Practica 2

El propósito de este documento es documentar el alcance y uso de la aplicación Cryptofinanciera

Desarrollada con python 3.8, la aplicación cliente está compuesta por tres módulos principales:

* Cifrado
* Signature
* Selfsigned\_cert

Por medio del método execute\_mode del cliente se ejecutan las instrucciones posteriormente definidas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Cifrado** | | | | |
| Objetivo | Encargado de cifrar de forma simétrico o asimétrica, así como generar los mecanismos necesarios para poder llevar a cabo la encriptación. | | | |
| Metodos | Alcance | Parametros | Libreria | Referencia |
| encrypt\_simetrico | Cifra en modo AES | FicheroEntrada, FicheroSalida, Contraseña | pyAesCrypt | <https://pypi.org/project/pyAesCrypt/> |
| decrypt\_asimetrico | Descifra en modo AES | FicheroEncrptadoEntrada, FicheroDescifradoSalida, Contraseña | pyAesCrypt | <https://pypi.org/project/pyAesCrypt/> |
| encrypt\_asimetrico | Cifrado asimétrico RSA con llave publica y algoritmo SHA256 | FicheroEntrada, FicheroSalida | cryptography.hazmat | <https://cryptography.io/en/latest/hazmat/primitives/asymmetric/rsa/> |
| decrypt\_asimetrico | Cifrado asimétrico RSA con llave privada y algoritmo SHA256 | FicheroEncrptadoEntrada, FicheroDescifradoSalida | cryptography.hazmat | <https://cryptography.io/en/latest/_modules/cryptography/hazmat/backends/> |
| generate\_keys | Se crean las llaves pública y privada con un tamaño de 2048\*2 bytes para poder encriptar un archivo con un peso menor a 500 bytes |  | cryptography.hazmat | <https://cryptography.io/en/latest/hazmat/primitives/asymmetric/rsa/> |
| store\_keys | Almacena las llaves en sus ficheros respectivos | Llave publica, Llave privada | cryptography.hazmat | <https://cryptography.io/en/latest/_modules/cryptography/hazmat/backends/> |
| create\_keys | Encargada de orquestar la generación y guardado de las llaves pública y privada |  |  |  |
| read\_public\_key | Obtiene la llave publica de su fichero respectivo |  |  |  |
| read\_private\_key | Obtiene la llave privada de su fichero respectivo |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Signature** | | | | |
| Objetivo | Firma de fichero | | | |
| Metodos | Alcance | Parametros | Libreria | Referencia |
| sign\_file | Se firma el fichero utilizando hash256 | FicheroEntrada | hashlib RSA | [https://docs.python.org/3/library/hashlib.html https://pycryptodome.readthedocs.io/en/latest/src/public\_key/rsa.html](https://docs.python.org/3/library/hashlib.html) |
| verify\_signature | Se verifica el hash256 con el hash generado sobre la firma | FicheroEntrada, firma, tuplaRSA(modulo'exponente privado) | hashlib RSA | [https://docs.python.org/3/library/hashlib.html https://pycryptodome.readthedocs.io/en/latest/src/public\_key/rsa.html](https://docs.python.org/3/library/hashlib.html) |

Ejecución y funcionalidades:

El programa se invoca de la siguiente forma: CriptoFinanciera.exe -m [cs | ds | h | vh | ca | da | cert | ts | tsv] [-p contraseña] -i fichero.xml [-ad adicionales.txt] [-o salida.cpt]

-m indica el modo:

--help: despliega interfaz de ayuda

-cs/ds: cifrado/descifrado simétrico. La contraseña se establecerá con -p

-h/vh: función resumen y su verificación

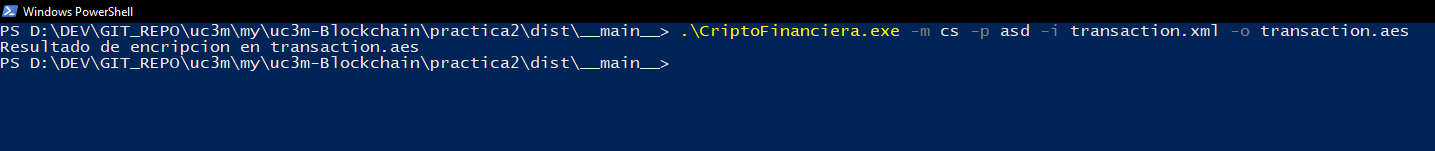
-ca/da: cifrado/descifrado asimétrico. Las claves pública/privada se guardan en la misma carpeta

