Secure Storage of Crime Data Using Modern Cryptography

- PROGRAMMING LANGUAGE
- Programmed in Python utilizing numpy
- Tested on Windows 10 and Ubuntu Linux
- Fully compatible with python2.x
- Python3.x NOT supported

EXECUTION INSTRUCTIONS:

- -Make sure you have both Python2.x and pycrypto installed
- -Download the zip folder and store it on your hard drive

- -Open your terminal and navigate to the directory in which you stored the folder

- -type cipher.py and press enter

USING THE SOFTWARE:

-Running the cipher.py file will display command line arguements

---

./cipher <CIPHER NAME> <KEY> <ENC/DEC> <INPUTFILE> <OUTPUT FILE> <--OPTIONS/-O>

Supported Ciphers:

- DES: Indicates the 64bit DES cipher

- DES-CBC: DES Cipher in CBC Mode

- DES-CFB: DES Cipher in CFB Mode

- AES: Indicates 128bit AES cipher

- AES-CBC: AES Cipher in CBC Mode

- AES-CFB: AES Cipher in CFB Mode

--OPTIONS - Optional setting: If enabled will ask for converting

to lowercase and removing non-alpha characters

-Type ./cipher and do the following on the same line:

-The abbreviation of the cipher name

-The key (depending on the cipher the key format will be different)

-DES takes hex characters of 16 bits

-AES takes hex characters of 32 bits

-Either type ENC to encrypt or DEC to decrypt

-Type the name of the input file, including the extension

-Type the name of the output file, including the extension

-Optionally type -O to ask to convert the input file to lower case and remove special characters

-Press the enter key

-Within the terminal your input and output will be displayed along with if execution was a success or any errors were found

-Your encrypted or decrypted message will be stored in the output file within the directory you are in as well as displayed within the terminal

* - please note that if you set your own IV you must use the same one for both encryption and decryption

---

-The following is an example of encrypting with a DES-CFB cipher, while using the key "0123456789ABCDEF" and reading in from the file "input.txt" and outputing to the file

"output.txt"

./cipher DES-CFB "0123456789ABCDEF" ENC input.txt output.txt

EXTRA CREDIT:

-We did implement the extra-credit portion of the assignment

-An Initialization Vector (IV) is being used for the extra credit nodes

-If the IV is not set by the user it will be randomly generated
-If it's randomly generated it will add it to the first bytes of the file
-This means when decrypting the cipherText, you must select randomly generated so that it knows to strip the first bytes of the ciphertext

-CFB shifts 1 byte (8 bits)

EXAMPLES OF RUNNING EACH CIPHER

DES

ENCRYPT

cipher.py DES "aabbccddeeff0000" ENC input.txt output.txt

INPUT:

thisisatest

OUTPUT:

S¢ ë⍰|╧j┐┐!D╚ +V

Success!

DECRYPT

C:\Users\Matt\Desktop\modern_ciphers>cipher.py DES "aabbccddeeff0000" DEC output.txt test.txt

INPUT:

S¢ ë⍰|╧j┐┐!D╙ +V

OUTPUT:

thisisatest

Success!

DES-CBC

ENCRYPT

cipher DES-CBC "aabbccddeeff0000" ENC input.txt output.txt

Do you want to enter your own Initialization Vector (Y/N): n

Randomly Generated IV: fa5e8dd2104a258f

INPUT:

thisisatest

OUTPUT:

·^ì╥ J%Å═éüI rA~*oçB¡àú

Success!

DECRYPT

cipher DES-CBC "aabbccddeeff0000" DEC output.txt test.txt

Do you want to enter your own Initialization Vector (Y/N): n

INPUT:

·^ì╥ J%Å═éüI rA~*oçB¡àú

OUTPUT:

thisisatest

Success!

DES-CFB

ENCRYPT

cipher DES-CFB "aabbccddeeff0000" ENC input.txt output.txt

Do you want to enter your own Initialization Vector (Y/N): n

Randomly Generated IV: 892fc46684741867

INPUT:

thisisatest

OUTPUT:

ë/–fät gª╩╩ΓÑ   n╣╠┌xa

Success!

DECRYPT

cipher DES-CFB "aabbccddeeff0000" DEC output.txt test.txt

Do you want to enter your own Initialization Vector (Y/N): n

INPUT:

ë/–fät gª╩╩ΓÑ   n╣╠┌xa

OUTPUT:

thisisatest

Success!

AES

ENCRYPT

cipher AES "aaaabbbbccccddddeeeeffff00000000" ENC input.txt output.txt

INPUT:

thisisatest

OUTPUT:

▒╬~Φ╥≥DPts ¿∞òⁿ╤

Success!

DECRYPT

C:\Users\Matt\Desktop\modern_ciphers>cipher AES "aaaabbbbccccddddeeeeffff00000000" DEC output.txt test.txt

INPUT:

▒╬~Φ╥≥DPts ¿∞òⁿ╤

OUTPUT:

thisisatest

Success!

AES-CBC

ENCRYPT

C:\Users\Matt\Desktop\modern_ciphers>cipher AES-CBC "aaaabbbbccccddddeeeeffff00000000" ENC input.txt output.txt

Do you want to enter your own Initialization Vector (Y/N): n

Randomly Generated IV: 27529a449aa4c836d400c0b4f72bc09d

INPUT:

thisisatest

OUTPUT:

'RÜDÜñ╚6╞└┤≈+└¥╚ £ç«α °╬╢ u`z┐ì

Success!

DECRYPT

C:\Users\Matt\Desktop\modern_ciphers>cipher AES-CBC "aaaabbbbccccddddeeeeffff00000000" DEC output.txt test.txt

Do you want to enter your own Initialization Vector (Y/N): n

INPUT:

'RÜDÜñ╚6╞└┤≈+└¥╚ £ç«α °╬╠ u`z┐ì

OUTPUT:

thisisatest

Success!

AES-CFB

ENCRYPT

C:\Users\Matt\Desktop\modern_ciphers>cipher AES-CFB "aaaabbbbccccddddeeeeffff00000000" ENC input.txt output.txt

Do you want to enter your own Initialization Vector (Y/N): n

Randomly Generated IV: f03123a6dfdd04405a85bbb0dac28d60

INPUT:

thisisatest

OUTPUT:

≡1#ª▀ @Zà╗▒┌┬ì` >TªNb h ╬

Success!

DECRYPT

C:\Users\Matt\Desktop\modern_ciphers>cipher AES-CFB "aaaabbbbccccddddeeeeffff00000000" DEC output.txt test.txt

Do you want to enter your own Initialization Vector (Y/N): n

INPUT:

≡1#ª▀ @Zà╗▒┌┬ì` >TªNb h ╬

OUTPUT:

thisisatest

Success!