Wrocław 10.26.2022

C Y B E R S E C U R I T Y

L A B 4/\ Solutions

1. Introduction.

The laboratory presents issues related to asymmetric encryption. Asymmetric key cryptography relies on one key for encryption and a different but related key for decryption. It is very important to note that even with knowledge of the cryptographic algorithm and the encryption key, it is computationally infeasible to determine the decryption key. The advantage of the system is that each communicating party needs just a key pair for communicating with any number of other communicating parties. Besides, some algorithms, such as RSA, also exhibit an important characteristic, i.e., either of the two related keys can be used for encryption, with the other used for decryption.

Asymmetric cryptography is based on one-way functions - those that can be easily calculated in one direction, but very difficult to calculate in the other. There are two types of asymmetric algorithms. The first one, which is called "factoring problem," whose encryption security is based on the difficulty of factoring the product of two large prime numbers. For example, multiplication is easy and distribution into factors (factorization) is difficult (on which RSA is based, for example). The second type is algorithms based on elliptical curve cryptography whose security is based on the computational complexity of discrete logarithms on elliptical curves - known as Elliptic Curve Discrete Logarithm Problem (ECDLP). Modulo amplification is easy, and discrete logarithms are difficult (this applies to such algorithms as ElGamal, DSA, and ECC).

The main interest of the laboratory is the RSA algorithm. The RSA algorithm is the most popular asymmetric key cryptographic algorithm. The scheme is a block cipher. In this case, a plain text and a ciphertext are integers from 0 to N. A typical size for N is 1024 bits or 309 decimal digits, however, currently, 4096 bits are considered to be safe. This means that *N* is less than $^{1024}2$(for *N* equal 1024 bits). The principle of the algorithm is explained below.

Contrary to popular belief, Diffie-Hellman was the first publicly described asymmetric algorithm (not RSA). This cryptographic protocol allows two entities to establish a common key on an insecure channel. In other words, Diffie-Hellman is often used to allow actors to exchange a symmetrical key via some insecure medium such as the Internet. Despite the passage of time, the DH protocol is still widely used, for example in the most recent version of TLSv 1.3 (Transport Layer Security). ElGamal is based on the Diffie-Hellman key exchange algorithm and is used in some versions of Pretty Good Privacy (PGP).

The process of asymmetric cryptography is visualized in Figure 1. With public key/asymmetric cryptography, Alice will get Bob's public key and use that to encrypt the message she sends to Bob. If Eve intercepts the message and gains access to Bob's public key, that's OK, because that key won't decrypt the message. Only Bob's private key will do so, and this he safeguards.

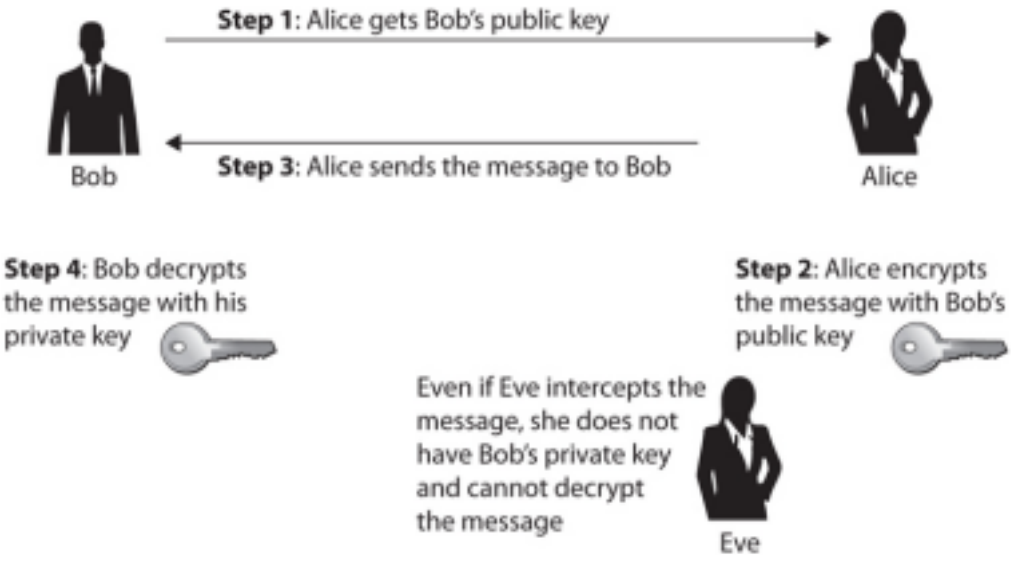


Figure 1 Exchange of messages with asymmetrical cryptography.

Description of asymmetrical cryptography from a mathematical point of view. A plain text is encrypted in blocks. For some blocks of plain text M and encryption block C, the whole process can be summarized as follows:

1. Select two large prime numbers $P\,And\,Q$, such that $P\,Q\neq$.
2. Calculate $N\,P\,Q$ = × .
3. Calculate ϕ = − − ( ) (1)(1) $N\,P\,Q$.
4. Select an integer E as the public key (i.e., the encryption key) such that it is not a factor of ϕ( ) N .
5. Calculate$D$as the private key (i.e., the decryption key) such that: (

)mod ( ) 1 $D\,E\,N$ × ϕ =

6. From the plain text, M the ciphertext is calculated as:

   mod $^{E}\,C\,M\,N$ =

7. This ciphertext is sent to the receiver for decryption.
8. The ciphertext is decrypted to the plain text as:

   mod $^{D}\,M\,C\,N$ =

2. Properties of the RSA algorithm.

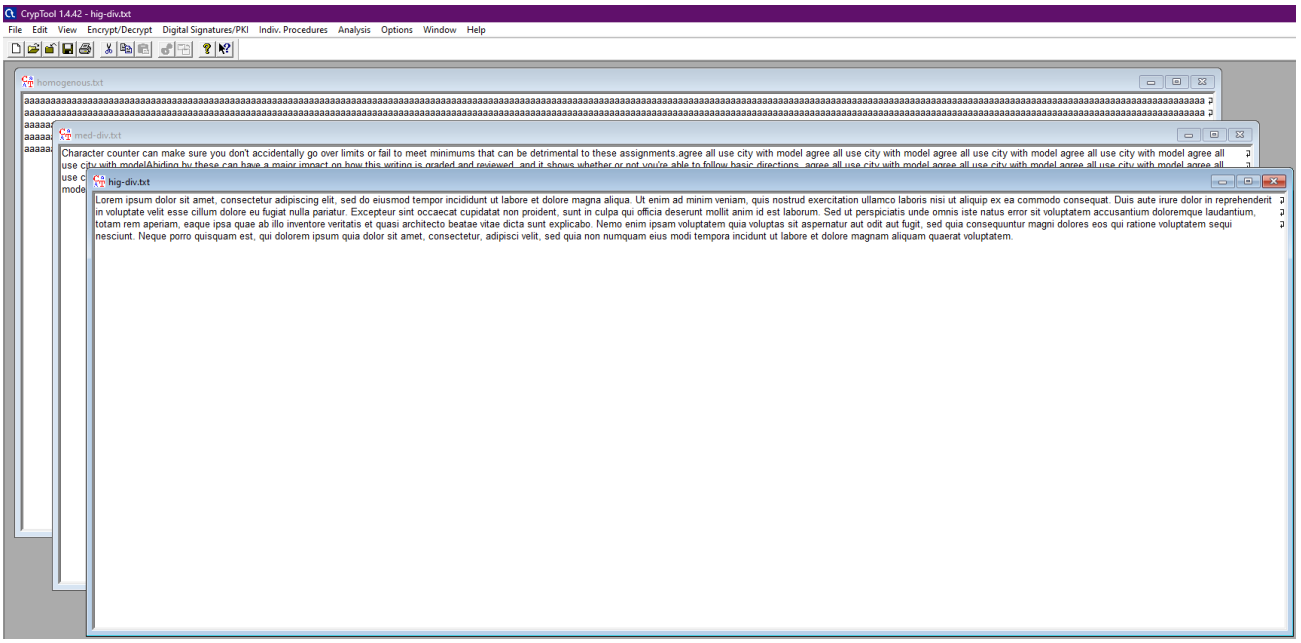
I. Tasks:

1. See the demonstration of the RSA algorithm (Encryption/Asymmetric/Demonstration RSA).

   - **Did read the given information and a little research from the internet.**

2. Prepare three plaintexts - homogeneous text, medium-diversified text, highly diversified text, (at least 1500 characters each).

-

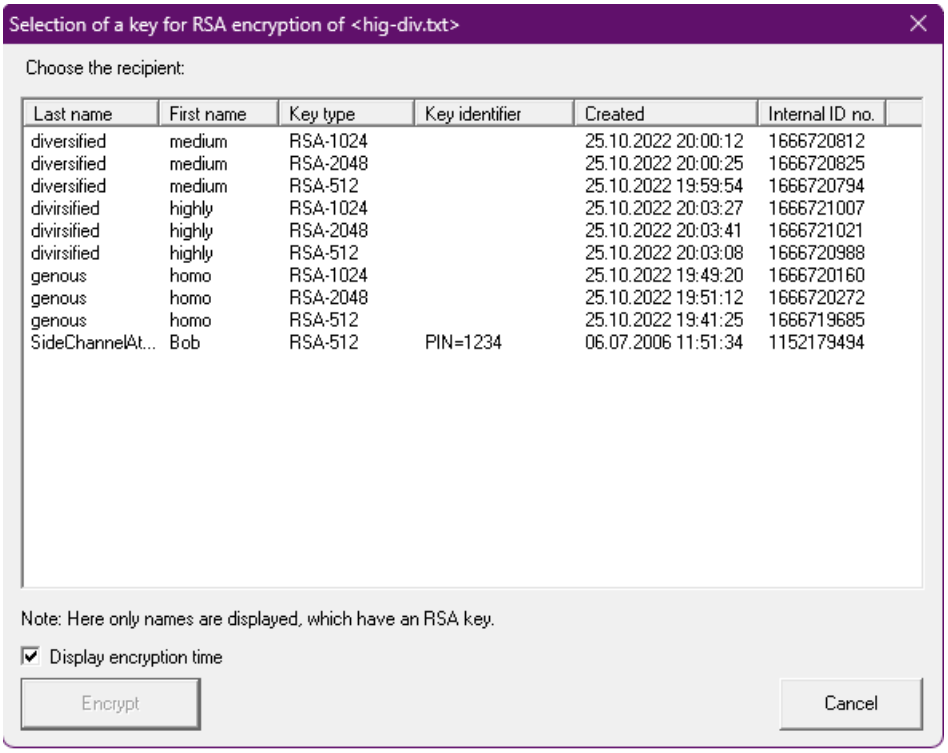


3. Generate the following cryptographic keys (Digital Signature/PKI/Generate/Import keys):

   a. RSA length 512 bits

   b. RSA length 1024 bits

   c. RSA length 2048 bits

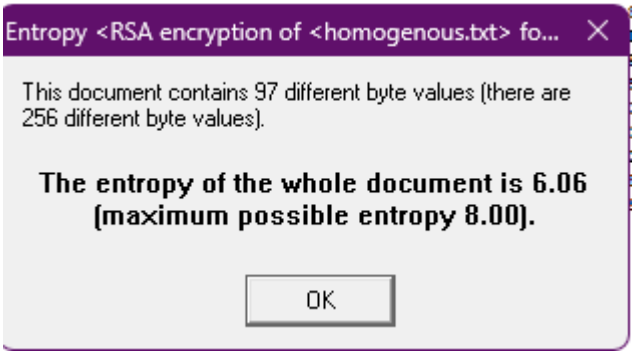


4. For different key lengths (512,1024,2048) and previously prepared plaintexts, compare the entropy of plain text with the entropy after encryption (encryption entropy). Check and compare the autocorrelation of the ciphertext and plaintext.

