

| **Course Title:** | **Network Security** |
| --- | --- |
| **Course Number:** | **COE817** |
| **Semester/Year (e.g.F2016)** | **W2022** |

| **Instructor:** | **Dr. Khalid Abdel Hafeez** |
| --- | --- |

| *Assignment/Lab Number:* | *4* |
| --- | --- |
| *Assignment/Lab Title:* | *Course Project: Secure Purchase Order System* |

| *Submission Date:* | *Saturday April 9th 2022* |
| --- | --- |
| *Due Date:* | *Saturday April 9th 2022* |

| **Student**    **LAST Name** | **Student**    **FIRST Name** | **Student**    **Number** | **Section** | **Signature\*** |
| --- | --- | --- | --- | --- |
| Keays | Riley | 500819720 | 04 | R.K. |
| Farooqi | Taimoor | 500833873 | 05 | T.F |
| Chen | XuZe(Rick) | 500909131 | 01 | R.C. |
| Liang | Aiyang | 500706771 | 01 | AL |

\*By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your

own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a “0” on the work,

an “F” in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic

Conduct, which can be found online at: <http://www.ryerson.ca/senate/current/pol60.pdf>

**Introduction:**

The purpose of this project is to implement and demonstrate the various network security protocols and cryptography techniques by using a real-life scenario as a simulation. The goal is to achieve a fully functional application that is able to establish connections between a server and all entities of the network via a secure email authentication system. After a secure connection is established, users will be able to interact with the server and communicate with other entities in a secure manner.

In this project, we will be building a Secure Purchase Order system that simulates a real-life online purchase by allowing the user to submit purchase requests to a secure server and a transaction taking place using the inputted information. The Purchase Order (PO) is then shared with the supervisor using a mail server which encrypts the PO using RSA public-key cryptography before sending the PO to the supervisor. The supervisor is responsible for verifying the validity of the order and must sign their signature on the PO before passing it off to the purchasing department using the same mail server and encryption algorithm. Once the purchasing department has received the signed PO, it will then cross-check the digest signed by the supervisor with the digest signed by the purchaser which contains the signatures and time-stamps of both purchaser and supervisor.

Cross-checking is important as it will prevent repudiation. A repudiation attack occurs when one of the entities in a secure system denies certain actions and commits unauthorized actions intended to disrupt the security of the system. Throughout the project, all connections between parties are preceded by public-key mutual authentication - which involves parties to satisfy each other mutually without having to exchange session keys. Finally, all messages are encrypted using RSA public-key cryptography which is a public-key encryption algorithm containing encryption/decryption capabilities, digital signatures and key exchange.

**Design and Implementation:**

The Secure Purchase Order System consists of 4 main modules, each of which controls one aspect of the Secure Purchase Order system, and 1 minor module that provides utility. The first main module, contained in the file Project\_Client.py, requires the user to enter a password and then authenticates itself with the mail server using RSA public-key encryption. It also negotiates a session key to be used to encrypt messages between it and the mail server using DES. The Project Client then allows the user to enter the name, unit cost, and quantity values for a Purchase Order which is then hashed, signed and time stamped before being encrypted and sent to the mail server.

