Testing Procedures

# Login – Nonce comparison

1. Client Login Initiation:

* The client enters their email and passphrase to start the login process.

1. Public Key Retrieval:

* The Certificate Authority (CA) imports the client’s public key. This key is retrieved from the ‘users.json’ file using the client’s email.

1. Nonce Creation and Encryption:

* The CA creates a nonce (a random number used once).
* The CA encrypts the nonce with the client’s public key (obtained via their email).
* The encrypted nonce is sent to the client.

1. Client Decrypts and Re-encrypts the Nonce:

* The client decrypts the nonce using their private key.
* The client then encrypts the same nonce using the CA’s public key.
* The client sends the re-encrypted nonce back to the CA.

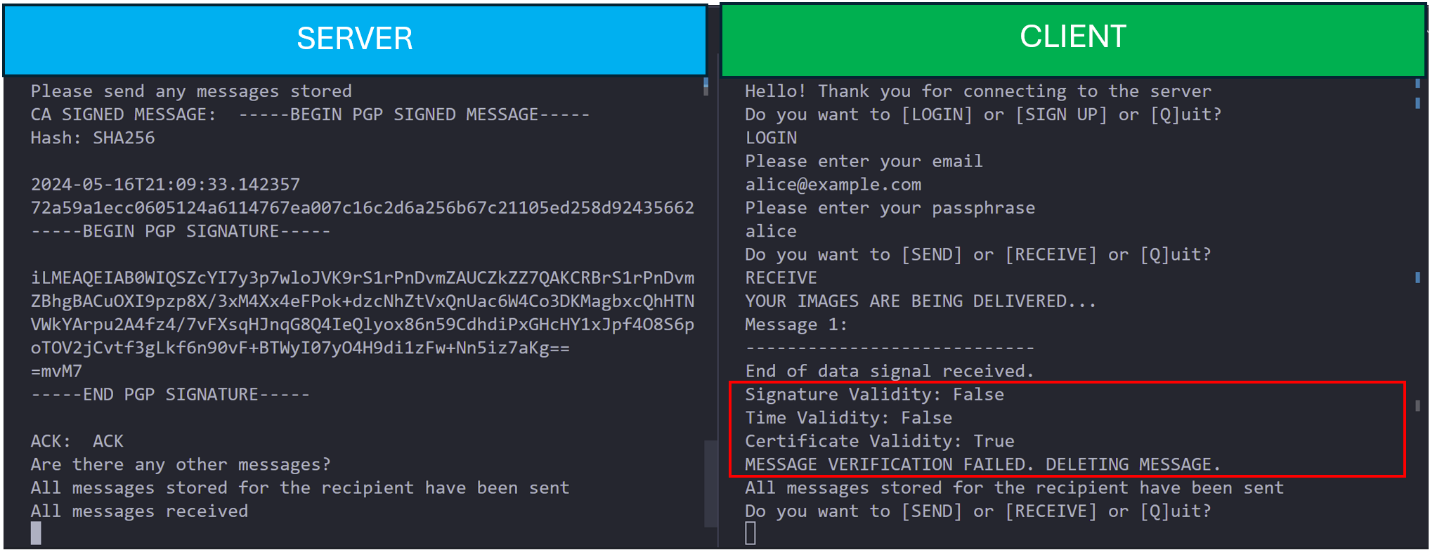
1. Nonce Verification:

* The CA decrypts the received nonce using its private key.
* The CA compares this decrypted nonce with the original nonce it created.
* If the nonces match, the login attempt is deemed successful.

This process ensures authenticity during login. The CA can confirm that the client is indeed the true client since they were able to decrypt the nonce encrypted with their public key and re-encrypt it with the CA's public key.

# Penetration Testing

## Man in the Middle Attack



Description: There are two potential cases of a man in the middle attack. The above image displays a message which has had its contents altered. In both of the below cases, it will result in a message verification failure as shown above.

1. A client attempts to send a message, a bad actor intercepts this message, alters the content of the message and sends it to the server.
   * This attack will not succeed as the attacker would need access to the sender’s private key in order sign the message digest, thus leading to a failed verification and the message will be deleted for safety. Attack unsuccessful.
2. The server sends a stored message to the rightful recipient on request, a bad actor intercepts this message, alters the content of the message and sends it to the recipient.
   * This attack will not succeed as the signature validity fails due to the altered message not having been signed by the CA, thus leading to a failed verification and the message will be deleted for safety. Attack unsuccessful.

## Replay Attacks

A screenshot of a computer

Description automatically generated

Description: A replay attack in which an attacker intercepts and captures a valid data transmission from the server to the client, then fraudulently retransmits it to the client. There are two cases to consider. In both of these cases, the user will be notified and will result in a message verification failure and the deletion of the message as a safety precaution.