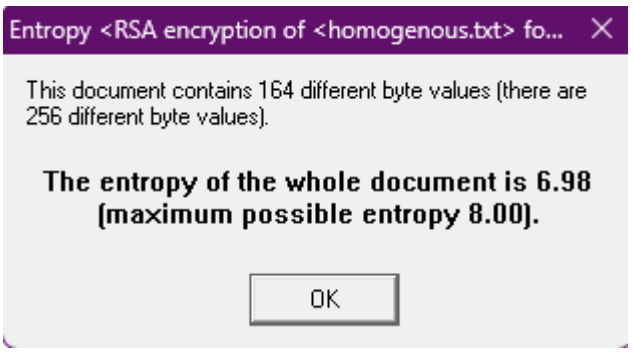
-

_entropy :

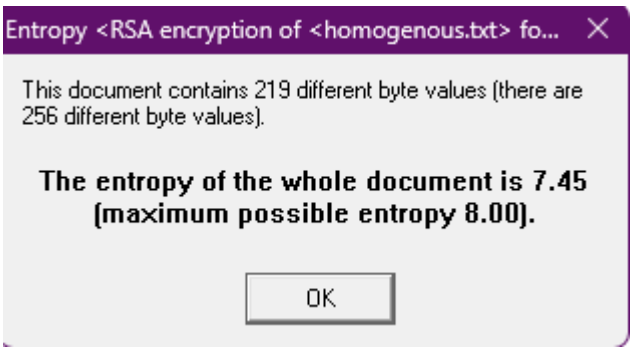
> homogenous, length 512 ;



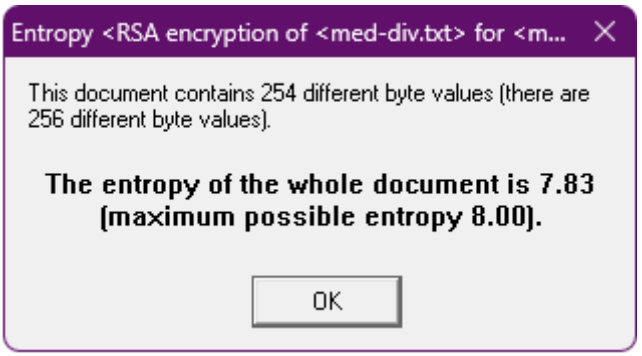
> homogenous, length 1024;



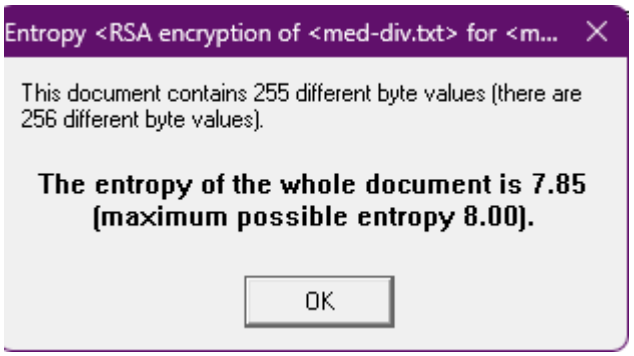
> homogenous, length 2048;



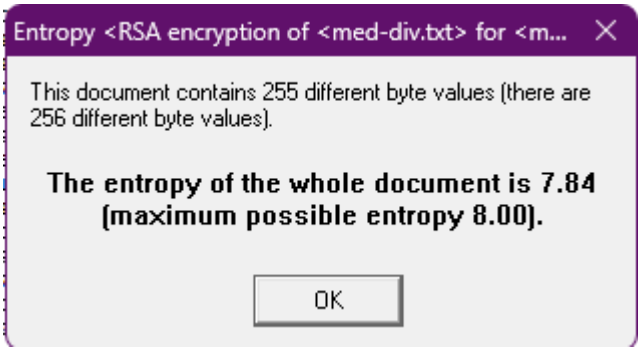
> mid-div, length 512;



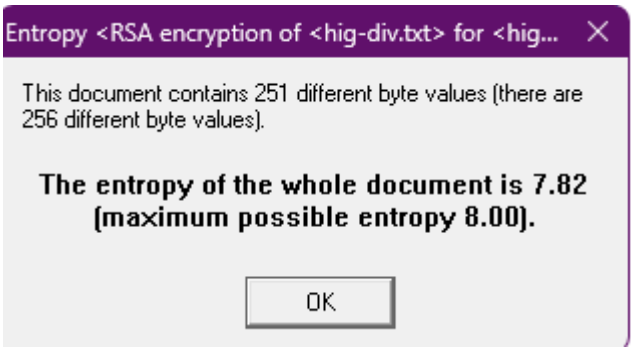
> mid-div, length 1024;



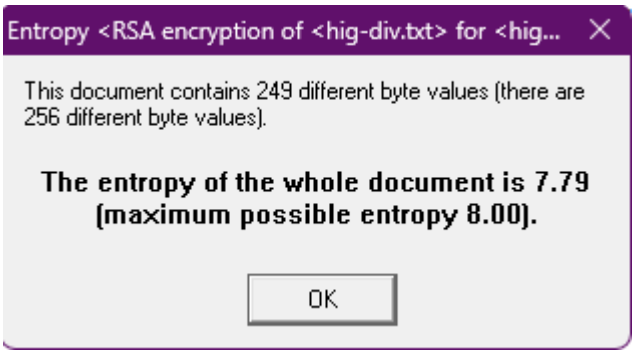
> mid-div, length 2048;



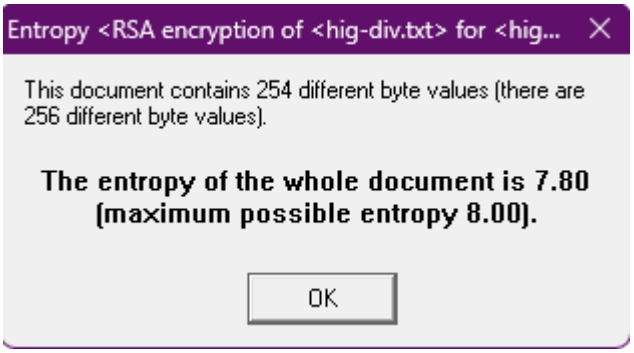
> hig-div, length 512;



> hig-div, length 1024;

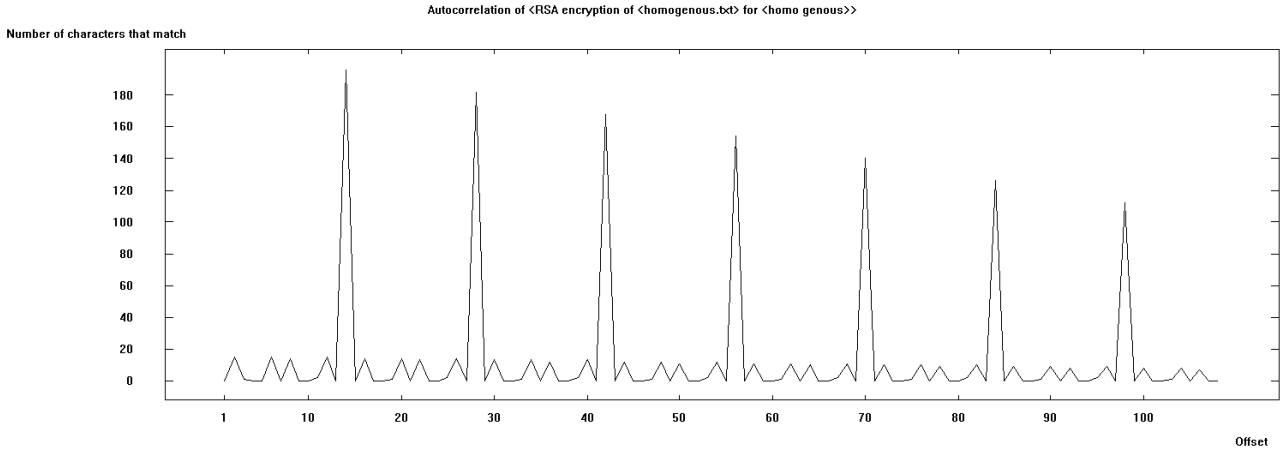


> hig-div, length 2048;

