Group Members:-

Tanushri Singh - [tts150030@utdallas.edu](mailto:tts150030@utdallas.edu)

Ko-Chen Chen - [kxc170002@utdallas.edu](mailto:kxc170002@utdallas.edu)

Andrew Shirley - [ars092220@utdallas.edu](mailto:ars092220@utdallas.edu)

We are going to initially divide up the work by each taking an entity.

Tanushri - Client

Ko-Chen - Merchant

Andrew - Broker

We decided to use Java with socket programming and tentatively choosing the Chilkat library for public/private key, symmetric key and key exchange implementation.

***Visual Representation of How the Communication will flow:-***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Client | <- Auth with 3 msg -> | Broker  Can’t see List  Store Payment  Can’t see Item | <- Auth with 3 msg -> | Merchant  Only know broker |
| <- List of Merchants |  |
| Request Catalog from 1 Merchant -> |  |
|  | Request Catalog -> |
| \*client to the merchant, the client needs to be anon\* | \*client to merchant connection, the client must be anon\* |
| <- Return Catalog | <- Return Catalog |
| Choose and tell amount to broker -> | Send payment and item -> |
| <- Give Item | <- Give Item |

***Explanation of the Diagram in Words:-***

1. The client will initiate the process by informing the Broker that she wants to browse the merchant’s website.

2. The Broker will create a connection with the merchant.

3. The client and the merchant will communicate with each other through the Broker in such a way that the Broker cannot decrypt any information.

4. The merchant will send the product catalog file to The client.

5. The client will choose the product and will tell the Broker to pay $X to the merchant.

6. The Broker will store the payment order from The client to its database and will pay the merchant.

7. The merchant will send the product to The client over the tunnel through the Broker .

***How the communication would flow amongst Broker, Merchant, and Client:-***

**Broker -> Merchant**

Establish SK-BM initially for each merchant using the following message structure

* SK-BM is the Session Key Between Broker and Merchant

**Broker -> Merchant**

Mpub{“Broker”, init}

* Use Merchant’s public key to encrypt the contents of the bracket.

**Merchant -> Broker**

Bpub{nonce}

**Broker-> Merchant**

Mpub{nonce+1, nonce2}

**Merchant -> Broker**

Bpub{nonce2 +1, SK-BM}

\*any communication with a session key will be accompanied by the sender’s name\*

E.g. SK-BM{Data sent from broker} can be read as {“Broker”,SK-BM{Data}} instead

**Client -> Broker->Client**

Authorize exactly how the broker authorized with merchants to get SK-BC

**Client -> Broker**

SK-BC{“Requesting Merchants”)

**Broker -> Client**

SK-BC{“List of Merchants (public keys)”)

*Client will choose a merchant M*

**Client -> Broker**

SK-BC{“Selected merchant”, Mpub{“Request catalog”, SK-CM}}

* SK-CM is the Session Key Between Client and Merchant
* Generated randomly by the client with no handshakes needed

since the client is to remain anonymous

*Broker will now generate a sessionId for Client-->Merchant*

*Broker will maintain [Client, SK-BC, SessionId, selected merchant] as state for client*

**Broker -> Merchant**

SK-BM {SessionId, Mpub{“Request catalog”, SK-CM}}

**Merchant -> Broker**

SK-BM {SessionId, SK-CM {Catalog info}}

**Broker -> Client**

SK-BC {SK-CM{Catalog info}}

*Client Chooses Item*

**Client -> Broker**

SK-BC {Cpriv{Price, timestamp}, SK-CM{item}}

*A nonce/timestamp is sent to both protect against replays and to give the broker a verifiable record that the client authorized this payment*

**Broker -> Merchant**

SK-BM {Payment, SessionId, SK-CM{item}}

*Merchant will either reject or accept the payment based on if enough was payed*

**Merchant -> Broker**

SK-BM {SessionId, SK-CM {Product or Actualprice}}

**Broker -> Client**

SK-BC {SK-CM {Product or Actualprice}}

If the product is received, the client will close the connection and the broker will invalidate the session information for the client. If the product is not received, the client will validate and resend the payment information with the correct price.