

Secure Channel Communication between IOT Devices and Computers

Harsha Kata, Harshith Singathala, Swarag Narayansetty, Sairam Reddy, Jyotsna Malla, Rishita Konda

¹Harsha Kata, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

²Harshith Singathala, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

³Swarag Narayansetty, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

⁴Sairam Reddy, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

⁵Jyotsna Malla, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

⁶Rishita Konda, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

Abstract - The proliferation of Internet of Things (IoT) devices and applications is on the rise, resulting in an increase in both the quantity and complexity of malicious attacks. It is imperative to establish robust security measures for IoT networks to counteract malicious attacks, particularly with the aim of preventing unauthorized control over these devices. While numerous security solutions for IoT have been proposed in recent years, a significant portion of them lacks standardization and interoperability. As the IoT landscape continues to expand, the diversity and intricacy of IoT applications also grow, rendering such networks susceptible to attacks aimed at data theft, device takeover, and service disruption.

A multitude of protocols and networking frameworks have been developed for IoT, with some achieving standardization and facilitating interoperability among devices and internet connectivity. These protocols have received endorsement from prominent standardization bodies such as IETF, IEEE, and industry consortia like the Rawan coalition and thread group.

This paper's objective is to present a model that establishes secure communication channels among IoT devices as well as between these devices and a server or router, accomplished through the implementation of encryption algorithms. The popularity of smart home IoT systems is increasing due to their efficiency in simplifying various tasks. However, this trend also introduces vulnerabilities in terms of user privacy. Safeguarding the privacy of personal data remains a paramount concern for electronic services. To address this challenge, we employ well-established encryption algorithms such as RSA to create and utilize secure communication channels through Socket Programming for IoT devices.

Key Words: —IoT devices, encryption algorithms, RSA, socket programming, communication channels, SHA256, cyber attacks, cybersecurity, communication protocols, cryptography

1. INTRODUCTION

Lately, there has been a significant surge of interest in the concept of the Internet of Things (IoT), capturing the attention of both the industrial sector and academic circles alike. Within the realm of IoT, an immense array of objects equipped with sensors are tasked with amassing data, subsequently transmitting this information to servers for the purpose of analysis, management, and utilization. The ultimate goal is to construct intelligent systems such as smart grids, sophisticated transportation networks, healthcare systems, and even entire smart cities [1], [2], [3].

Due to the prevailing tendency of relegating IoT security to a secondary concern, malicious actors commonly perceive smart devices as easily accessible targets, akin to "low-hanging fruit," susceptible to compromise and manipulation. The significance of prioritizing security, along with the inherent need for privacy, cannot be overstated in the context of IoT. The most reliable and effective approach for ensuring the protection of communication in smart devices is through encryption. Regrettably, striking a balance between convenience and security remains a formidable challenge, particularly when dealing with resource-constrained devices. As a result, numerous IoT products incorporate encryption features that are either ineffective, providing a false sense of security for communications and stored data, or they lack such features entirely [4].

Reported in a 2020 publication by Unit 42, a threat intelligence team, an astounding 98% of the 1.2 million IoT devices scrutinized across corporate networks exhibited a

significant deficiency: the absence of encryption capability for their network traffic. This critical shortcoming led to a vulnerability where 57% of these IoT devices became susceptible to potential threats, including unauthorized interception and manipulation of their data traffic. The same comprehensive report also shed light on the hazardous consequences of amalgamating IoT and IT assets within a Virtual Local Area Network (VLAN). This practice, as elucidated in the report, could result in compromised IoT devices owned by employees inadvertently disseminating malware throughout the entirety of the corporate network [5], [6].

In the proposed model, we have incorporated MongoDB for authentication purposes. Additionally, within this project, the utilization of the SHA256 algorithm for password hashing has been employed. This approach ensures that even individuals overseeing the database cannot ascertain the actual password, as it is stored in a hashed form. The security of SHA-256 rests upon three key properties:

1. **Resistance to Reverse Engineering:** Foremost, it is exceedingly arduous to reverse engineer the original data from its corresponding hash value. A brute force attack, attempting to deduce the original data, would necessitate an astronomical 2^{256} attempts. This level of computational infeasibility provides robust protection.
2. **Collision Resistance:** The likelihood of encountering two distinct messages producing the same hash value (referred to as a collision) is incredibly remote. With a vast 2^{256} range of potential hash values, surpassing the count of particles in the observable universe, the probability of encountering two identical hash values is infinitesimal.
3. **Avalanche Effect:** Lastly, even a minor alteration to the initial data precipitates a significant change in the hash value, rendering it virtually impossible to discern that the new hash value is derived from similar data. This phenomenon is termed the "avalanche effect," further enhancing the security of the hash.