1. Message arrival within window
   * Timestamps are utilised in this system and compared such that when the server sends a message to the client, the time difference is measured between sending and arrival time. If the difference exceeds the time window, it fails the time validity check and therefore the message verification fails, leading to a deleted message for safety. For this system, the appropriate window was found to be 10 seconds and as seen above, the replay attack was unsuccessful due to the fraudulent message arriving in a time exceeding the window.
2. Use of a previous CA signature
   * A nefarious actor can intercept a RECEIVE message sent by the server, which contains the CA’s signature of the hash of the entire base-64 message based on the timestamp, in the header of the message – which is not encrypted and easily accessible. If someone tries to reuse this signature contained in the header, this will fail on two fronts. Firstly, the verification of the CA signature will fail as the contents of the message which was signed will differ from the original contents signed by the CA. Secondly, the timestamps with the signature will be compared, as described in the first case above, and the signature will be found to be stale.

## Certificate and Signature Validation:

A screenshot of a computer

Description automatically generated

Description: There are three cases to be checked. In the first two cases, it will lead to a message verification failure and a permutation of the printed validity checks as seen above.

1. Verify the certificate of the sender upon a client receiving a message.
   * If the verification of the sender’s certificate returns false, this means that the certificate presented by the sender is fraudulent, the ‘CERTIFICATE VALIDITY’ will be false and will lead to a message verification failure.
2. Verify the CA’s certificate in the header of the message.
   * A check to confirm that the CA’s certificate in the header does indeed belong to the CA. If the CA’s certificate is fraudulent, the ‘SIGNATURE VALIDITY’ will be false and will lead to a message verification failure.

A screenshot of a computer

Description automatically generated

1. Verify that the signed message digest was indeed signed by the sender.
   * If the signed message digest was in fact not signed by the sender it will result in a “signature bad” status and the validity will be false leading to a message verification failure, as seen above.

# Cryptographic Testing

## Base-64 Encoding of an Image

A screenshot of a computer program

Description automatically generated

Description: the above shows the first and last 50 characters of the base-64 encoding of the image (string would be far too large to print in its entirety).

## Session Key Generation

A screenshot of a computer

Description automatically generated

Description: the above shows the generation of the session key which is encrypted with the recipients public when sending a message and the subsequent receiving of the session key which is decrypted with the recipient’s private key.

## Compression and Decompression

## A screenshot of a computer program Description automatically generated

Description: the above image shows the original message size as well as the first and last 50 characters of the message before compression. The message is then compressed, reducing the total of number of bytes, as shown. The message is then sent. Once received, the compressed message received matches the compressed message before sending. Finally, the decompressed message matches the original message sent.

## Encrypted Payload and Encrypted Session Key

A screenshot of a computer program

Description automatically generated

Description: the above image shows the session key and nonce which is encrypted with the public key of the recipient followed by the payload which is encrypted with the session key.

## Base-64 Encoding of Entire PGP Message

## A screenshot of a computer screen Description automatically generated

Description: the above image shows the base-64 encoding of the entire PGP message. The first and last 100 characters are shown as the entire string would be too large to print in its entirety).

# Functional Testing

## Sign Up – Regex for email Submission

A screenshot of a computer

Description automatically generated

Description: the above image shows that on sign up, a user must enter an email conforming to the format of text followed by an ‘@’ and a ‘.’ after the ‘@’.

## Sign Up – Can’t Sign Up with an Existing User’s Email

A screenshot of a computer screen

Description automatically generated

Description: the above image shows the handling of a sign up when the user attempts to sign up with an email that is already registered. The user will be notified that the email cannot be used and redirected back to the main menu.

# Login – Nonce comparison

A screenshot of a computer program

Description automatically generated

Description: since passphrases are neither sent nor stored in any capacity, a login is only successful if the exchanged encrypted nonce, and subsequently decrypted nonce, matches with the original nonce created. As seen above, the nonces match and thus the login attempt was successful.

A screenshot of a computer program

Description automatically generated

Description: in the above case, the incorrect passphrase is entered, thus when the original nonce is encrypted with the client’s public key, it cannot be correctly decrypted, and subsequently encrypted with the CA’s public key, thus leading to the nonces not matching. This leads to the client being disconnected from the server.

## Login – Incorrect/Unregistered Email

A screenshot of a computer program

Description automatically generated

Description: the above image shows that if an email is entered on login that is not registered, the user will be notified and asked to resubmit an email with a registered account.