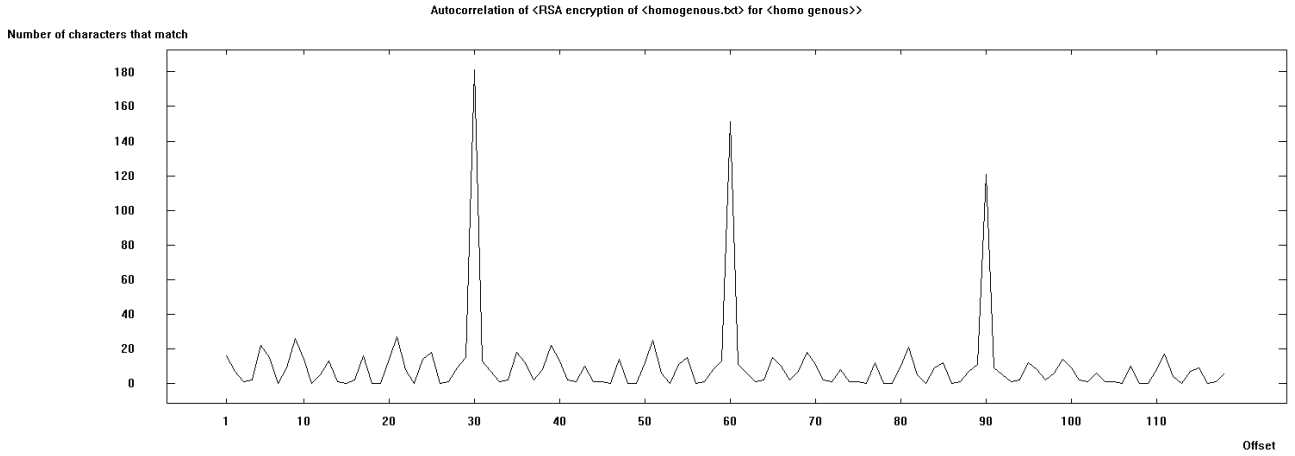


_Autocorrelation :

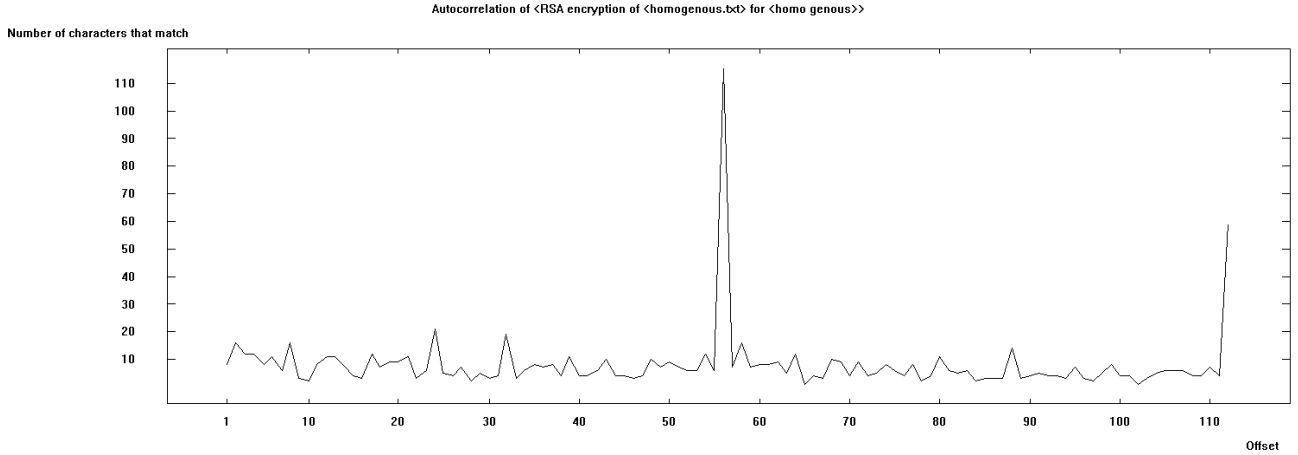
> homogenous, length 512;



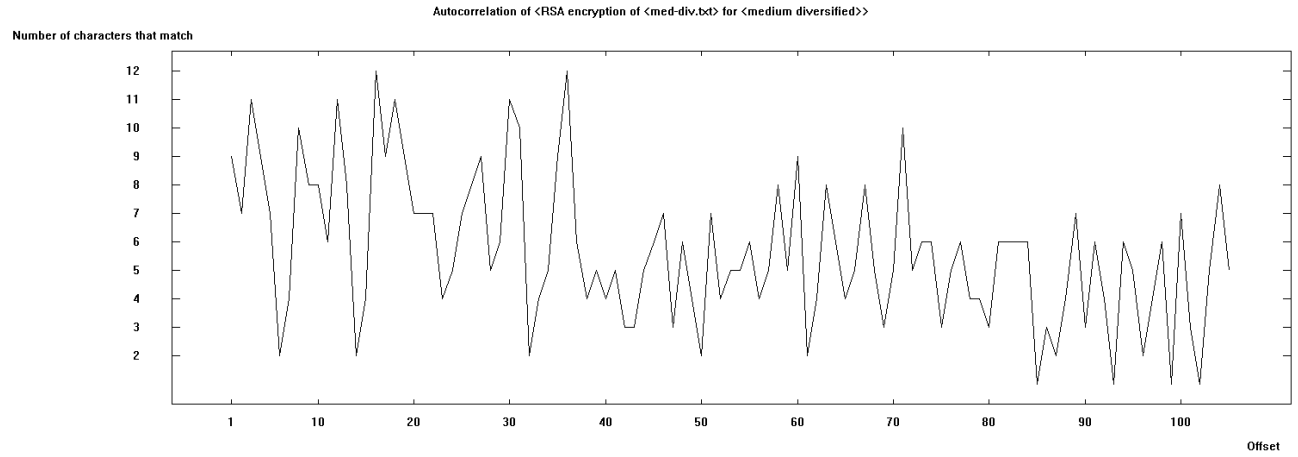
> homogenous, length 1024;



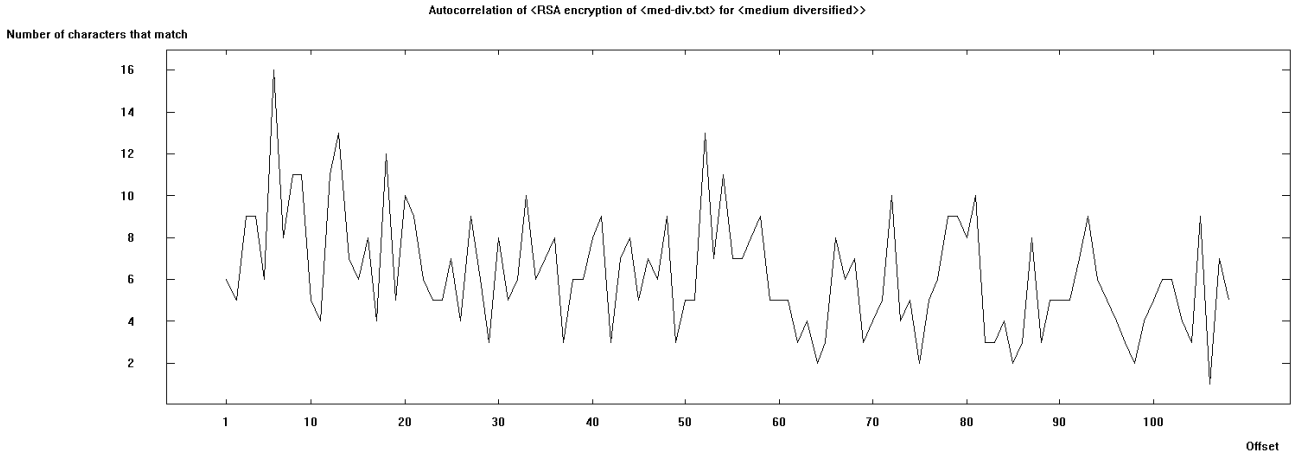
> homogenous, length 2048;



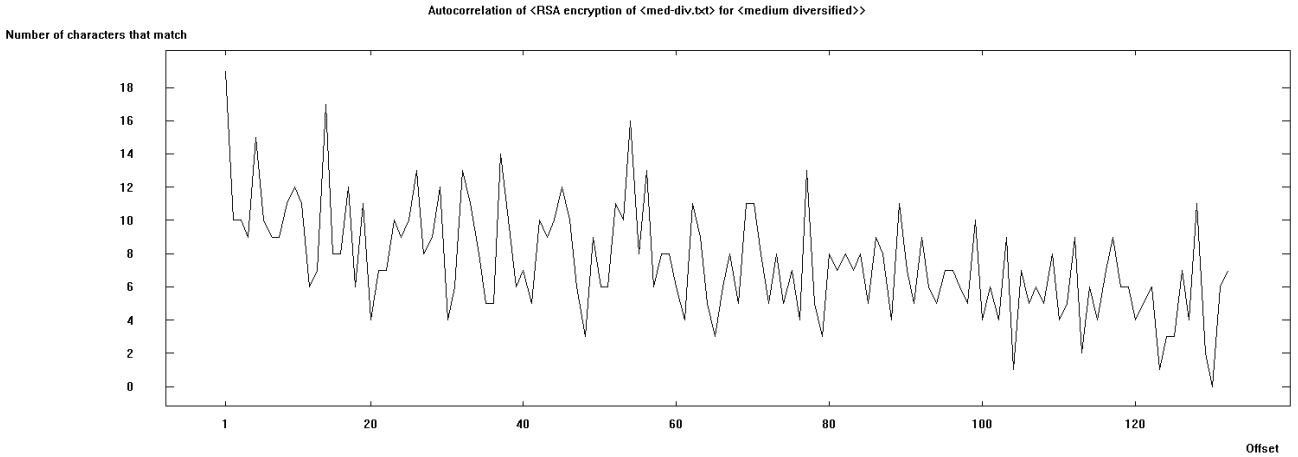
> mid-div, length 512;



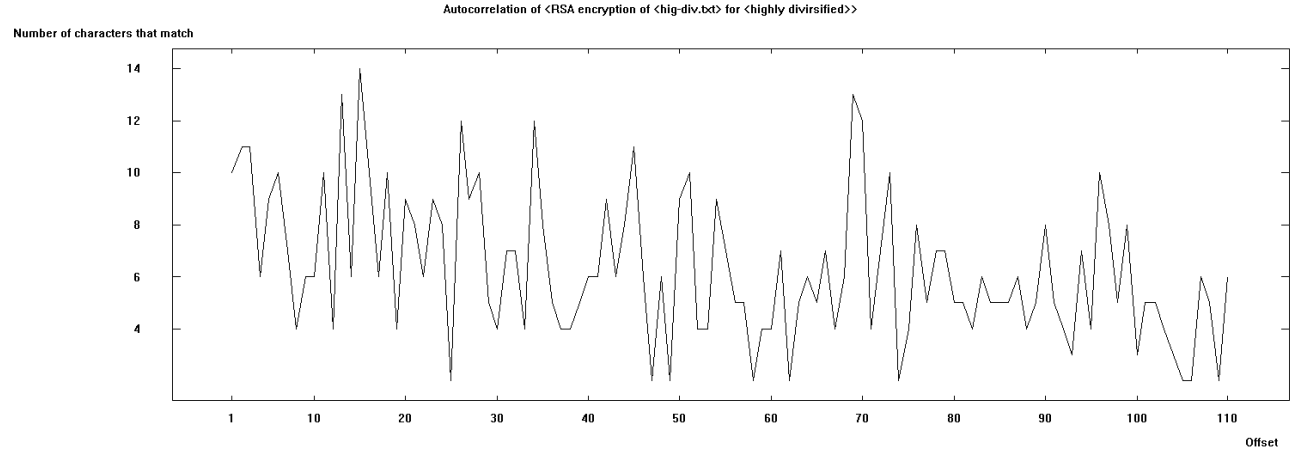
> mid-div, length 1024;