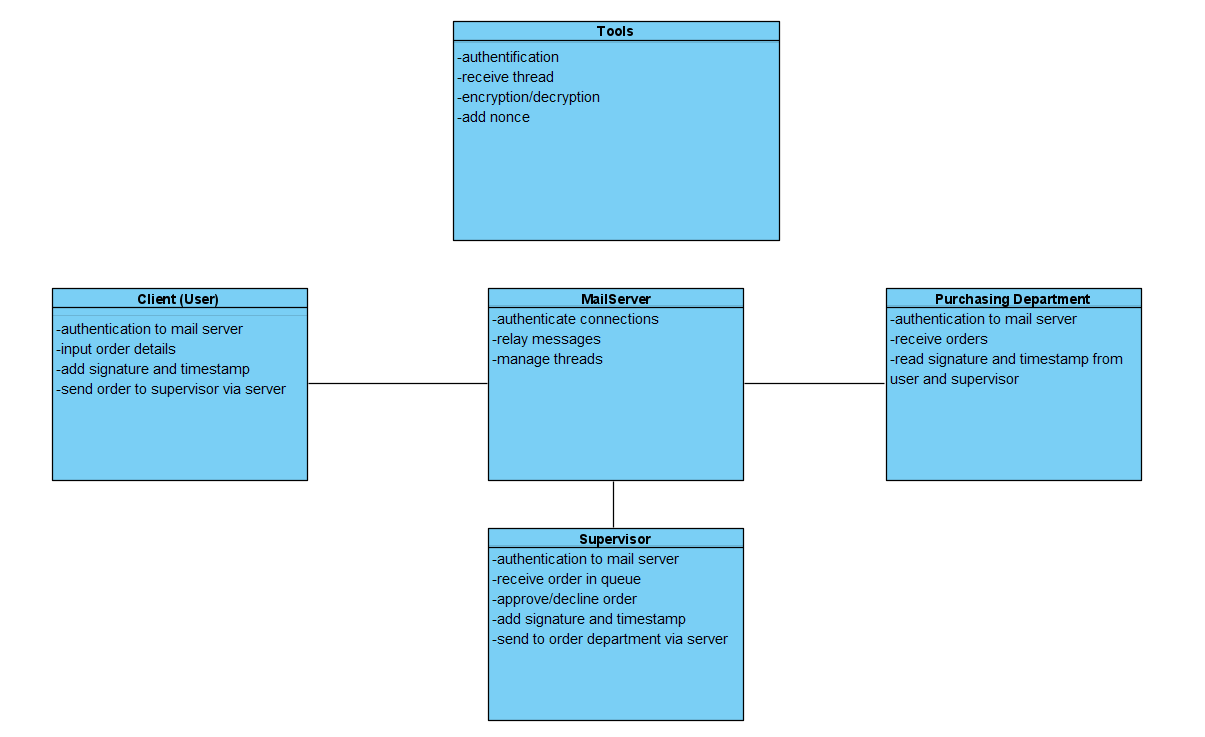
The second main module, contained in the file supervisor.py, follows the same authentication and key distribution procedure as the first to authenticate itself to the mail server and negotiate a session key for communication between the supervisor and the mail server. The Supervisor module then waits until a valid Purchase Order is received. Once a valid order is received the order information is displayed and the supervisor is prompted to decide whether to approve or reject the order. If the order is approved the Supervisor adds the supervisor signature and a fresh timestamp to the order before encrypting it and sending it to the mail server.

The third main module, contained in the file orderDepartment.py, follows an identical authentication and key distribution procedure as the previous two modules to negotiate a session key and authenticate itself to the mail server. The Purchasing Department module then receives Purchase Orders from the mail server, checks their validity using the timestamp and provided signatures, and then displays them for the purchasing department to see.

The fourth main module, contained in the file mail\_server.py, authenticates each of the other modules and exchanges session keys to communicate with each of them.Once a module has been authenticated the server then receives messages from and sends messages to that module as required to facilitate communication between the modules. The mail\_server also checks the validity of each message using the timestamp before refreshing the timestamp and re-encrypting the message with the appropriate session key for retransmission.

Finally the tools.py module provides some useful functions for encrypting and decrypting messages, padding and unpadding messages, generating nonces, authenticating from the initiator perspective, and for receiving messages.