\*Note: Incorrect passphrase has been shown in the Nonce Comparison section

## Sending – Can Only Send to Existing Users

A screenshot of a computer

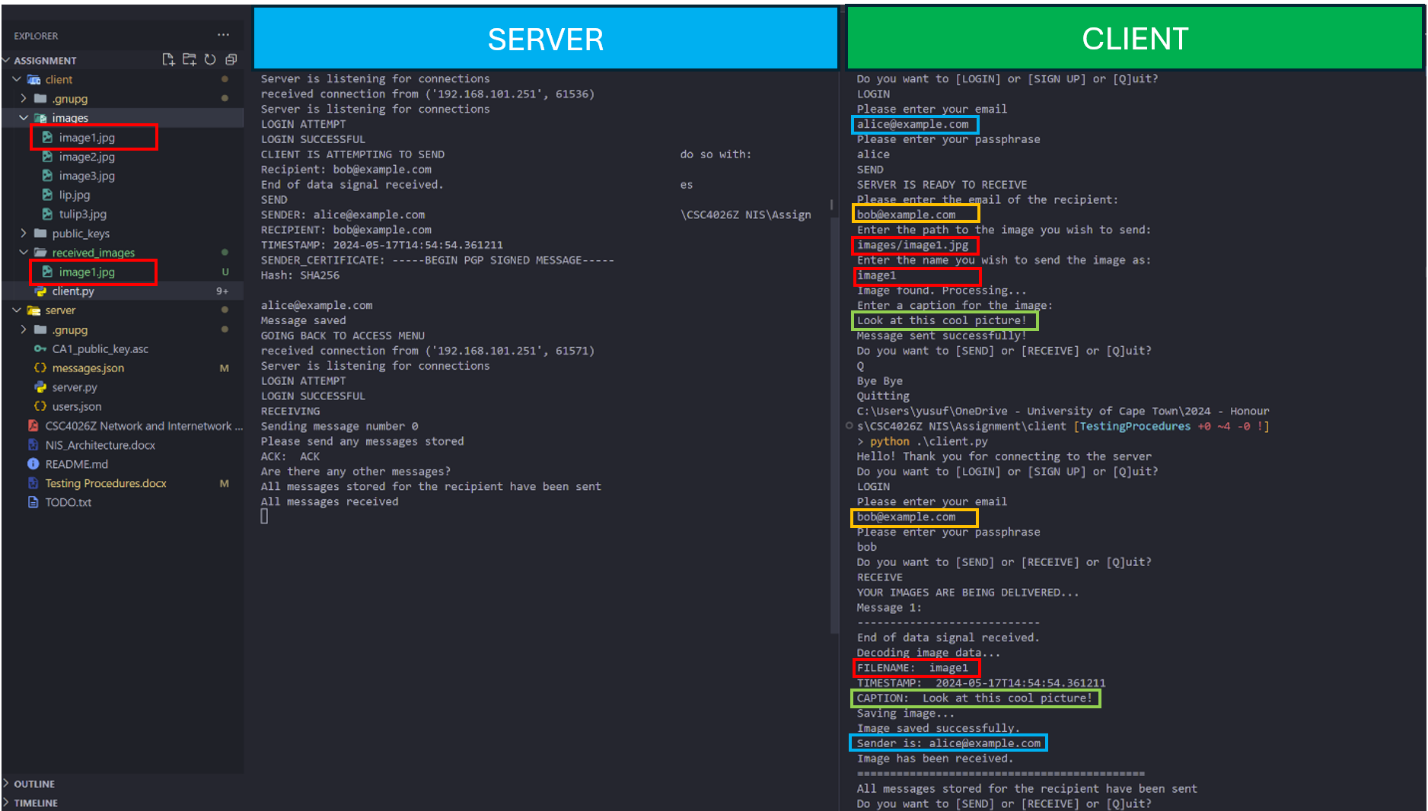
Description automatically generated

Description: the above image shows that messages can only be sent to registered users on the platform.

## A screenshot of a computer Description automatically generatedSending – Image not Found

Description: the above image shows that in the case of a user entering an incorrect path to the image, they will be prompted to resubmit the path to the image they wish to send.

## Successful Send and Receive



Description: the above image shows a full end-to-end sending and receiving of a message. Alice logs in and chooses to send an image, submits Bob as the recipient, specifies the file path and the name of image she wishes to send it as alongside a caption, she is the notified the message was sent successfully to the server. Bob then logs in, makes a request to receive all messages meant for him and gets sent the message from Alice, including the filename, timestamp, caption and sender. The image is then saved and can be seen in the ‘received\_images’ directory. Bob is the notified that all messages for him has been sent.