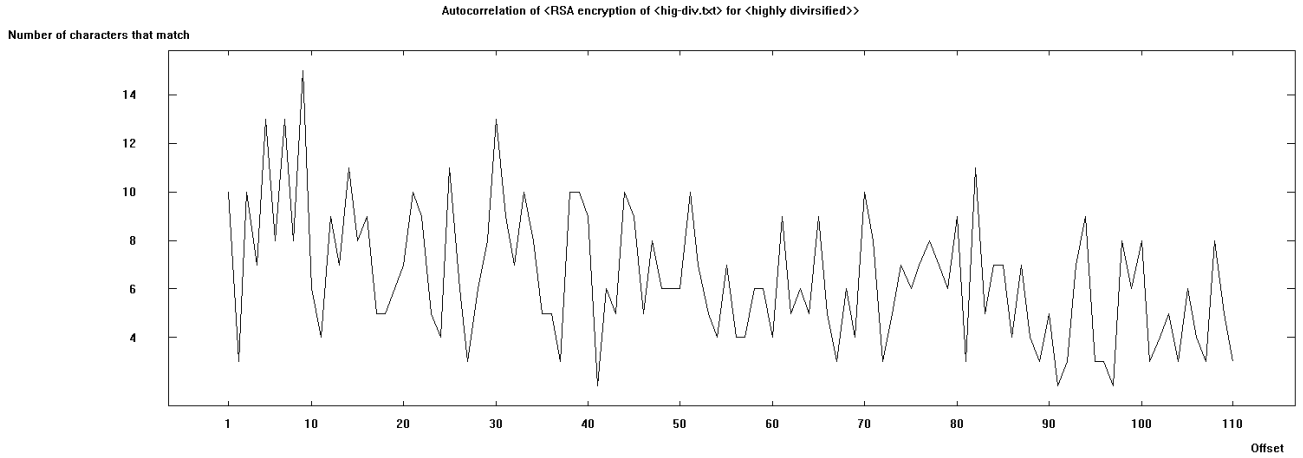
> mid-div, length 2048;



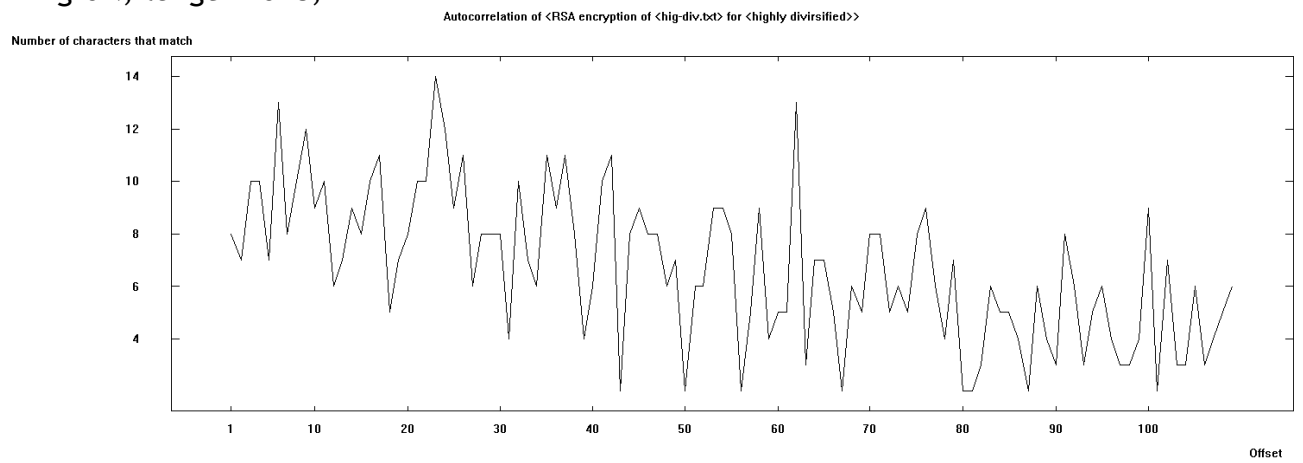
> hig-div, length 512;



> hig-div, length 1024;



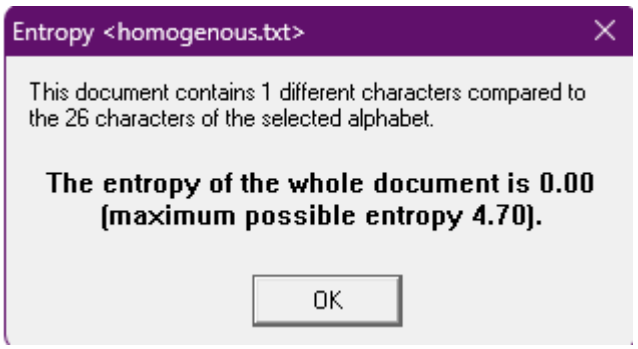
> hig-div, length 2048;



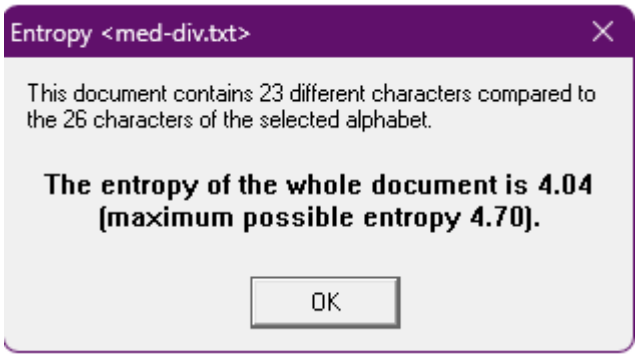
Mentioned entropy and autocorrelation are from ciphertext where we compare with plaintexts mentioned below;

entropy (plaintexts);

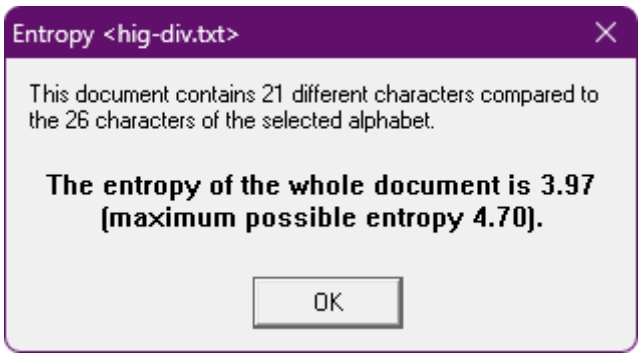
Homogenous,



Mid-div,

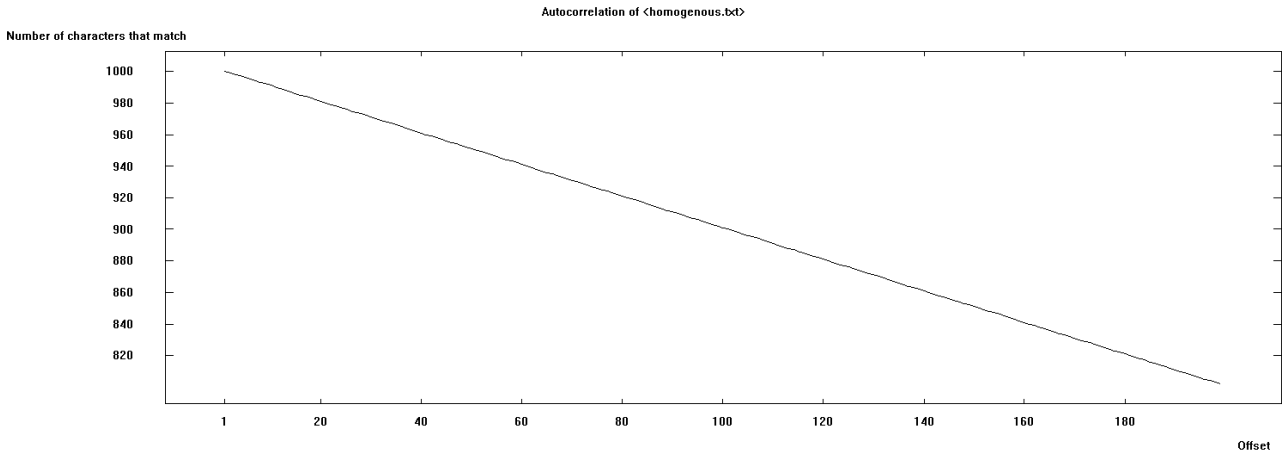


Hig-div,

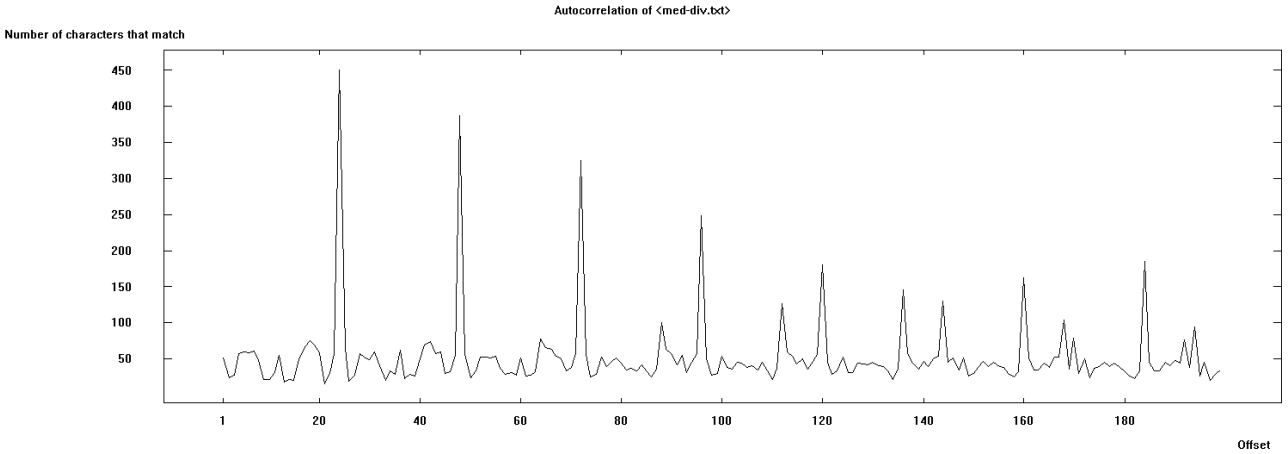


autocorrelation (plaintexts);