These properties collectively contribute to the robustness and security of the SHA-256 algorithm, making it a prudent choice for protecting sensitive data within our project.

The remaining part of the paper is organized as follows. Section 2 contains the Literature Survey. The Proposed Model is discussed in Section 3. Section 4 contains the Experiments and Results. Lastly, the Conclusion and Future Directions is presented in Section 5.

2. LITERATURE REVIEW

The Internet of Things (IoT) has the potential to revolutionize numerous aspects of our daily routines and behaviors. With the pervasive nature of data sources, a substantial amount of information encompassing nearly every facet of human activity, whether public or private, will be generated, transmitted, collected, stored, and processed. As a result, ensuring the integrity and confidentiality of transmitted data, along with verifying the authenticity of the services providing that data, becomes paramount. This underscores the critical importance of security within the IoT framework.

Data networks, particularly wireless ones, are susceptible to a wide array of attacks, including eavesdropping, impersonation, denial of service, and more. Traditional security measures employed in legacy Internet systems mitigate these threats by relying on encryption and authentication at various layers, including the interface, network, transport, and application layers. While some of these security solutions can be adapted for use in the IoT domain, the inherent limitations in processing power and communication capabilities of IoT devices hinder the implementation of full-fledged security suites. [8].

Secure end-to-end communication channels are achievable within the unconstrained network (UCN) domain using advanced technologies like IPsec, SSL/TLS, or DTLS. However, these technologies cannot be directly applied to constrained network (CN) nodes due to their limitations in memory space and processing power. To address these challenges, this paper proposes a practical security architecture for the Internet of Things (IoT) with the following objectives:

1. **Continued Use of UCN Security Suites:** Existing security suites employed within UCNs will remain unaltered, ensuring compatibility with the UCN side.

2. **Distinct Handling of Initial Security Handshakes:** Initial security handshakes and protocols are managed differently within the CN to accommodate the constraints of limited nodes. This enables the establishment of secure end-to-end channels, allowing constrained devices to manage their complexity effectively.

3. **Unconstrained Nodes Remain Unaffected:** Unconstrained nodes continue to operate without any deviation from their standard procedures.

The proposed solution revolves around offloading computationally intensive tasks to a trusted unconstrained node. This node assumes responsibility for calculating the master session key on behalf of the limited IoT devices under its jurisdiction. IoT Gateways, positioned at the interface between UCN and CN, play a crucial role in mediating communication between these two domains. These gateways

often manage protocol layer variations, spanning physical and interface layers, and even extending to the application layer [13].

In spite of end to end security, lower layers may keep utilizing heterogeneous security highlights across network subspaces or for highlight point communication [14], [18].

Recent advancements in semiconductor technology have paved the way for cost-effective solutions to seamlessly integrate wireless connectivity into embedded processors and sensors. This, in turn, has sparked significant interest in the Internet of Things (IoT), which involves the interconnectedness of everyday objects. The IoT has emerged as a promising technology for the consumer electronics market, with the realm of smart homes being particularly highlighted. Smart homes hold immense potential for IoT deployment, facilitating home automation and efficient energy management. However, the rate at which IoT gains traction among homeowners hinges on their willingness to adopt these devices, and two primary factors influencing their decision are convenience and security. To address this, a Wi-Fi-based IoT smart home system has been conceptualized and realized, utilizing a gateway to establish secure communication among IoT devices. This gateway also enables users to configure, access, and control the system through an intuitive interface accessible from ubiquitous mobile devices, such as smartphones [7], [9]. Inter-device communication is facilitated using the User Datagram Protocol. Prior to transmission, data messages undergo encryption using symmetric cryptography techniques like the Advanced Encryption Standard. The encryption employs a shared key generated through the Elliptic Curve Diffie-Hellman (ECDH) process [16].

The concept of the Internet of Things (IoT) revolves around creating a network of extensively interconnected devices, often referred to as "things." In the current context, the IoT encompasses a wide range of device types, including sensors, actuators, and RFID tags. These devices vary significantly in terms of size, capabilities, and functionalities. The central challenge lies in orchestrating this diverse network to operate seamlessly within the framework of the conventional Internet.

