

# **The University of Texas at Dallas**



## **Network Security Project**

**Fall 2023**

## **Privacy Preserving E-Commerce system**

Chandan Alwala (CXA220003)

## **Contents**

1. Introduction and Problem statement.....	3
2. Security features.....	4
2.1 Authentication.....	4
2.2 Confidentiality.....	4
2.3 Integrity.....	6
3. Application Flow.....	7
4. Results.....	8
4.1 How to run.....	8
4.2 Screenshots.....	8
5. Conclusion and Future Work.....	10
6. Contribution of each team member.....	11

## **1. Introduction and Problem Statement**

This project aims to design and implement a privacy-preserving e-commerce system involving three main parties: a customer, a broker, and a merchant. The merchant sells e-products at the same price. Customers interested in purchasing these e-products connect with the broker, who, in turn, connects to the merchant. This process is designed to conceal the customer's identity from the broker.

However, it is crucial that the broker remains unaware of the specific products customers are purchasing, either directly or indirectly, to safeguard their privacy. Therefore, the customer and merchant need to establish a cryptographic tunnel for communication. The merchant can pad data to ensure the broker cannot deduce the e-product based on the file size, as the broker will mediate the file transfer.

The requirements of the system are integrity, confidentiality and authentication. All of the design must be done with certain constraints. The only mechanism available for authentication is public key cryptosystem. The only security primitive available for confidentiality is keyed hash mechanism and similar mechanism maybe used for integrity.

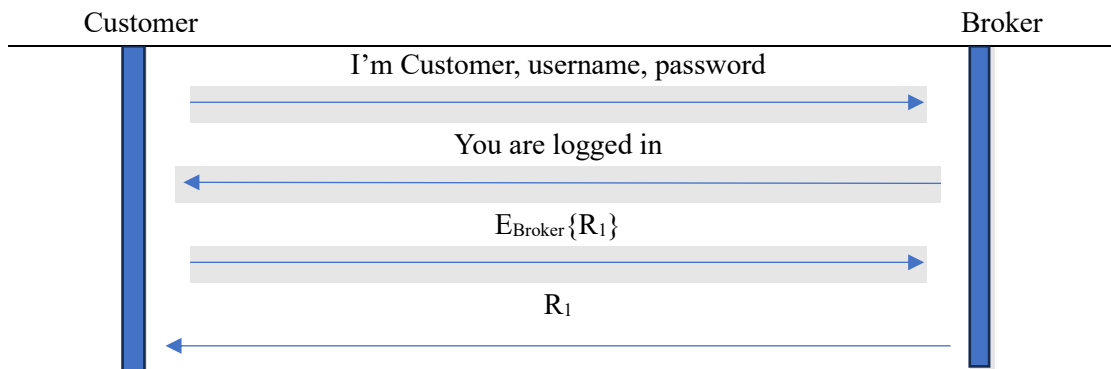
## **2. Security Features**

### **2.1 Authentication**

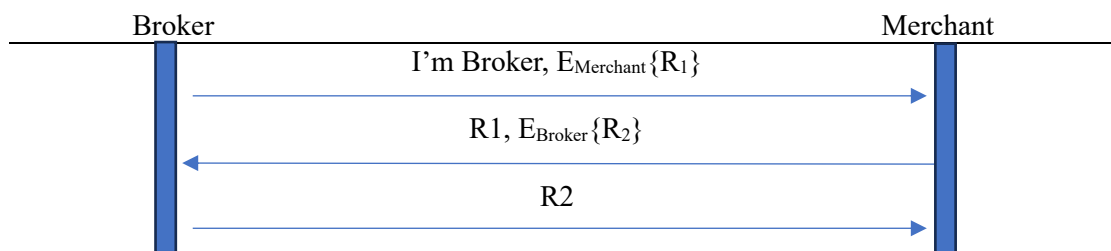
Merchant and Broker have RSA public-private key pairs.

#### **2.1.1 Customer-Broker Mutual Authentication**

Customer Authenticated using username, password and broker authenticated using Random number challenge



