**Programming Assignment-4**

Abhinav Ujjawal (2021120) & Nikhil Suri (2021268)

**Assumptions:**

* We can handle all kinds of files, but we’ve used PDF and JPG for demonstration purposes.

**Project no. 0: Securely time-stamping a document**

**(Bonus included: It works for both pdf and jpg format files)**

This application relates to securely time-stamping a document. The process envisaged is to upload the document to the date/time stamping server (or perhaps some version of the document) and expect to receive the same but with the current date and time stamped onto the document. It uses that to time-stamp documents (in some standard format) with the current GMT date/time and a digital signature. We have used gRPC for the purpose of communication.

The following section answers all the questions that have been asked in the assignment:

1. **How and where do you get the correct GMT date and time? Your laptop or the local Linux server is not good enough.**

**Answer:** The correct GMT date and time are obtained from a trusted external time server. In the provided code, the system fetches the GMT from the [World Time API](https://worldtimeapi.org/api/timezone/Etc/UTC) using a **secure HTTPS** **request**. This ensures that the time obtained is accurate and synchronised with **reliable** sources.

1. **When is the correct GMT date/time obtained?**

**Answer:** The correct GMT date and time are obtained when the document is submitted for time stamping to the **server**. This ensures that the time stamp accurately reflects the time when the server processed the document.

1. **Is the source reliable? Is the GMT date and time obtained securely? The term ‘obtained’ refers to the security of communication.**

**Answer:** The source for obtaining the GMT date and time, that is, the World Time API, is indeed **reliable** as it provides **accurate** time information from reputable time servers. The communication with the time server is also **secure**, as the code uses **HTTPS** to fetch the time information. This ensures that the GMT date and time are obtained **securely** without the risk of tampering or interception during communication.

1. **How do you ensure privacy if the server does not see/keep the original document?**

**Answer:** Privacy of the original document is ensured by sending the hash value of the document. This prevents the server from accessing the content of the document. **Only the hashed version of the document is sent to the server for time-stamping**, **ensuring that the server does not see or keep the original document.**

1. **How do you share the document with third parties in a secure manner with the GMT date/time preserved and its integrity undisturbed?**

**Answer:** To securely share the document with third parties, the client can send them the document over a secure SSL/TLS connection, along with the time-stamped verification (timestamp and signature) obtained from the server. The third parties can then use the provided verification information to independently verify the authenticity and integrity of the document. This ensures the third parties can get the document without its integrity being compromised and the GMT date/time preserved.

1. **How does one ensure that the user (both the owner and anyone else verifying the date/time) uses the correct “public key” of the server stamping/signing the “GMT date/time”?**

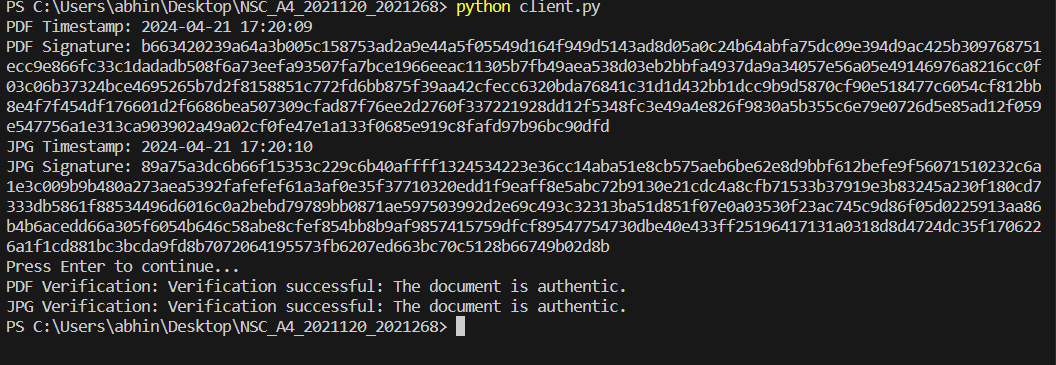
**Answer:** To ensure that users (both the owner and anyone verifying the date/time) use the correct public key of the server for stamping/signing the GMT date/time, a secure method of key distribution is necessary. This involves publishing the public key through trusted channels. Additionally, clients can verify the authenticity of the server's public key during the initial handshake or through other secure mechanisms to prevent man-in-the-middle attacks and ensure the correct usage of the public key. However, this is not required in our implementation since we’ve used secure SSL/TLS communication between the verifier and the server for secure sharing of keys.

**\*Note that we have used PDF and JPG files for demonstration in our code.**

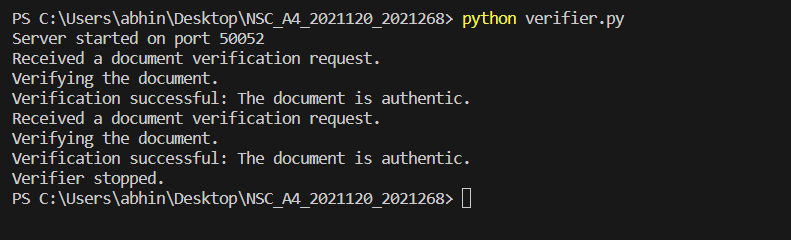
**Algorithm followed:**

* The client sends the file hash (PDF and JPG) to the time-stamping server.
* The server connects with WorldTime API to access the current GMT and adds the relevant timestamp received from the API to the document (its hash in this case), and sends this timestamp back to the client.
* The client received the time-stamped document and sent it to the verifier for verification.
* The verifier successfully verifies the files (PDF and JPG) and regenerates them using the received bytes.

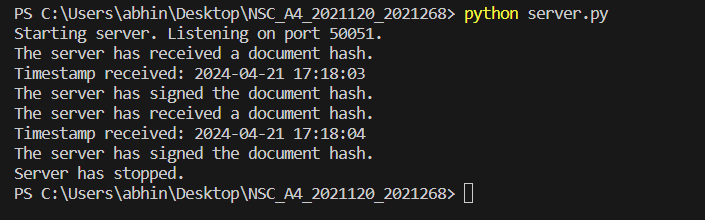
**Sample Inputs & Outputs**

1. ****

**Client**

1. ****

**Verifier**

1. ****

**Server**