Motivated by this challenge, ongoing research endeavors are focused on adapting, applying, and transforming standard Internet protocols for use within the IoT. The efforts of the 6LoWPAN working group have enabled even the smallest, resource-constrained devices to become integral parts of the Internet by enabling IP communication over these devices. This exceptional feature empowers the connection of billions of devices to the Internet. For instance, a humidity sensor or an RFID tag can communicate not only with each other but also with a human using a smartphone or a remote backend server. While the concept of the IoT may seem straightforward, significant research efforts are still required. Various aspects of the IoT are under investigation, including

its applications and architectures. Furthermore, an increasing number of research initiatives are being undertaken to address challenges related to security, privacy, and trust as IoT devices become more widespread [15], [18].

This study delved into a range of secure, lightweight, and resilient solutions for Wireless Sensor Networks (WSNs) and the Internet of Things (IoT), addressing the distinct security requirements and challenges inherent in these domains. We introduced a novel classification of existing protocols based on their key bootstrapping strategies, with the aim of establishing secure communication. Through a comprehensive analysis, we evaluated these protocols against various criteria to identify their respective advantages and limitations. Our findings suggest a departure from the conventional reliance on symmetric approaches for IoT security. Public key cryptography, particularly asymmetric methods, is emerging as a more viable option within the IoT landscape, provided that these methods are suitably optimized. Additionally, a trusted third party is poised to assume a more active role in enhancing IoT security, considering the heterogeneous nature of IoT deployments.

Furthermore, it is imperative for security protocols to account for the resource-constrained nature of IoT devices. Cumbersome cryptographic operations like RSA and Diffie-Hellman key exchange protocols must be substituted with lightweight alternatives. For instance, the adoption of symmetric cryptography or more streamlined asymmetric algorithms such as ECC (Elliptic Curve Cryptography) and NTRU can significantly improve efficiency. Lightweight security protocols are also crucial in reducing the complexity of communication processes [19], [20].

The realm of smart-home Internet of Things (IoT) solutions is still in its infancy, and the factor of security looms large, significantly influencing the rate of their adoption. Designing a software security solution for smart-home environments presents a challenge due to the lack of user control over various elements. Additionally, integrating a privacy solution into existing software frameworks while considering user experience is another intricate task.

In this paper, we introduced an authorization scheme for smart-home IoT devices that are connected to an untrusted cloud framework. Furthermore, we presented a proof-of-concept software solution that leverages the FIDO protocol, enabling users to authenticate with their devices. We also proposed several extensions and adaptations of the FIDO protocol suitable for an IoT environment. The protocol we introduced prioritizes user anonymity by ensuring that manufacturers cannot establish a link between different user accounts. The most relevant user-related information consists of the FIDO public key and a pseudonym. Experimental results indicated that the additional delay introduced by the FIDO authentication protocol has a minimal impact on smart-home applications. Looking ahead,

our future efforts will focus on various directions. These include testing the FIDO protocol with hardware authenticators, enhancing the security scheme with FIDO U2F features (second-factor authentication), adapting the software security stack for different hardware platforms and programming languages, and proposing an IoT device authentication scheme for smart homes using the privacy concepts outlined in this paper[10],[11],[12].The user-to-device connections, which form the foundation of our scenario, consider the user-device-driven nature of the security framework we propose. The authorization messages presented in this paper are secured through the FIDO protocol, and unlocking the FIDO private key requires biometric (fingerprint) authentication on the user device. The user-to-device authentication relies on the Android security framework and hardware security modules found in smartphones, such as fingerprint readers or ARM TrustZone [17].

3. PROPOSED MODEL

The vulnerabilities present in IoT systems, combined with the potential interception and manipulation of communications by hackers, pose a significant threat. This can lead to inaccurate data reaching IoT devices, resulting in malfunctions and erroneous outcomes. Companies relying on IoT-derived data are exposed to risks, as manipulated data traffic can undermine the accuracy and reliability of their operations. To mitigate this, we have proposed the implementation of encrypted communication messages between IoT devices and clients, effectively thwarting the ability of hackers to tamper with the traffic.

Our encryption scheme offers several distinct advantages:

a. **Comprehensive Security:** Our scheme achieves multiple security objectives in a single logical step, ensuring confidentiality, integrity, authentication, non-repudiation, and anonymity simultaneously.

b. **Heterogeneous Compatibility:** The scheme accommodates the heterogeneity of IoT environments, allowing a sensor node utilizing identity-based cryptography to transmit messages to a server employing a public key infrastructure. This versatile compatibility renders our scheme well-suited for secure data transmission within IoT ecosystems.

By incorporating encryption into the communication process between IoT devices and clients, we provide a robust defense against unauthorized tampering and manipulation, enhancing the overall security and trustworthiness of IoT systems.