#### **2.1.2 Broker-Merchant Mutual Authentication**



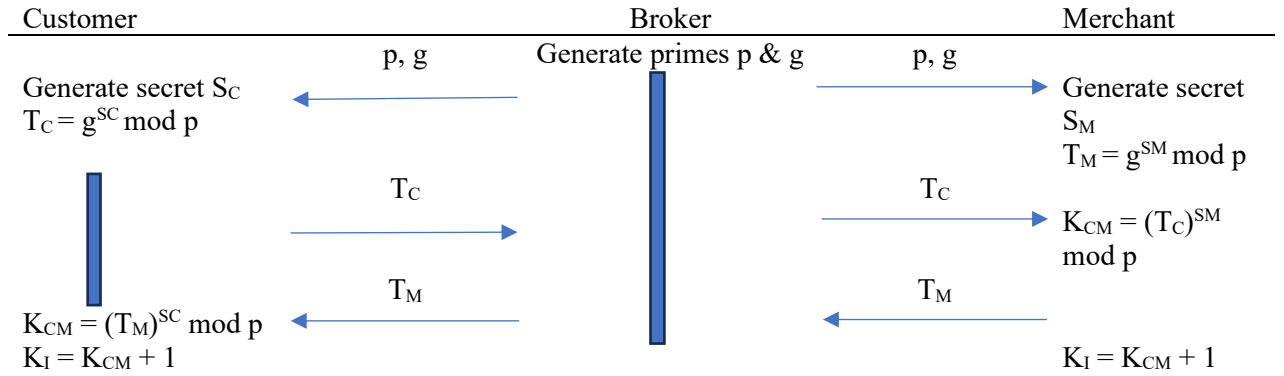
#### **2.1.3 Customer-Merchant Mutual Authentication**

Since Broker authenticated Merchant, and Customer authenticated Broker, Customer and Merchant authentication is not required.

### **2.2 Confidentiality**

#### **2.2.1 Session Key generation**

Establish DH session key between Customer and Merchant. All further communication between Customer and Merchant via Broker is encrypted with this session key.



### 2.2.2 Encryption

Encryption is done using “Mixing in the plain text” concept, outlined as follows

Initial Vector – all 0s

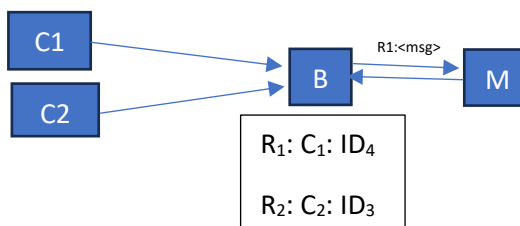
$$\begin{aligned}
 b_1 &= \text{SHA-256}(K_{CM} \parallel \text{IV}) & c_1 &= p_1 \text{ XOR } b_1 \\
 b_2 &= \text{SHA-256}(K_{CM} \parallel c_1) & c_2 &= p_2 \text{ XOR } b_2 \\
 \dots & & \dots & \\
 b_i &= \text{SHA-256}(K_{CM} \parallel c_{i-1}) & c_i &= p_i \text{ XOR } b_i
 \end{aligned}$$

### 2.2.3 Preserving Customer’s privacy from Broker (File Padding)

In order to prevent the Broker from deducing a Customer’s purchase based on the file size it is transferring, all files are padded to 1KB at the Merchant’s side.