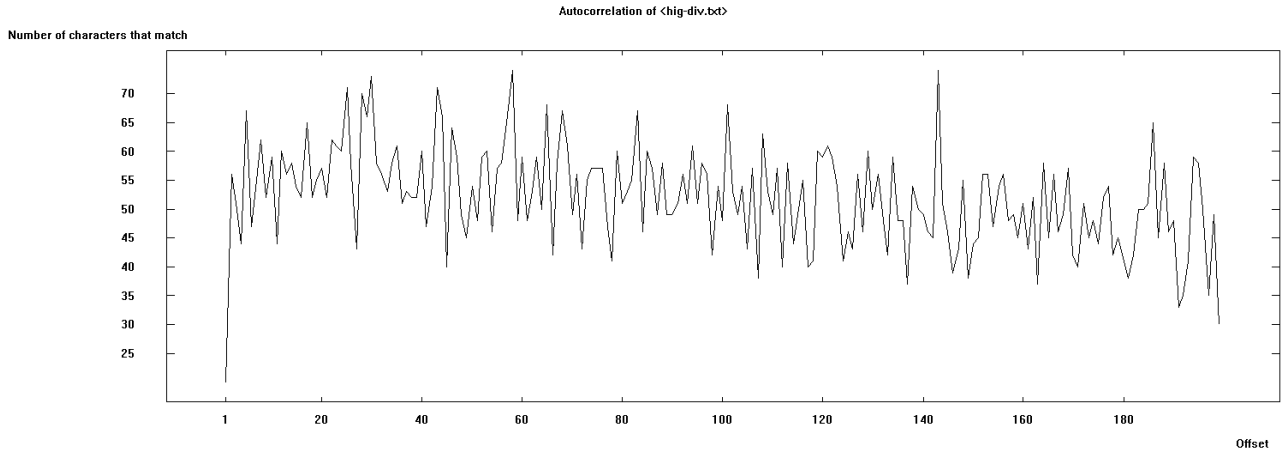
Homogenous,



Mid-div,



Hig-div,



5. For different RSA algorithm key lengths, measure the encryption and decryption time for files of 1MB, 2MB, 5MB (with an accuracy of 2 seconds).

Encryption time	1MB	2MB	5MB
512-bits	0.293s	0.577s	1.423s
1024-bits	0.497s	0.991s	2.444s
2048-bits	0.954s	1.850s	4.577s

Decryption time	1MB	2MB	5MB
512-bits	4.360s	8.789s	21.939s
1024-bits	11.454s	22.891s	58.457s
2048-bits	37.109s	74.126s	187.068s

6. Encrypt and decrypt the plain texts from point 4 using a symmetric algorithm, try to notice the time of these operations.

- **It was <1 sec.**

7. Introduce the following changes to the ciphertext (use a different length of a key). Then, decrypt the ciphertexts and check data integrity:

   a. change the value of 1 bit,

   b. change a few bits close together,

   c. change a few bits away from each other,

   d. remove 1 byte,

   e. delete several (several dozen) bytes,

   f. delete a fragment equal to the length of the algorithm module (512,1024,...).

> **For changing the values accordingly the decrypted text varies from the original, as in some cases it shows totally random text(s)/strings.**

II. Questions:

1. Does the key length affect the ciphertext entropy? If so, how?

- **It is directly correlated with key length as the higher it gets the more the entropy.**

2. Does the length of the key affect the autocorrelation of the ciphertext? If so, how?

- **For longer key lengths - autocorrelation is more complex.**

3. Does the ciphertext entropy depend on the plaintext entropy? If so, how?

- **The higher the entropy of the plaintext - the higher the entropy of ciphertext.**

4. How does the encryption/decryption time depend on the file length?

- **Larger the file size, the more time it consumes to process.**

5. What is the time of encryption/decryption with asymmetric algorithms compared to the execution of these operations with a symmetric algorithm?

- **Time for encryptions is similar yet time to decrypt us longer in asymmetric algorithms.**

6. What do the changes in the plaintext look like when you introduce the ciphertext falsification? What is their scale? What does it depend on?

- **Crashed in multiple places with 1 byte scale. It depends on the number of bits changed & key length.**

7. Is it possible to remove a fragment from the ciphertext so that the remaining text of the plaintext after decryption is readable?

- **The result would be unreadable.**

8. What are the advantages and disadvantages of asymmetric algorithms compared to symmetric ones?

- **Asymmetric algorithms compared to symmetric algorithms are more secure &**

take longer to decipher.

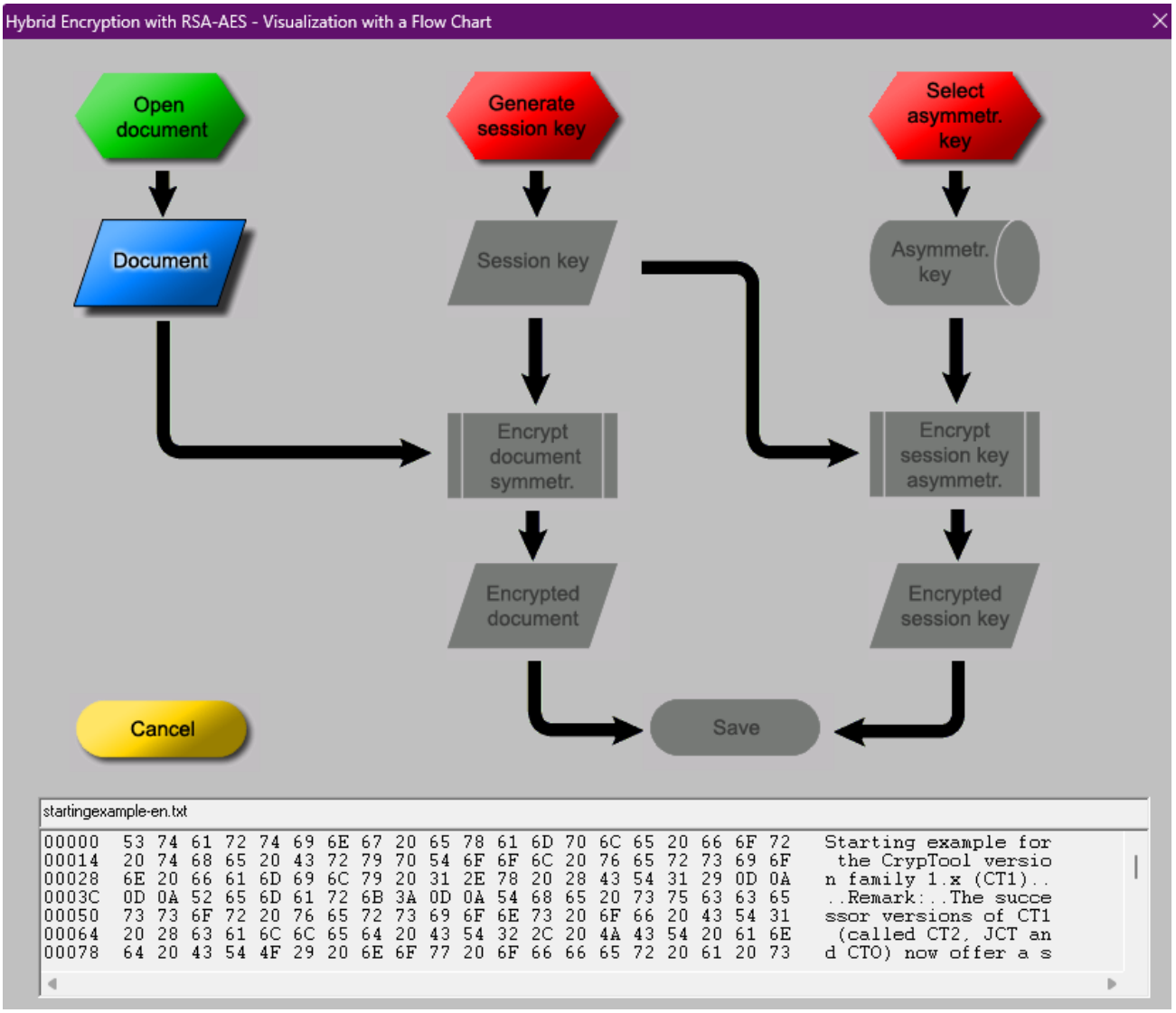
9. In which applications is it better to use asymmetric algorithms, and in which symmetric algorithms?

- **Symmetric Algorithms are good for lot of data processing applications for example payment applications & validations where**
  With more secure appliances it is better to use Asymmetric algorithms for digital signatures.

