File Encryption for Secure File Sharing

Abstract:

Advanced end-to-end communication technology systems have led to a situation where almost all systems should implement security for data safety nowadays. This assignment has provided a knowledge over pseudo-random number generator, public-private key exchange methods and symmetric file encryption/decryption methods.

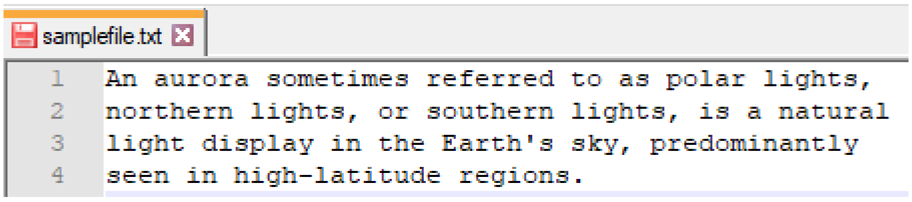
Introduction:

Diffie–Hellman key exchange is a method of secure communication over public channel. Where we can securely transfer data over the internet and between sender (alice) and receiver (bob) party. Traditionally in a real world example secure communication happens by exchanging keys or some physical means which is transported by a trusted partner [1]. The Diffie–Hellman key exchange method overcomes this problem by without having knowing each other with helps of shared secret key over an insecure channel [1]. With DH key exchange we can generate private, public and shared key. For more secure communication we can also use pseudo random number generator (PRNG). Blum Blum Shub (BBS) is a very popular is this case. In BBS we need to take 2 large prime number along with a seed value [2]. The equation is $x_{n+1} = x_n^2 \bmod M$. Here M = pq is the product of two large primes p and q [2]. This will enhance the security of shared key which we got from the DH key exchange. After that 128-bit AES encryption used to encrypt/decrypt the file and then it can be sent over the receiver party.

Design and Implementation:

The main program divided into 4 parts. The coding language used is python. Here I have implemented 3 class and a main driver program. First this program will take user input for private key which will be using in DH class. There will create 2 instances. One is for alice and another is for bob. Generator g=2 already predefined and prime p = 2q + 1 will generate using random number. Here I have created a method so that we can ensure that generated random number is initializing as a valid prime. As we already took the input of private key then this program will for generating the public key using the formula of pow (generator, privateKey, prime). After that program will generate the shared key using md5 which will give a 128-bit shared key. After getting the shared key CSPRNG class will enhance the security of shared key and will produced another key for next step file encryption. For file encryption system python package pycryptodome was used. Program takes the key which we got from CSPRNG class and encrypt the file and its information using that key. Then decryption also done with that key. For testing purpose only file with .txt extension have used.

The blue marked text are used for input. Green marked text is user input and the red marked text used for the output of the file. Before execution of this program we must create a sample .txt file for encrypt its data. In this case I have created a file name samplefile.txt and in this file can contain any information. After that in the same directory 2 file will auto generate where we can find the encrypted and decrypted file information.



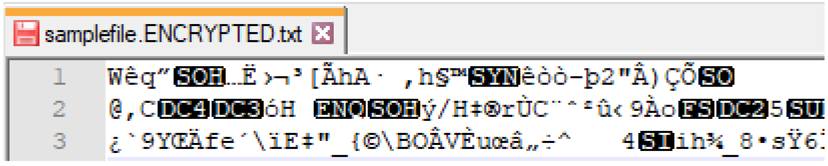
samplefile.txt

```
An aurora sometimes referred to as polar lights,
northern lights, or southern lights, is a natural
light display in the Earth's sky, predominantly
seen in high-latitude regions.
```

Figure 1: Sample of plaintext file

Test Results:

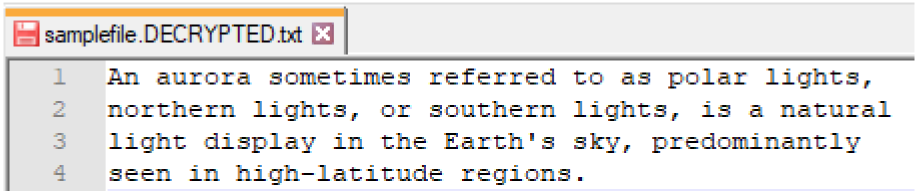
During test I observed that encryption and decryption is working properly and files are also generating.



samplefile.ENCRYPTED.txt

```
Wêq"SOH…Ë›¬³[ÃhA·  ,h§™SYNêòò-þ2"Â)ÇÕSO
@,CDC4DC3óH ENQSOHý/H‡®rÙC¨ˆ²û‹9ÀoFSDC25SU
¿`9YŒÄfe´\ïE‡"_{©\BOÂVÈuœâ„÷^    4SIih¾_8•sŸ6
```

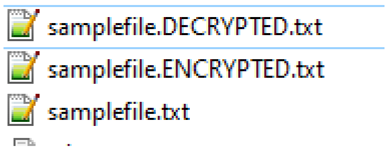
Figure 2: Encrypted file sample



samplefile.DECRYPTED.txt

```
An aurora sometimes referred to as polar lights,
northern lights, or southern lights, is a natural
light display in the Earth's sky, predominantly
seen in high-latitude regions.
```