Architecture Diagram:

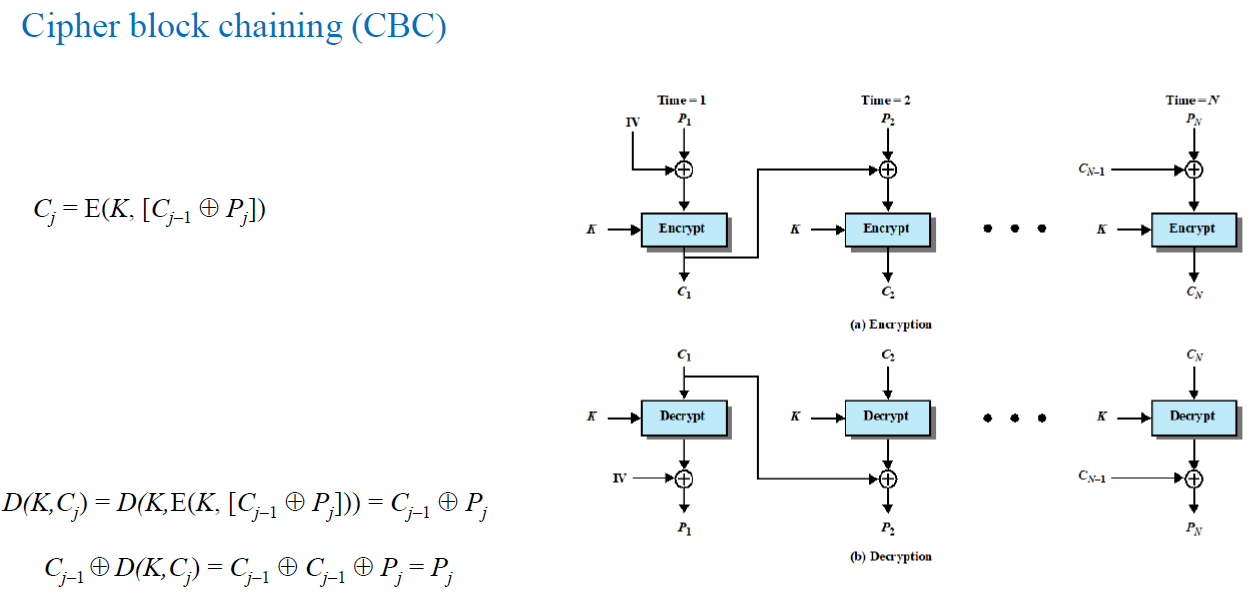
**Figure 1:** Architecture Diagram outlining the structure of our Secure Purchase Order System

Modules Descriptions and Their Functionalities:

* Client: After the client logged in with the password, first calls the authentication method to make sure the connection between the server. After authentication, the client sends a message that contains itemName, unitCost, amount, signature, public key and timestamp to the server.
* Supervisor: The second user of the mail server. Also logs on to the system using a username and password. After receiving the purchase order from the mail system, the supervisor approves or denies the purchase order. If the order is approved, the order will be forwarded to the order department via mail system to be verified.
* Purchasing department: The end point of the purchase order system. Checks the signature from the client and supervisor and the validity of the purchase order
* Mail server: The center of the system, where the communication is managed. Receive messages from 3 users and depending on where the message is from, this message will forward to the associated user.
* Tools: A built-in library for our system. This file contains methods for encryption, decryption, authentication and a method that creates a thread for receiving messages.

Data Encryption Standard (DES):

The main security protocol used by the modules to ensure confidentiality is DES. When using DES data is encrypted in 64-bit blocks using a 56-bit key. The cipher block chaining mode is used for the general-purpose block-oriented transmission. In this mode the input to the encryption algorithm is the XOR of the next 64 bits of the plaintext and the previous 64 bits of ciphertext.