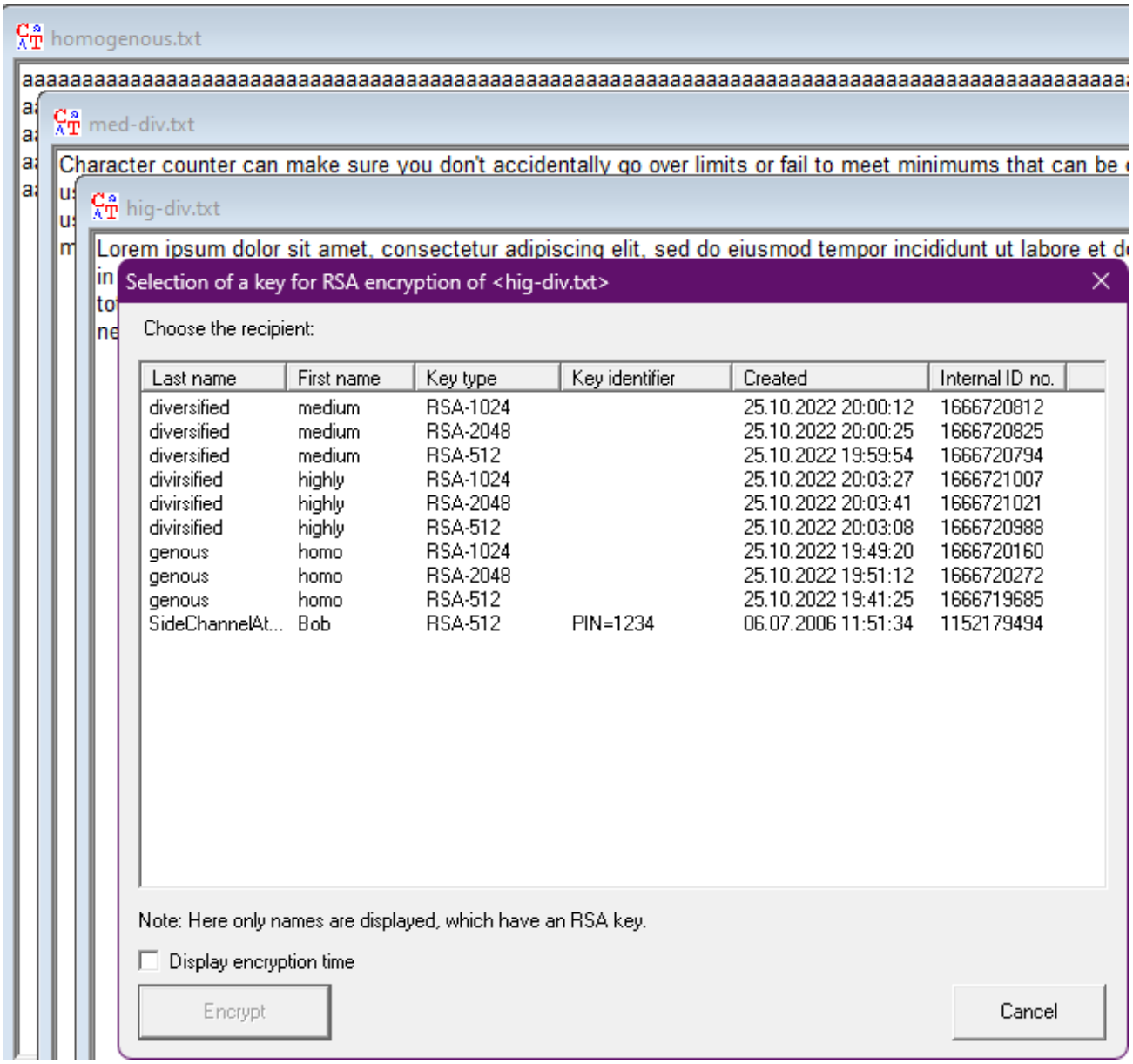
3. Properties of RSA-AES encryption.

I. Tasks:

1. Go to the encryption option in Cryptool - Encrypt-Decrypt/Hybrid/RSA-AES. See the flow of plaintext encryption using a combined symmetric (AES) and asymmetric (RSA) code. Pay attention to how a plaintext is encrypted and how the symmetric cipher key is transmitted.



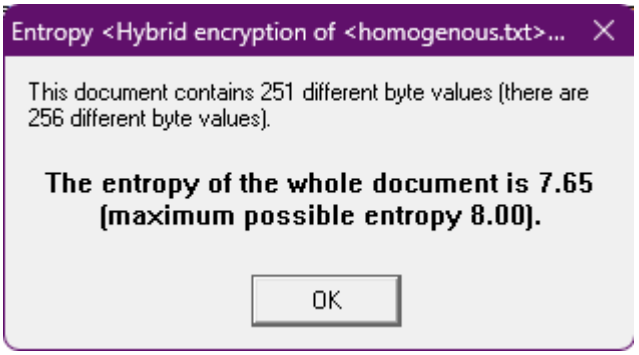
2. Use previously created RSA keys (512, 1024, and 2048 bits).



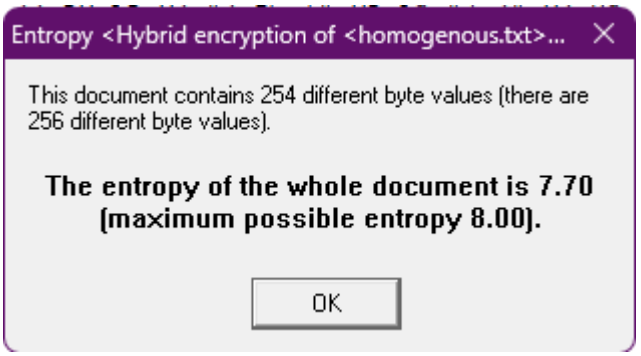
3. Encrypt the three plain texts (with different entropy) used in section 2 using a hybrid RSA-AES algorithm. Compare the entropy of plain text with the entropy after encryption (encryption entropy). Compare also the entropy after RSA-AES encryption and RSA encryption conducted in section 2. Check and compare the autocorrelation of the ciphertext (RSA-AES) and ciphertext (RSA). Note: use the same RSA key as in section 2.

_entropy :

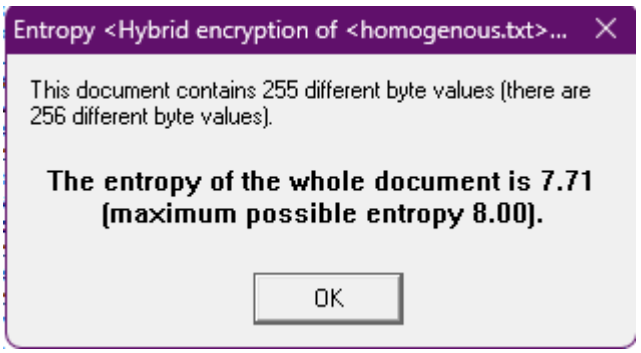
> homogenous, length 512 ;



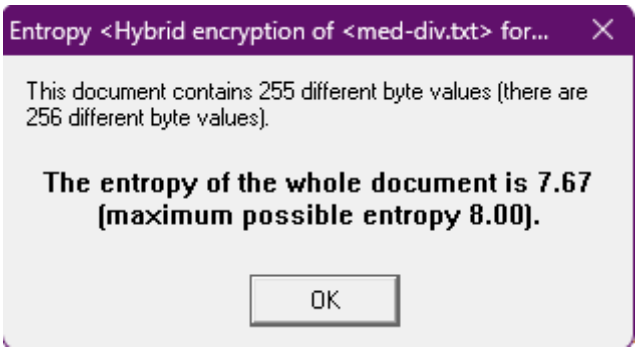
> homogenous, length 1024;



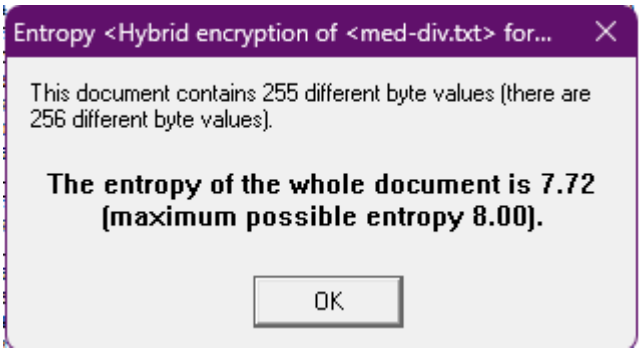
> homogenous, length 2048;



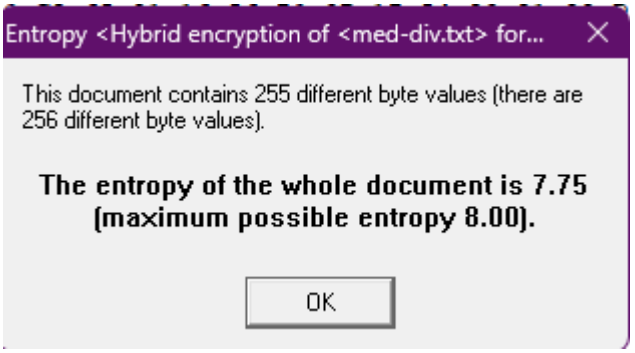
> mid-div, length 512;



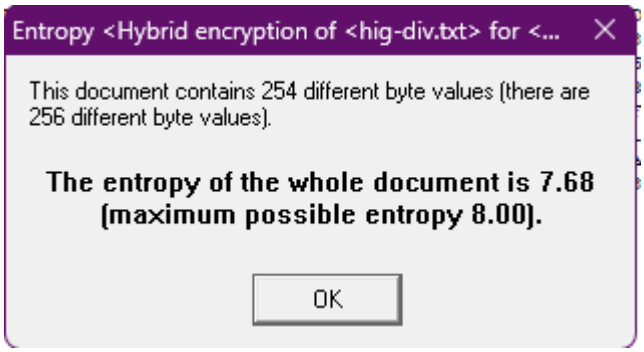
> mid-div, length 1024;



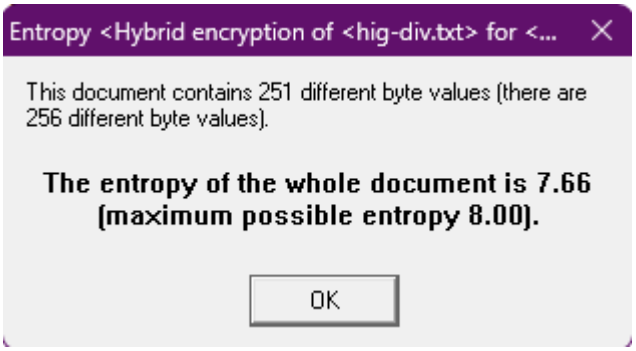
> mid-div, length 2048;



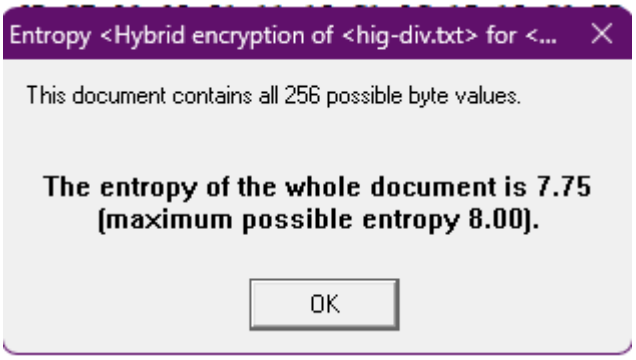
> hig-div, length 512;