### 2.2.4 Preserving Customer’s privacy from Merchant (Request Numbers)

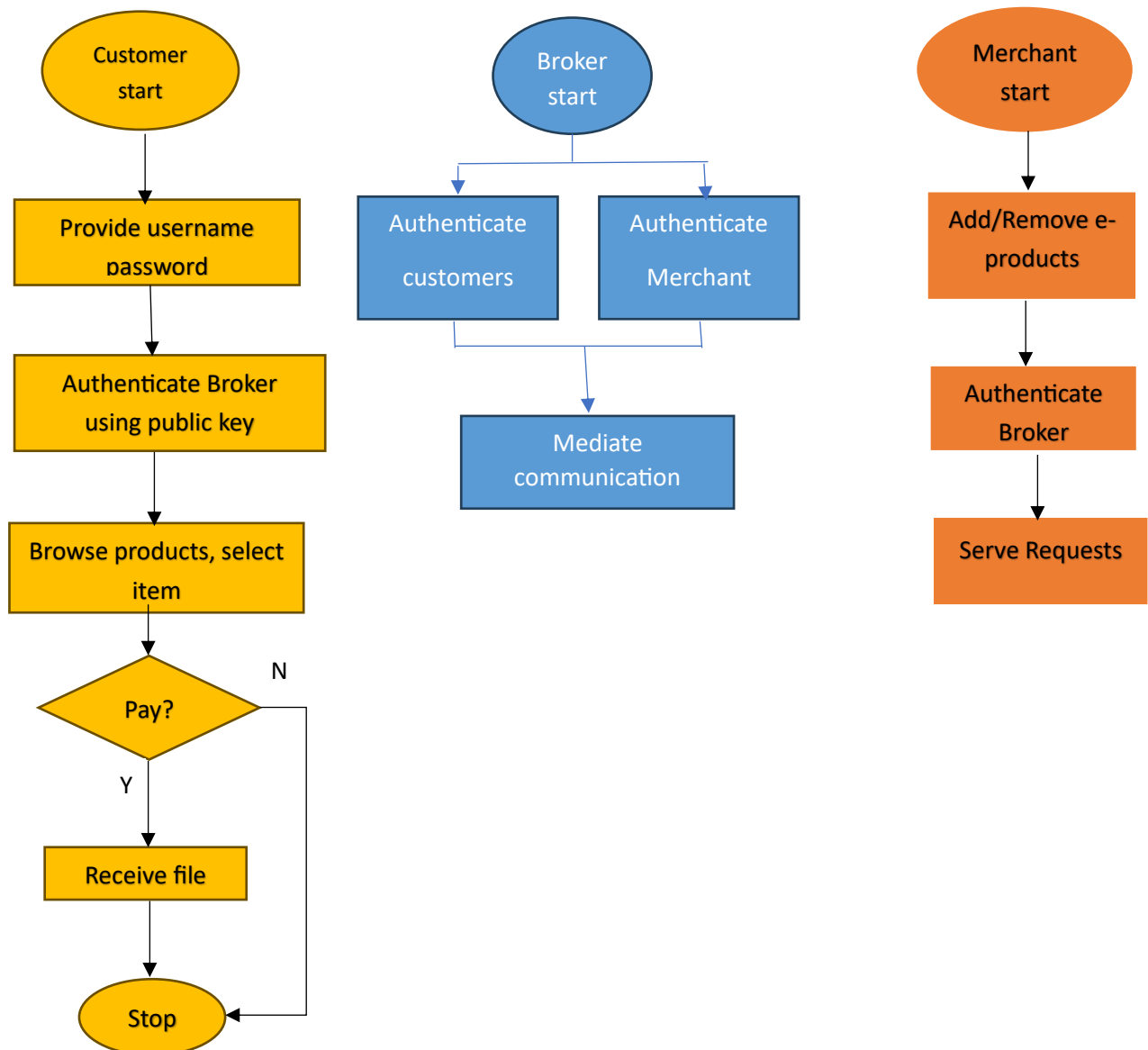
- The merchant shares session keys with multiple customers. However, without revealing the customer's identity, the merchant should be able to encrypt/decrypt a particular customer's request.
- The broker facilitates this by mapping a customer to a request number. The merchant will only be aware of a key associated with a specific request number.
- For every communication from a particular customer, the broker will send the request number along with the message. This process enables the merchant to encrypt/decrypt using the correct key solely based on the request number, without knowing the actual key.



### **2.3 Integrity**

- Integrity key = Diffie-Hellman key + 1
- Integrity is verified by appending the contents with integrity key and then taking a SHA-256 hash (  $\text{hash}(\text{content}||K)$  ).
- This is sent by the Merchant along with encrypted files so that Customer can decrypt the contents and verify if the information is untampered.

### 3. Application Flow



- **E-Product Browsing** - Customer should be able to view the Merchant's product list – Merchant sends the list of products to the Broker. Broker will forward it to Customer.
- **E-Product Checkout** - Customer inputs a product Id, which is confidentiality protected. The broker will forward it to Merchant.
- **E-Product Payment** - Merchant asks broker for payment, Broker asks confirmation from customer. If customer accepts, certain amount will be deducted from the customer and added to merchant and file transfer is initiated.
- **E-Product Delivery** – The file is delivered encrypted with session key. The Customer receives it, decrypts it, removes the padding and integrity checks the contents.
- **E-Product Addition/Removal** – Merchant can add/remove an e-product.

## 4. Results

### 4.1 How to run

1. Clone the repo to a desired location
2. Ensure all installations are present. (If not, use pip install <library>).
3. Navigate to the Merchant folder and run python Merchant.py
4. Navigate to the Broker folder and run python Broker.py.
5. Navigate to the Customer<#> folder and run Customer<#>.py

### 4.2 Screenshots

- Mutual authentication

```
(ve2) C:\Users\sahca\Documents\MS CS\FALL 23\NETWORK SECURITY\NetSecProject-main\GenerateDHKey\Merchant>python Merchant.py
Merchant is listening on port 8100...
Broker authenticated successfully.

(ve2) C:\Users\sahca\Documents\MS CS\FALL 23\NETWORK SECURITY\NetSecProject-main\GenerateDHKey\Broker>python broker.py
Broker is listening on port 8000...
Merchant authenticated successfully.
```