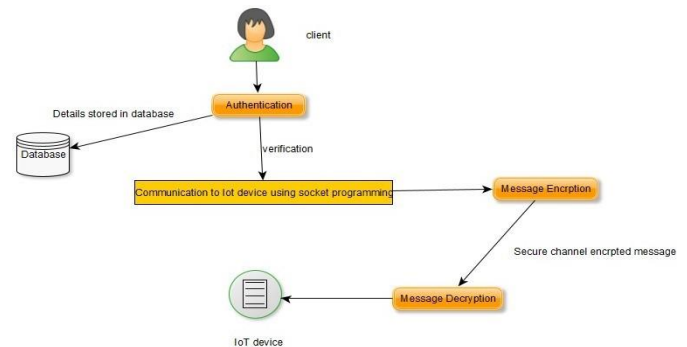


Fig -1: Architecture Diagram of the proposed model

A. SOCKET PROGRAMMING

Communication between the IoT device and the client is established through socket programming, enabling a two-way exchange of messages between these two entities. Sockets serve as the endpoints of a bidirectional communication channel, allowing data transmission and reception. This communication can occur within a single process, between processes on the same machine, or even across processes on geographically distant locations [22], [23].

The creation of a socket involves the following step: `s = socket.socket(socket_family, socket_type, protocol=0)`.

For server-side operations, the following socket methods are utilized:

1. `s.bind()`: This method associates the socket with a specific address and port to listen for incoming client connections.
2. `s.listen()`: It prepares the socket for accepting incoming client connections.
3. `s.accept()`: This method accepts an incoming client connection and returns a new socket object representing the connection, along with the address of the client.

On the client side, the primary method is:

1. `s.connect()`: This method establishes a connection to the server, facilitating message exchange and communication between the client and the server.

In summary, socket programming enables seamless communication between IoT devices and clients through a well-defined set of methods and operations, allowing for the exchange of messages in a bidirectional manner.

B. RSA ALGORITHM

To safeguard against potential attacks aimed at intercepting and pilfering messages and data, encryption becomes imperative. Encrypting messages ensures that unauthorized access and decryption by hackers become formidable challenges. In our project, we employ the RSA algorithm for data encryption—a robust asymmetric key cryptographic algorithm rooted in public key encryption [21]. This algorithm is renowned for its formidable security and near-impossibility to break. In addition to encryption, our proposed model encompasses an authentication feature to validate users whose information is stored in a database. For this purpose, we utilize MongoDB, a user-friendly and highly scalable database system. MongoDB was chosen for its ease of use and scalability. To enhance security, passwords entered by clients are hashed before being stored in the database, further fortifying the protective measures.

The design of our proposed model takes into account resilience against various attacks:

1. Eavesdropping Attack: By encrypting the data, we effectively thwart eavesdropping attempts, making it extremely difficult for attackers to comprehend the intercepted information.

2. Man-in-the-Middle Attack: The inclusion of encryption and authentication mechanisms helps mitigate the risk of man-in-the-middle attacks, as unauthorized intermediaries would be unable to decipher the encrypted communication or gain access through authentication.

3. ARP Spoofing Attack: Our model incorporates measures to counter ARP spoofing attacks, which involve manipulation of network address resolution. These precautions help maintain the integrity and authenticity of communication channels.

In essence, our proposed model's combination of encryption, authentication, password hashing, and measures against specific attacks serves to enhance the overall security and integrity of the IoT communication framework.

4. EXPERIMENTS AND RESULTS

In our experimental setup, we establish the MongoDB service to serve as the foundation for authentication within the proposed model. The primary objective is to ensure authorized access for making requests from clients to IoT devices. To achieve this, we follow these steps:

1. Adding User to MongoDB: Users are added to the MongoDB database to enable authorized

interactions. These users are granted the privilege of sending authorized commands to IoT devices from clients.

2. Data Storage in MongoDB: Relevant user data, including login credentials, is stored securely within the MongoDB database. Notably, passwords are hashed to bolster security measures, safeguarding sensitive information.

3. Connection and Login: When a connection attempt is initiated, the client is prompted to provide login credentials. These credentials serve as authentication for access to IoT devices.

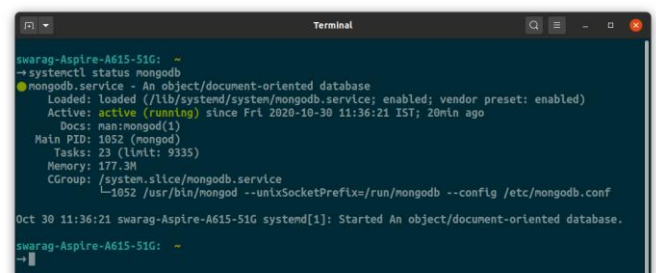
4. Valid Credentials: Upon entering correct login credentials, the user gains access to the IoT device and is authorized to issue commands. This ensures that only authenticated users can interact with the IoT device.