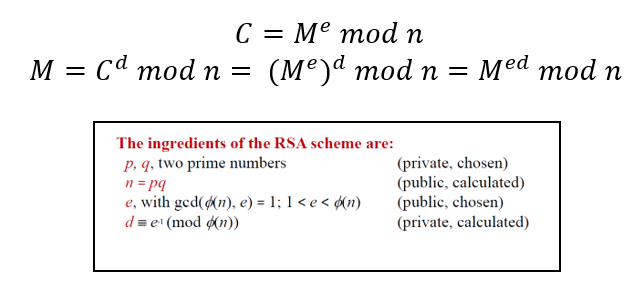


**Figure 2:** A Diagram of the CBC Mode of Operation

Rivest-Shamir-Adleman (RSA)

RSA is the second main security protocol used by the modules to authenticate each other, generate signatures, and distribute the session key needed for the DES encryption. RSA is the most widely used approach to public key encryption. The encryption and decryption are as follows, where *M* is the plaintext, *C* is the ciphertext, n is a known value to both parties:

The public key is PU = {*e, n*} and the private key = {*d, n*}



**Figure 3:** RSA Algorithm formulae

Other Security Techniques and Principles Used:

Throughout the modules several different techniques were used to ensure the confidentiality, integrity, and availability of the Secure Purchase Order system. One such technique involved the use of timestamps appended to the end of each message before encryption. This technique was used to ensure the availability of the system by preventing replay attacks from being used to flood the system. To do this each module appends a new timestamp to the end of the Purchase Order before encryption and then sends the message. When receiving a message each module checks the timestamp, after decrypting the message, by comparing it to the current time and if the timestamp is valid it accepts the message. Otherwise the message is discarded. A valid timestamp is one which is less than 5 seconds old.

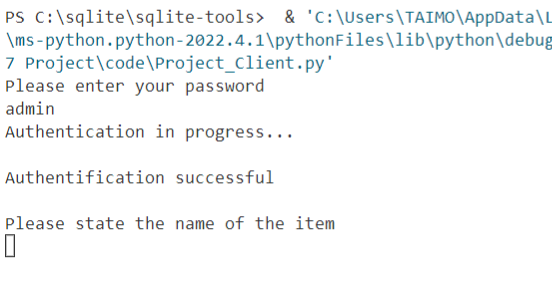
Another technique used, involved using nonces to authenticate the modules to each other. The nonces are a random string of 8 bytes. During authentication the modules exchange nonces with each other and send the received nonces back to the sender. By doing this the modules are able to confirm that the entity with which they are communicating possess the required private key for decryption. Since it is assumed that only the owner of the private key has access to the key the modules can determine that the entity they are communicating with is in fact who they claimed to be during authentication.

The third technique used a hash of the Purchase Order and the private key of either the Project Client or Supervisor module to generate a signature which was encrypted and sent along with the Purchase order. By hashing the Purchase Order we can verify the integrity of it when it is received by the other modules while the signature is used to validate the authenticity of the hash. This ensures the Purchase Order received is not tampered with by any malicious actor. Each time a message is received by a module it is verified to ensure the message is valid. The Project Client signs the Purchase Order with its signature when the order is first generated and the Supervisor signs any orders that it approves. Both these signatures are checked by the Order Department to ensure the order received is the same one which was approved which in turn was the same order requested by the Project Client.

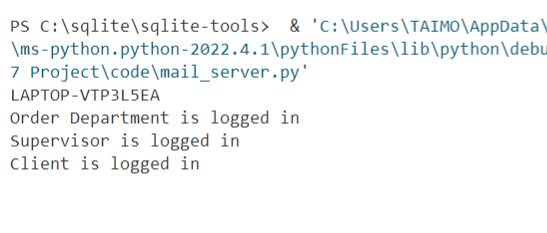
The next technique used a symmetric session key in place of the RSA encryption for sending the Purchase Order between modules. Using a one time session key for each connection between the modules provides increased confidentiality by ensuring that if one session key is compromised only a single connection is vulnerable and only for that session. The session key is generated and distributed during the authentication process and then used to encrypt each Purchase Order using DES.

The final technique used a password to ensure that the user operating the Project Client workstation is in fact authorized to do so. This prevents unauthorized individuals from forging an order using someone else's work station. The Project Client accomplishes this by requiring a password be entered on startup and only progressing once the correct password has been entered. It is assumed that only employees authorized to make Purchase Orders will know the password and that they will not share it or otherwise make it accessible to unauthorized individuals. As a result only authorized users will be able to generate Purchase Orders.