> hig-div, length 1024;

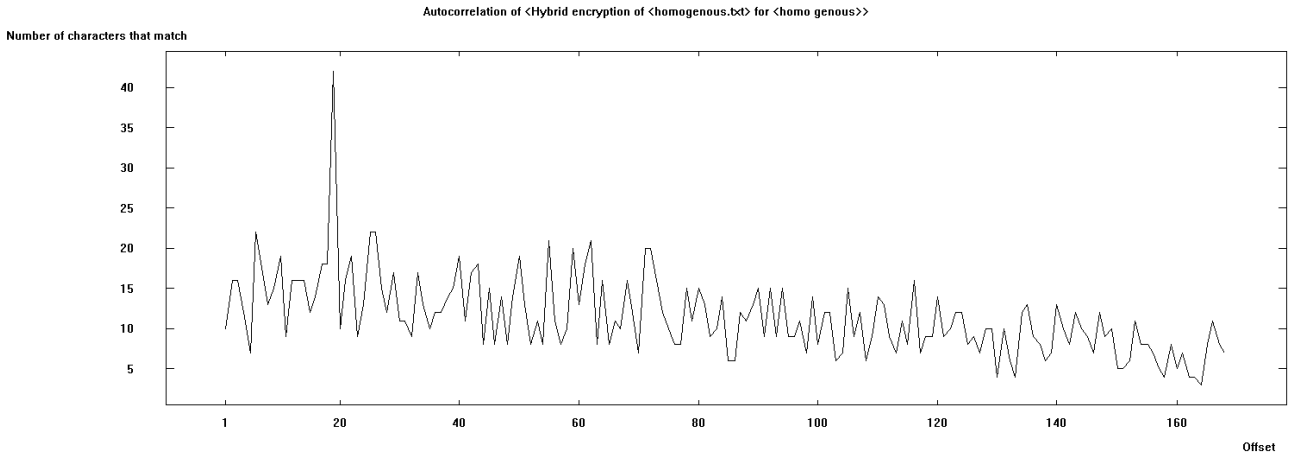


> hig-div, length 2048;

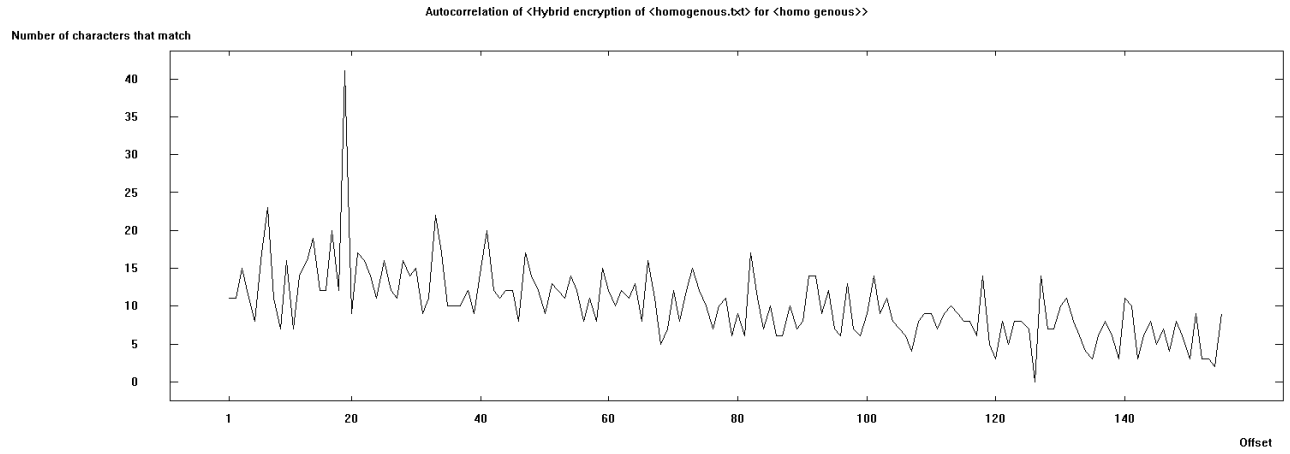


_Autocorrelation :

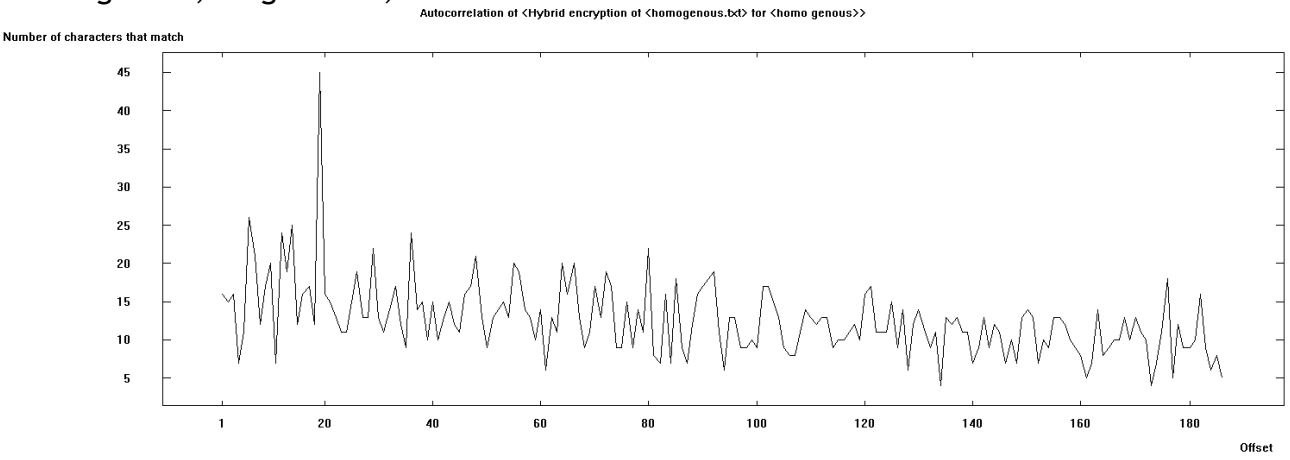
> homogenous, length 512;



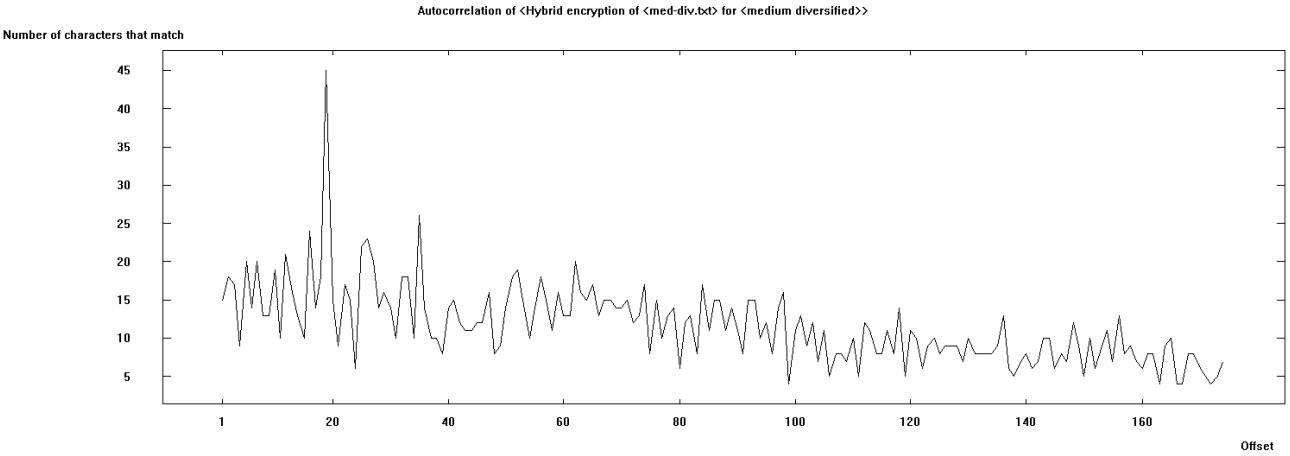
> homogenous, length 1024;



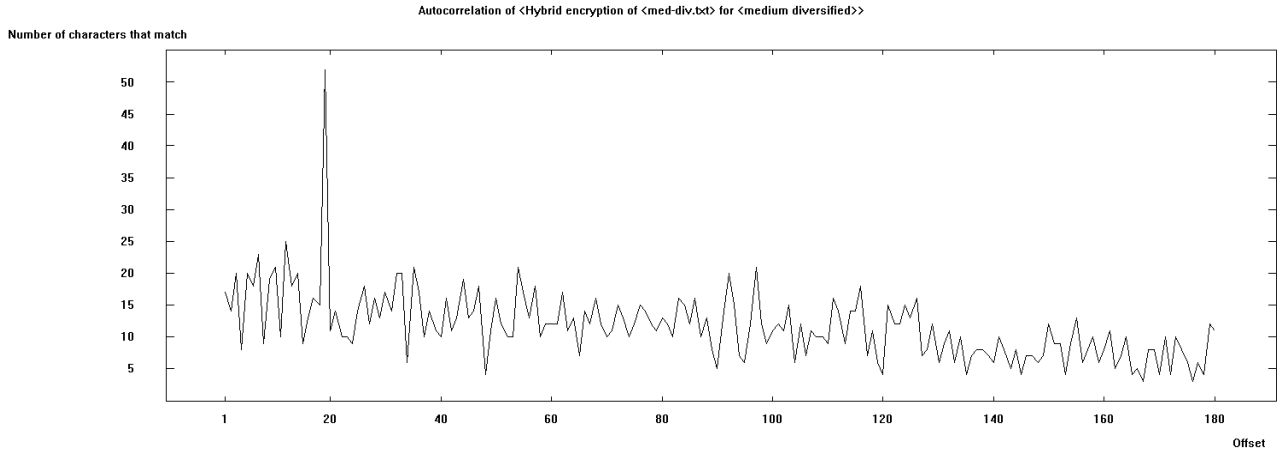
> homogenous, length 2048;



