

+get\_timestamp()

**Clock**

+add\_sensor()

+get\_data()

+\*Sensor

**SensorArray**

**BendingSensor**

**Movements**

**Blood Pressure**