5. Invalid Credentials: In cases of incorrect or invalid credentials, access to the IoT device is denied, preventing unauthorized users from entering the system.

6. Encrypted Command Transmission: Any command issued by an authorized user is encrypted before transmission. This encrypted command is then securely sent to the IoT device.

7. Decryption and Execution: Upon reaching the IoT device, the encrypted command is decrypted and processed. The IoT device accurately interprets and executes the decrypted command as intended by the authenticated user.

This experimental setup showcases the seamless integration of MongoDB for user authentication, secure storage, and hashed passwords. The interaction between clients and IoT devices is streamlined through encrypted command transmission, ensuring that only authorized users can access and control IoT devices. The process reinforces the security, privacy, and overall integrity of the proposed model.



```

swarag-Aspire-A615-51G: ~
└─$ systemctl status mongod
● mongod.service - An object/document-oriented database
   Loaded: loaded (/lib/systemd/system/mongod.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2020-10-30 11:36:21 IST; 20min ago
     Docs: man:mongod(1)
    Main PID: 1052 (mongod)
      Tasks: 23 (limit: 9335)
     Memory: 177.3M
    CGroup: /system.slice/mongod.service
            └─1052 /usr/bin/mongod --unixSocketPrefix=/run/mongod --config /etc/mongod.conf

Oct 30 11:36:21 swarag-Aspire-A615-51G systemd[1]: Started An object/document-oriented database.
swarag-Aspire-A615-51G: ~

```

Fig -2: Starting MongoDB

```

hash.py
1  import sys, time
2  from socket import *
3  from Crypto.PublicKey import RSA
4  from Crypto import Random
5  from Crypto.Cipher import PKCS1_OAEP
6  from base64 import b64decode
7
8  def generateRSAKeys(pubName, privName):
9      rng = Random.new().read
10     key = RSA.generate(4096, rng)
11
12     binPrivKey = key.exportKey('PEM')
13     binPubKey = key.publickey().exportKey('PEM')
14
15     with open(pubName, "w") as f:
16         f.write(binPrivKey)
17         f.close()
18
19     with open(privName, "w") as f:
20         f.write(binPubKey)
21         f.close()
22
23     generateRSAKeys("IOTrsa", "IOTrsa.pub")
24     generateRSAKeys("CLIENTrsa", "CLIENTrsa.pub")

```

Fig -3: Development Environment

```

generateKeys.py > ...
1  import sys, time
2  from socket import *
3  from Crypto.PublicKey import RSA
4  from Crypto import Random
5  from Crypto.Cipher import PKCS1_OAEP
6  from base64 import b64decode
7
8  def generateRSAKeys(pubName, privName):
9      rng = Random.new().read
10     key = RSA.generate(4096, rng)
11
12     binPrivKey = key.exportKey('PEM')
13     binPubKey = key.publickey().exportKey('PEM')
14
15     with open(pubName, "w") as f:
16         f.write(binPrivKey)
17         f.close()
18
19     with open(privName, "w") as f:
20         f.write(binPubKey)
21         f.close()
22
23     generateRSAKeys("IOTrsa", "IOTrsa.pub")
24     generateRSAKeys("CLIENTrsa", "CLIENTrsa.pub")

```

Fig -4: generateKeys.py

```

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
- ls
client.py generateKeys.py hash.py IOT.py messaging_util.py passwords testExport.py
swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
+ python generateKeys.py
swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
- ls
client.py CLIENTrsa.pub hash.py IOTrsa messaging_util.py testExport.py
CLIENTrsa generateKeys.py IOT.py IOTrsa.pub passwords
swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
+

```

Fig -5: Generating RSA Public and Private Keys

```

hash.py > ...
1  import os
2  import hashlib
3  from pymongo import MongoClient
4
5  def add_user(username, password):
6      client = MongoClient()
7      db = client.IOT
8
9      #check to see if that username is already used
10     if db.login_info.find_one({"username": username}) is None:
11         #hash password
12         m = hashlib.sha256()
13         m.update(password)
14         db.login_info.insert_one({"username": username, "password": m.hexdigest()})
15     else:
16         print "Error: Username \""+username+"\" is already in use."
17
18     username = raw_input("username : ")
19     password = raw_input("password : ")
20     add_user(username, password)

```

Fig -6: hash.py

```

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→ python hash.py
username : IOT_JCOMP
password : qwerty

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→