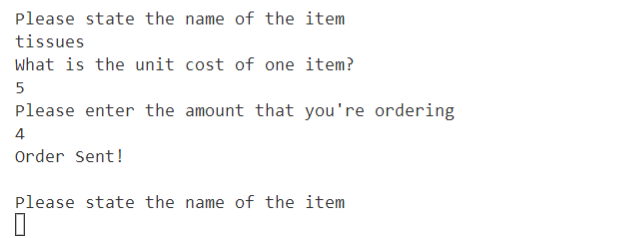
**Results:**



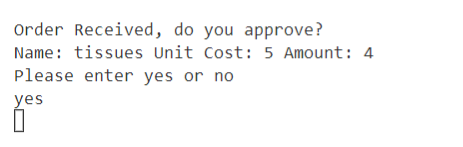
**Figure 4**: Output of the Project\_Client after logging in and authentication



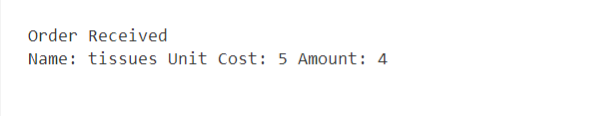
**Figure 5:** Output of the mail\_server after the order department, supervisor and client have logged in.



**Figure 6:** Client inputs all order information and system sends it off to the supervisor for approval



**Figure 7:** Supervisor receives and approves the order requested by the client.



**Figure 8:** After the order is approved by the supervisor, the order department receives the approved order with the complete order details.

As described in the Design and Implementation section, the sequence of the order flow is chronologically illustrated in **Figures 4-8** with sample inputs and all of their outputs.

In order to recreate the results, a user may run all 4 modules and enter in the password “admin” into the Project\_Client.py module. Afterwards, they may follow the screenshots above in the associated modules to recreate the results.

**Conclusion:**

Through this project Riley gained more experience using public key encryption to establish a trusted socket connection between multiple entities and distribute a symmetric session key for communicating over the socket. She also gained further experience implementing the security techniques discussed in this course (such as nonces, timestamps, encryption techniques, signatures, etc.) as well as learned how to apply these techniques to design a secure solution to a security problem. Riley attended and participated in all the planned group meetings over the course of the project to work collaboratively with her group members to design the project and solve any difficulties encountered. She also worked independently on her tasks for the project as required. For the code these tasks included completing the supervisor.py module and integrating the final draft of each module together to ensure their smooth operation. For the report Riley wrote the brief description of the design and implementation, the description of the other security techniques used, as well as completed her section of the conclusion.

In this project Aiyang followed the directions given by other members. The code of the order department was formed in accordance with the specifications outlined. The first draft of the orderDepartment.py was modeled after the supervisor.py. The communication format between the various modules was determined in meetings amongst the group members. Sections of the architectural diagram and security protocols were completed.

In this project, Rick’s assignment is to make a mail server for the communication between 3 users, client, order department and supervisor. Since Riley finished her supervisor.py first, what Rick did is used her code as a guideline to finish up the authentication. Next, he uses multithreading and creates 3 different functions for dealing with each user. Also, he creates a tool.py file that contains several reusable functions, taken from Riley’s supervisor code for consistency. For the project, Rick wrote some about design and implementation.

Throughout this project, Taimoor was responsible for the Project\_Client.py source code which allowed the user to represent themselves as a “Buyer” and input the item name, unit price and total amount of items they would be purchasing. After the user would input their desired items, a signature and timestamp was incorporated and appended to the order prior to sending it to the supervisor for approval. Taimoor also ensured that the socket connection between the Project\_Client.py and server was secure and authenticated using the tools.py file available within the project directory. After the order’s details are signed and timestamped, Taimoor encrypted the order using the encrypt function and send off the order to the supervisor for approval. In the project report, Taimoor wrote the introduction, results and his personal conclusion paragraph.