Figure 3: Decrypted file sample

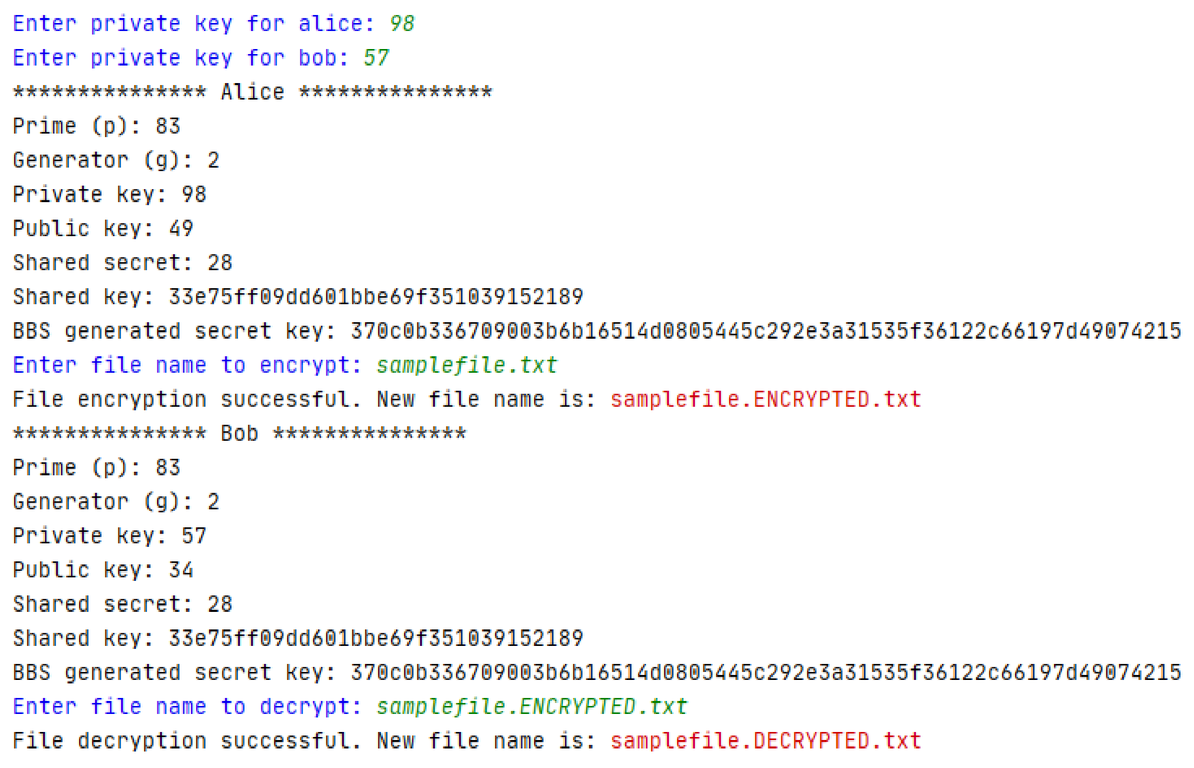


samplefile.DECRYPTED.txt
samplefile.ENCRYPTED.txt
samplefile.txt

Figure 4: Output files in directory

Discussion:

As per test analysis I can finalize that outcome of this program is good and I am very confident about my implementation.



```
Enter private key for alice: 98
Enter private key for bob: 57
*************** Alice ***************
Prime (p): 83
Generator (g): 2
Private key: 98
Public key: 49
Shared secret: 28
Shared key: 33e75ff09dd601bbe69f351039152189
BBS generated secret key: 370c0b336709003b6b16514d0805445c292e3a31535f36122c66197d49074215
Enter file name to encrypt: samplefile.txt
File encryption successful. New file name is: samplefile.ENCRYPTED.txt
*************** Bob ***************
Prime (p): 83
Generator (g): 2
Private key: 57
Public key: 34
Shared secret: 28
Shared key: 33e75ff09dd601bbe69f351039152189
BBS generated secret key: 370c0b336709003b6b16514d0805445c292e3a31535f36122c66197d49074215
Enter file name to decrypt: samplefile.ENCRYPTED.txt
File decryption successful. New file name is: samplefile.DECRYPTED.txt
```

Figure 5: Full output of the program

Conclusion:

This report is mainly focused on assignment task 1. I have tried to complete the task 2 but I was unable to implement it properly. So as I have only described the properties of task 1 and only limitation I have found that the file encryption method statically depends on text format document. In future development we can implement method to extend the feature of this program.

References:

[1] https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange

[2] https://en.wikipedia.org/wiki/Blum_Blum_Shub

[3] https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[4] https://www.di-mgt.com.au/public-key-crypto-discrete-logs-1-diffie-hellman.html

[5] https://medium.com/asecuritysite-when-bob-met-alice/the-cyclic-group-g-of-order-p-f9688dc9cc27

[6] http://dandylife.net/blog/archives/295

[7] https://eli.thegreenplace.net/2010/06/25/aes-encryption-of-files-in-python-with-pycrypto

[8] https://stackoverflow.com/questions/20852664/python-pycrypto-encrypt-decrypt-text-files-with-aes

[9] https://www.instructables.com/Printing-Colored-Text-in-Python-Without-Any-Module/