```

Fig -7: Adding an user

```

swarag-Aspire-A615-51G: ~
+ mongo
MongoDB shell version v3.6.8
connecting to mongo://127.0.0.1:27017
Implicit session: session ["id": UUID("5bc2a976-3d64-4135-9076-fdcf823270d0")]
MongoDB server version: 3.6.8
Server has startup warnings:
2020-10-30T11:36:01.364+0530 I STORAGE [InitandListen]
2020-10-30T11:36:40.248+0530 I STORAGE [InitandListen] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2020-10-30T11:36:45.045+0530 I CONTROL [InitandListen] ** See http://docs.mongodb.org/manual/tutorial/enable-authentication
2020-10-30T11:36:45.045+0530 I CONTROL [InitandListen] ** WARNING: Access control is not enabled for the database.
2020-10-30T11:36:45.045+0530 I CONTROL [InitandListen] ** Read and write access to data and configuration is unrestricted.
2020-10-30T11:36:45.045+0530 I CONTROL [InitandListen]
+ use IOT
switched to db IOT
+ db.insert(login_info, {"username": "IOT_JCOMP",
+ "id": ObjectId("5f9bb3fabfd195e084de84bd"), "username": "IOT_JCOMP", "password": "65e84be33532fb784c48129675f9eff3a682b27168cbea74402cf58ee02337c5", "id": ObjectId("5f9bb3fabfd195e084de84bd")})

```

Fig -8: Data added to MongoDB

```

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→ python client.py
Data received from: ('192.168.0.104', 52632)
Do you want to connect to swarag-Aspire-A615-51G? (Y/N) Y
username :

```

Fig -9: Login details given by the user

```

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→ python IOT.py
CONNECT: a5a3c2ba7033442987405d55954ae602, 1, swarag-Aspire-A615-51G

```

Fig -10: Client Server and IOT device run together

```

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→ python client.py
Data received from: ('192.168.0.104', 36354)
Do you want to connect to swarag-Aspire-A615-51G? (Y/N) Y
username : IOT_JCOMP
password :
Data received from: ('192.168.0.104', 58001)
Congrats, we logged on.
What would you like to send?, enter 'exit' to end
+
swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→ python IOT.py
CONNECT: edf915c8ec9d4399b2ce92f4b916962b, 1, swarag-Aspire-A615-51G
ENTRY >>> { 'username': 'u'IOT_JCOMP', 'password': 'u'65e84be33532fb784c48129675f9eff3a682b27168cbea74402cf58ee02337c5', 'id': ObjectId('5f9bb3fabfd195e084de84bd') }
Decrypted Payload:
You sent IOT: InternetOfThings,50255
Encrypted Message:
UPxCKLkVwMwtW539htzm15/SK1S+RhsA2mUQ3JRFssUmZKUu8s9BeedlRt6DAWTJFmfmfgYmWj
s89GHBEYIHBj7aA92mNgn30qvJ0SkkwBmo2C4oyF4TufdFAMlDsTpW8Cdzma96wP2E2V2Q0Pv6f05
4Jm3nffqhhZDznfyEUsTwhdBydHEXFFoVtqG9iDpF+kHheLYeI107NDIVS/U3H0FK6/LM37xv
PfrdLecZAM/W4Cercz2yf5EXIAjDjckG63k1jseYU9d0V8FS+PAs98CCKPPrJ0B80yCfMmK1Se
kXAEbXhh+Estf0pKc48Bq10IDCCH40/mIwEKYRqkKAG0G6Xh1anHmberLKNIXzEQ+04RU0d3
5krq4Mz0GnH020u1yPqGyPskop6D45SvYf608C6wT8qgl9TSJ0H5m08Nd8Bf8m0VekHkQv+
T1x57a71PzW//aeXqvk8shSwarfo7q+kEhsVM03X2z4/y1zXrRX8brTqC842UQ20dX3EVhue
ZJIMLT7vpsFvsBd0MysKchalxxUInyehAIhZsy/Ig7abPbkiGrYnp1dJkFG263t8Fogwaz5F1/
mFOVgtAWQ5wsI9XC+oQ3fdhnc2kc6YEn1xbo1qfXTkeZv8y1RBoqA6FUAVsA8t0vR2p0Y34c=

