Project Delivery

Protocol to Protect IoT Data

Mitul Tandon
s223588983

Problem Statement

There is a lot of data which is being produced these days. Due to the advancement of Internet of Things(IoT), there has been a lot of conidential data being produced via IoT devices. If this data is in some wrong hands like attackers, they can use this for illegal practices and affect the authenticity of the data generated by these devices

Approach

Step 1

I have used a python script to create a protocol that could protect the data generated by an IoT device.

Step 2

For this I have used external librarires of python AES encryption key

Step 3

Applied it on the sample data generated by the protocol

Step 4

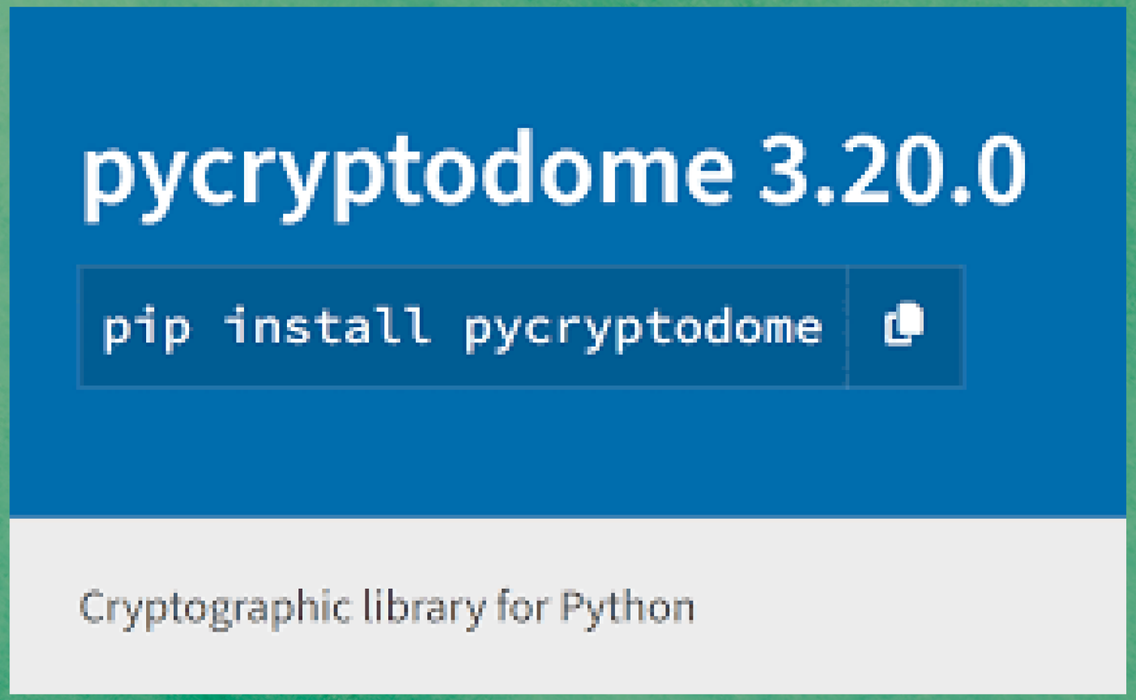
Get the results

Importing the External Libraries

After installing pycryptodome, you can run the provided code to encrypt and decrypt data using AES encryption

Importing Libraries:

Crypto.Cipher: Provides cryptographic ciphers such as AES.
Crypto.Random: Used to generate random bytes for the key and nonce.
hashlib: Provides cryptographic hash functions for key derivation.



https://shorturl.at/o1i4i

IoTSecurityProtocol Class:

__init__: Initialises the class with a key.

encrypt:
Encrypts the input data using AES in EAX mode and returns the ciphertext, nonce, and tag.

decrypt: Decrypts the cipher text using the provided nonce and tag and returns the original plaintext.

generate_key Function: Generates a key using PBKDF2-HMAC with SHA-256 as the hash function.

Protecting the Password

In this protocol we have used a password as an example which is a confidential data related to an IoT device and we use this password to apply AES encrption key to protect the original value of this data which ensures if the stakeholders of this information shares this password with an authorized person, there is less chance of data leakage.

Protecting the data generated

```
7] # Simulating data from IoT device
   original_data = "Temperature: 25C, Humidity: 60%"

   # Modify the original data
   modified_data = original_data.replace('60%', '70%')  # Changing the humidity value
```

Encrypting and Decrypting the data

```python
# Encrypt modified data
ciphertext, nonce, tag = protocol.encrypt(modified_data)

# Decrypt data (using the original ciphertext)
plaintext = protocol.decrypt(ciphertext, nonce, tag)

print("Original data:", original_data)
print("Modified data:", modified_data)
print("Decrypted data (using modified data):", plaintext)
```

Results

```
Original data: Temperature: 25C, Humidity: 60%
Modified data: Temperature: 25C, Humidity: 70%
Decrypted data (using modified data): Temperature: 25C, Humidity: 70%
```

Thank you very much!