-cert: Creación de un certificado X.509, cuyo nombre se especificará con -o

Ejemplos:

Descifrado simétrico:

.\CriptoFinanciera.exe -m ds -p asd -i transaction.aes -o transaction\_descifrada.xml

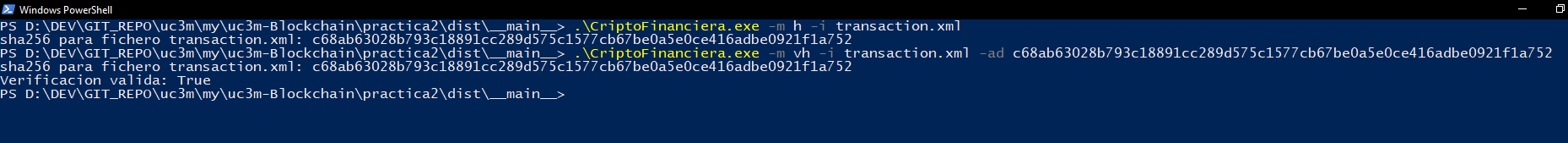


Función resumen (calculo sha256):

.\CriptoFinanciera.exe -m h -i transaction.xml

Función verificación sha256:

.\CriptoFinanciera.exe -m vh -i transaction.xml -ad c68ab63028b793c18891cc289d575c1577cb67be0a5e0ce416adbe0921f1a752

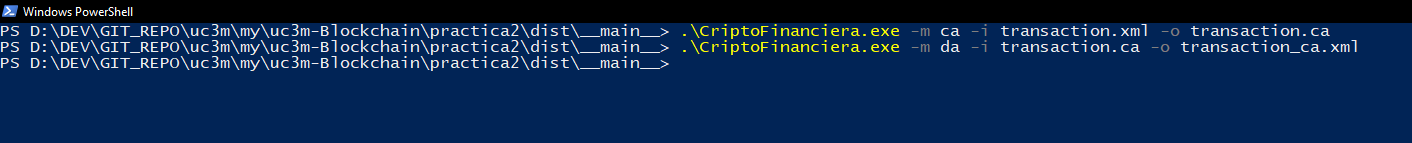


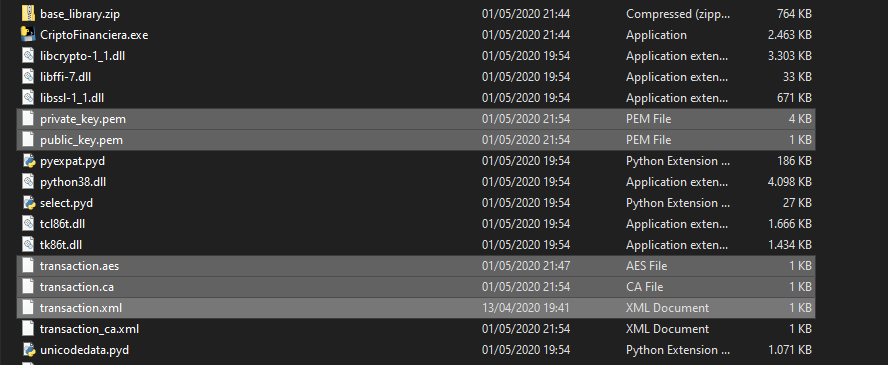
Cifrado asimétrico:

.\CriptoFinanciera.exe -m ca -i transaction.xml -o transaction.ca

Descifrado asimétrico:

.\CriptoFinanciera.exe -m da -i transaction.ca -o transaction\_ca.xml





Creación certificado X509:

.\CriptoFinanciera.exe -m cert -o mi\_cert.crt