```

Fig -11: User is logged into IOT device

```

swarag-Aspire-A615-51G: ~/Documents/iot-j/Security-of-Things
→ python IOT.py
CONNECT: 15b8ac9ad1284f889324accb8d6f8be, 1, swarag-Aspire-A615-51G
ENTRY >>> { 'username': 'u'IOT_JCOMP', 'password': 'u'65e84be33532fb784c48129675f9eff3a682b27168cbea74402cf58ee02337c5', 'id': ObjectId('5f9bb3fabfd195e084de84bd') }
Decrypted Payload:
You sent IOT: InternetOfThings,50255
Encrypted Message:
UPxCKLkVwMwtW539htzm15/SK1S+RhsA2mUQ3JRFssUmZKUu8s9BeedlRt6DAWTJFmfmfgYmWj
s89GHBEYIHBj7aA92mNgn30qvJ0SkkwBmo2C4oyF4TufdFAMlDsTpW8Cdzma96wP2E2V2Q0Pv6f05
4Jm3nffqhhZDznfyEUsTwhdBydHEXFFoVtqG9iDpF+kHheLYeI107NDIVS/U3H0FK6/LM37xv
PfrdLecZAM/W4Cercz2yf5EXIAjDjckG63k1jseYU9d0V8FS+PAs98CCKPPrJ0B80yCfMmK1Se
kXAEbXhh+Estf0pKc48Bq10IDCCH40/mIwEKYRqkKAG0G6Xh1anHmberLKNIXzEQ+04RU0d3
5krq4Mz0GnH020u1yPqGyPskop6D45SvYf608C6wT8qgl9TSJ0H5m08Nd8Bf8m0VekHkQv+
T1x57a71PzW//aeXqvk8shSwarfo7q+kEhsVM03X2z4/y1zXrRX8brTqC842UQ20dX3EVhue
ZJIMLT7vpsFvsBd0MysKchalxxUInyehAIhZsy/Ig7abPbkiGrYnp1dJkFG263t8Fogwaz5F1/
mFOVgtAWQ5wsI9XC+oQ3fdhnc2kc6YEn1xbo1qfXTkeZv8y1RBoqA6FUAVsA8t0vR2p0Y34c=

```

Fig -12: Encrypted command decrypted and read

9. CONCLUSIONS

Indeed, authentication and authorization are pivotal components for ensuring the security and trustworthiness of IoT devices. In our project, we have taken a comprehensive approach to address these aspects by implementing a secure channel communication protocol between IoT devices and clients. To achieve this, we have harnessed the power of several key concepts, including Socket Programming and Cryptography.

The utilization of Socket Programming enables seamless communication between IoT devices and clients, facilitating the exchange of data and commands. On top of this foundation, we have integrated robust cryptographic measures to ensure the confidentiality and integrity of the transmitted information. Particularly, the industry-standard RSA algorithm has been employed to encrypt commands originating from the client devices to the IoT devices.

Our implementation also showcases the significance of using MongoDB as a database for authentication purposes. MongoDB's popularity within the IoT developer community makes it a suitable choice for integrating this secure authentication mechanism into modern IoT applications. The database stores user credentials securely, and through proper authentication and authorization procedures, only authorized users gain access to the IoT devices.

The amalgamation of these elements—Socket Programming, Cryptography, and MongoDB—results in a comprehensive and reliable framework that enhances the security and functionality of IoT devices. By focusing on secure communication, encryption, and proper user authentication, our project contributes to the advancement of IoT security and paves the way for future innovations in this field.