- DH session key between Customer and Merchant

```
(ve2) C:\Users\sahca\Documents\MS CS\FALL 23\NETWORK SECURITY\NetSecProject-main\GenerateDHKey\Merchant>python Merchant.py
Merchant is listening on port 8100...
Broker authenticated successfully.
received primes p:227 and g:191 from broker
computed shared secret is 182

(ve2) C:\Users\sahca\Documents\MS CS\FALL 23\NETWORK SECURITY\NetSecProject-main\GenerateDHKey\Customer1>python Customer1.py
Authentication successful.
received primes p:227 and g:191 from broker
computed shared secret is 182
```

- Successful transaction at Customer

```
(ve2) C:\Users\sahca\Documents\MS CS\FALL 23\NETWORK SECURITY\NetSecProject-main\GenerateDHKey\Customer1>python Customer1.py
Authentication successful.
received primes p:227 and g:191 from broker
computed shared secret is 182
1. Item1
2. Item2
Enter your selection: 1
FROM BROKER: Pay to deliver
Do you want to pay? (y/n) y
Payment successful. Remaining balance 95
Receiving File...
File received from Broker.
```

- Entire Flow



```
alwala@Chandans-MacBook-Air Merchant % python3 Merchant.py
Merchant is listening on port 8100...
Do you want to add or remove files? (Enter: add/remove/no): add
Enter file name: item4
Enter file content: This is item4
New file created
Menu updated:
1. item1
2. item2
3. item3
4. item4
Waiting for broker to connect
```

```
alwala@Chandans-MacBook-Air Merchant % python3 Merchant.py
Merchant is listening on port 8100...
Do you want to add or remove files? (Enter: add/remove/no): remove
Enter the file name to remove: item4
File 'item4' removed from the filesystem
File 'item4' removed
Menu updated:
1. item1
2. item2
3. item3
Waiting for broker to connect
```

```
alwala@Chandans-MacBook-Air Merchant % python3 Merchant.py
Merchant is listening on port 8100...
Do you want to add or remove files? (Enter: add/remove/no): no
Waiting for broker to connect
Broker authenticated successfully.
received primes p:18869 and g:21767 from broker
computed shared secret is 8056
FROM BROKER: Send Menu
Sent Menu...
FROM BROKER: Send Selection
selected item: 1
awaiting payment Pay to deliver
FROM BROKER: R1 Payment Status - True
Payment Successful
Total funds 105
Initiating file transfer...
Delivered - Transaction complete

alwala@Chandans-MacBook-Air Customer1 % python3 Customer1.py
Enter your username: username1
Enter your password: password1
User authentication with Broker successful
Broker authenticated successfully.
received primes p:18869 and g:21767 from broker
computed shared secret is 8056
1. item1
2. item2
3. item3
Enter your selection: 1
FROM BROKER: Pay to deliver
Do you want to pay? (y/n): y
Payment successful. Remaining balance 95
Integrity check successful
File received from Broker.
final ack: done

alwala@Chandans-MacBook-Air Broker % python3 Broker.py
Broker is listening on port 8000...
Merchant authenticated successfully.
Connected by customer id: 1
Customer 1 authenticated successfully!
Sending primes to customer and merchant
Sent primes p:18869 and g:21767 to merchant and customer
FROM CUSTOMER1: Menu
Received Menu, forwarding to customer
response from the merchant is: 1. item1
2. item2
3. item3
response sent to client
FROM CUSTOMER1: selection : g
response from the merchant is: Pay to deliver
response sent to client
FROM CUSTOMER1: pay : y
Payment Authorized
Acknowledgement: received file_data
response from the merchant is: done
response sent to client
Transaction Done
```

- Edge case handling

```
alwala@Chandans-MacBook-Air Customer2 % python3 Customer2.py
Enter your username: username2
Enter your password: wrongpassword
Authentication failed. Closing connection.

Enter your selection: tygy
Invalid selection. Please enter valid item number.
Enter your selection: 1
FROM BROKER: Pay to deliver
Do you want to pay? (y/n): sure
Invalid input. Please enter 'y' or 'n'.
Do you want to pay? (y/n): y
```

## **5. Conclusion and Future Work**

### **5.1 Conclusion**

The proposed and implemented design successfully satisfies all requirements of the problem statement, along with ensuring perfect forward secrecy .

### **5.2 Future Work**

#### **5.2.1 Potential Threats**

- Trudy can get previous eavesdropped message sent by Customer to broker and get it decrypted while authentication – Solution – Introduce structure (odd # by initiator, even by sender).
- Broker is a single point of failure and if Broker is compromised, info about customer and the items they bought is compromised
- Man in the Middle attack during Diffie Hellman key generation.
- These maybe resolved with further research and advanced protocol design

#### **5. 2.2 Other work**

- Freshness guarantees may be introduced
- UI may be improved
- Code may be restructured and well-organized