REFERENCES

- [1] Y. Lu and L. Da Xu, "Internet of Things (IoT) Cybersecurity Research: A Review of Current Research Topics," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2103–2115, 2019, doi: 10.1109/JIOT.2018.2869847.
- [2] P. D. Babu, C. Pavani, and C. E. Naidu, "Cyber Security with IOT," in 2019 Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), 2019, vol. 1, pp. 109–113, doi: 10.1109/ICONSTEM.2019.8918782.
- [3] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022, doi: 10.1109/ACCESS.2022.3165809
- [4] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT Security: Ongoing Challenges and Research Opportunities," in 2014 IEEE 7th International Conference on Service-Oriented Computing and Applications, 2014, pp. 230–234, doi: 10.1109/SOCA.2014.58.
- [5] G. Cascavilla, R. Zwart, D. A. Tamburri, and A. Cuzzocrea, "Explaining IoT Attacks: An Effective and Efficient Semi-Supervised Learning Framework," in 2022 IEEE International Conference on Big Data (Big Data), 2022, pp. 5662–5671, doi: 10.1109/BigData55660.2022.10020894.
- [6] M. Roopak, G. Yun Tian, and J. Chambers, "Deep Learning Models for Cyber Security in IoT Networks," in 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 2019, pp. 452–457, doi: 10.1109/CCWC.2019.8666588.
- [7] T. Islam, R. A. Youki, B. R. Chowdhury, and A. S. M. T. Hasan, "An ECC Based Secure Communication Protocol for Resource Constraints IoT Devices in Smart Home," in Proceedings of the International Conference on Big Data, IoT, and Machine Learning, 2022, pp. 431–444.
- [8] H. Kim, A. Wasicek, B. Mehne, and E. A. Lee, "A Secure Network Architecture for the Internet of Things Based on Local Authorization Entities," in 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), 2016, pp. 114–122, doi: 10.1109/FiCloud.2016.24.
- [9] F. K. Santoso and N. C. H. Vun, "Securing IoT for smart home system," in 2015 International Symposium on Consumer Electronics (ISCE), 2015, pp. 1–2, doi: 10.1109/ISCE.2015.7177843
- [10] M. Malik, M. Dutta, and J. Granjal, "A Survey of Key Bootstrapping Protocols Based on Public Key Cryptography in the Internet of Things," *IEEE Access*, vol. 7, pp. 27443–27464, 2019, doi: 10.1109/ACCESS.2019.2900957.
- [11] R. Zahid, M. W. Anwar, F. Azam, A. Amjad, and D. Mukhtar, "Enhancing End-to-End Communication Security in IoT Devices Through Application Layer Protocol," in Information and Software Technologies, 2022, pp. 148–159.
- [12] P. Mall, M. Z. A. Bhuiyan, and R. Amin, "A Lightweight Secure Communication Protocol for IoT Devices Using Physically Unclonable Function," in Security, Privacy, and Anonymity in Computation, Communication, and Storage, 2019, pp. 26–35.
- [13] R. Sharma and R. Arya, "Secure transmission technique for data in IoT edge computing infrastructure," *Complex*

- Intell. Syst., vol. 8, no. 5, pp. 3817–3832, 2022, doi: 10.1007/s40747-021-00576-7.
- [14] K. T. Nguyen, M. Laurent, and N. Oualha, "Survey on secure communication protocols for the Internet of Things," *Ad Hoc Networks*, vol. 32, pp. 17–31, 2015, doi: <https://doi.org/10.1016/j.adhoc.2015.01.006>.
- [15] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, A. Serbanati, and M. Rossi, "Secure communication for smart IoT objects: Protocol stacks, use cases and practical examples," in *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2012, pp. 1–7, doi: 10.1109/WoWMoM.2012.6263790.
- [16] K. K. Rout, S. Mallick, and S. Mishra, "Design and Implementation of an Internet of Things based Prototype for Smart Home Automation System," in *2018 International Conference on Recent Innovations in Electrical, Electronics Communication Engineering (ICRIEECE)*, 2018, pp. 67–72, doi: 10.1109/ICRIEECE44171.2018.9008410.
- [17] Chifor, I. Bica, V.-V. Patriciu, and F. Pop, "A security authorization scheme for smart home Internet of Things devices," *Futur. Gener. Comput. Syst.*, vol. 86, pp. 740–749, 2018, doi: <https://doi.org/10.1016/j.future.2017.05.048>.
- [18] R. Roman, R. Arjona, and I. Baturone, "Post-quantum Secure Communication with IoT Devices Using Kyber and SRAM Behavioral and Physical Unclonable Functions (Extended Abstract)," in *Attacks and Defenses for the Internet-of-Things*, 2022, pp. 72–83.
- [19] R. V. S. Boolakee, S. Armoogum, R. Foogooa, and G. Suddul, "Ensuring Secure Communication from an IoT Edge Device to a Server Through IoT Communication Protocols," in *Progress in Advanced Computing and Intelligent Engineering*, 2021, pp. 337–349.
- [20] S. Encheva and S. Tumin, "Simple E2EE Secure Communication Protocol for Tiny IoT Devices," in *Cooperative Design, Visualization, and Engineering*, 2022, pp. 286–292.
- [21] S. Encheva and S. Tumin, "Simple E2EE Secure Communication Protocol for Tiny IoT Devices," in *Cooperative Design, Visualization, and Engineering*, 2022, pp. 286–292.
- [22] C. Mateo, F. Almagro, W.-J. Yi, and J. Saniie, "Design Flow and Implementation of an IoT Smart Power Socket," in *2021 IEEE International Conference on Electro Information Technology (EIT)*, 2021, pp. 151–156, doi: 10.1109/EIT51626.2021.9491841.
- [23] A. Rehman, A. R. Syed, I. U. Khan, A. A. Mustafa, M. B. Anwer, and U. A. Ali, "IoT-Enabled Smart Socket," *Wirel. Pers. Commun.*, vol. 116, no. 2, pp. 1151–1169, 2021, doi: 10.1007/s11277-020-07043-5.