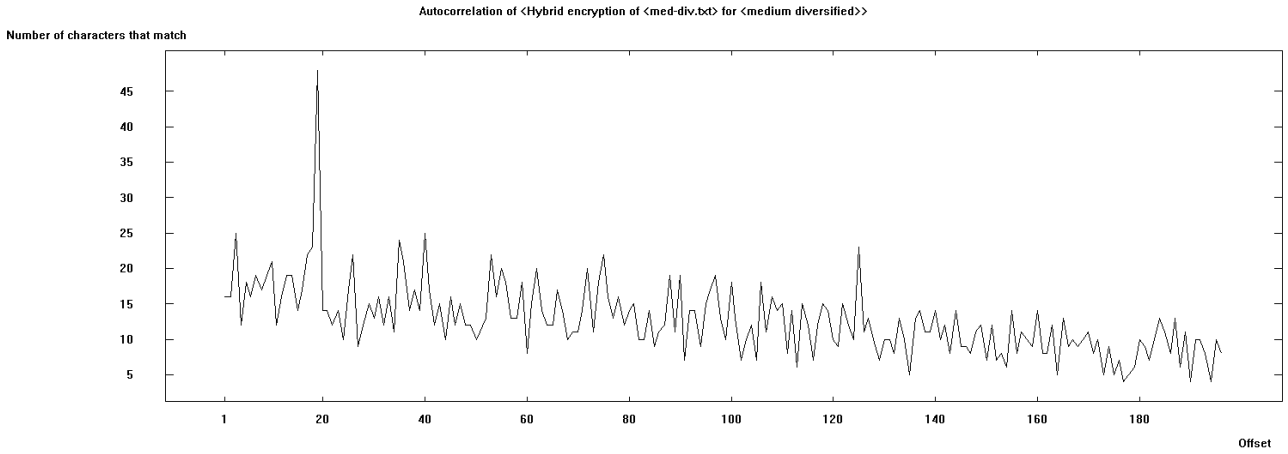
> mid-div, length 512;



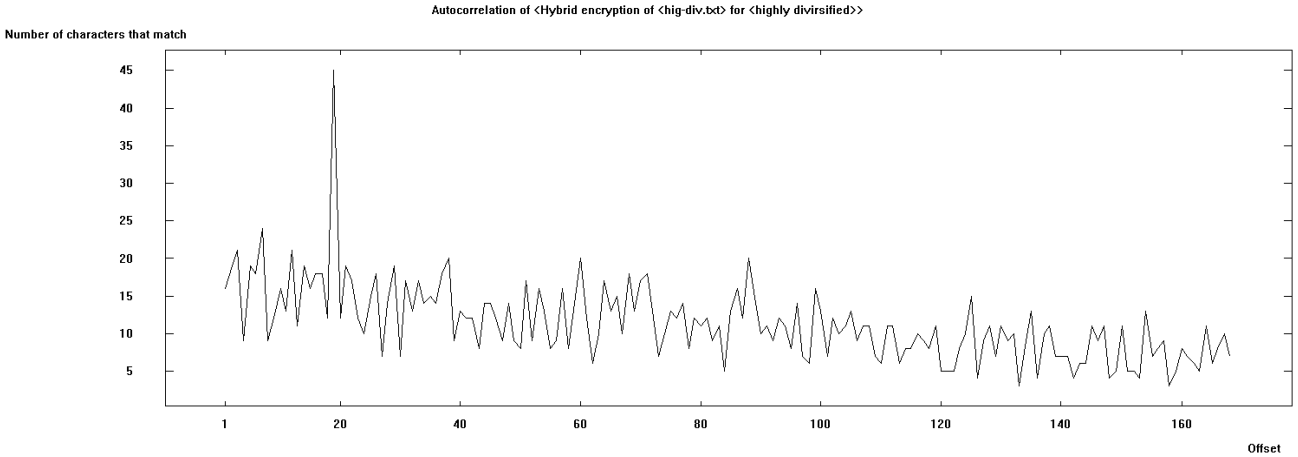
> mid-div, length 1024;



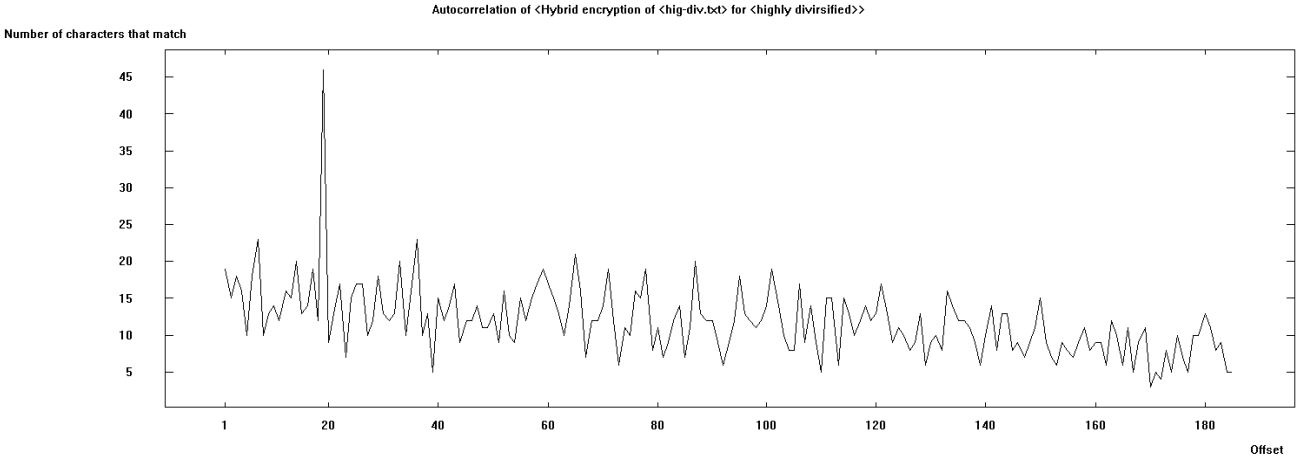
> mid-div, length 2048;



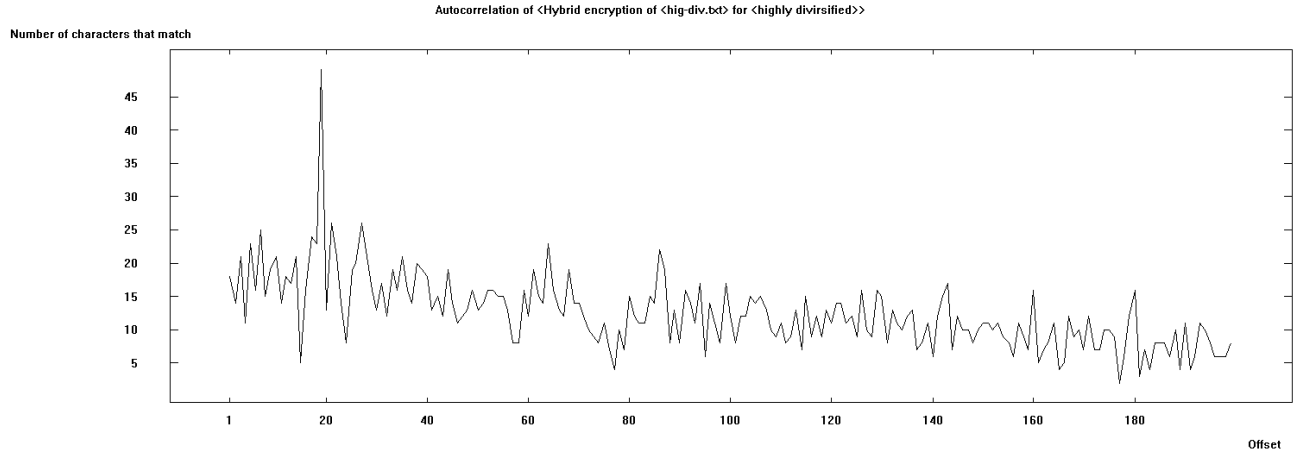
> hig-div, length 512;



> hig-div, length 1024;



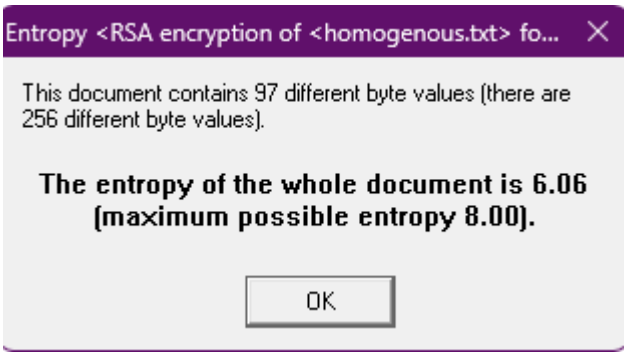
> hig-div, length 2048;



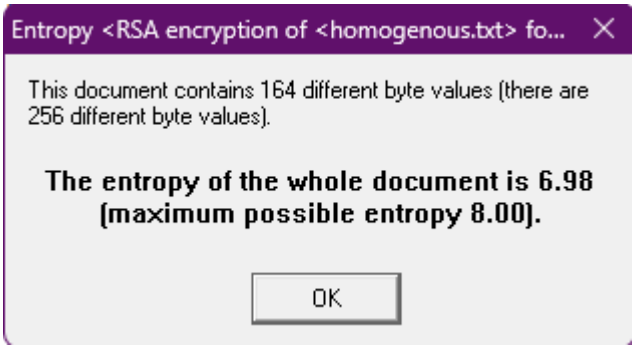
Mentioned entropy and autocorrelation are from ciphertext (RSA-AES) where we compare with ciphertexts(RSA) mentioned below;

_entropy :

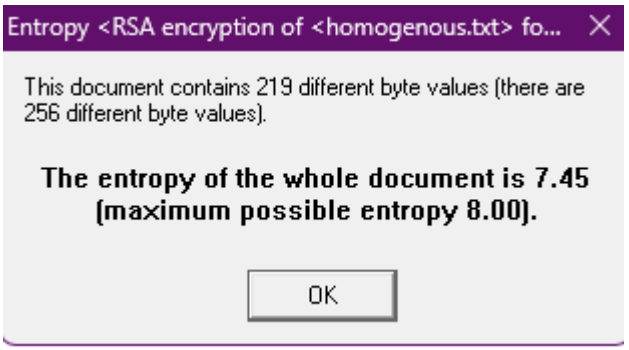
> homogenous, length 512 ;



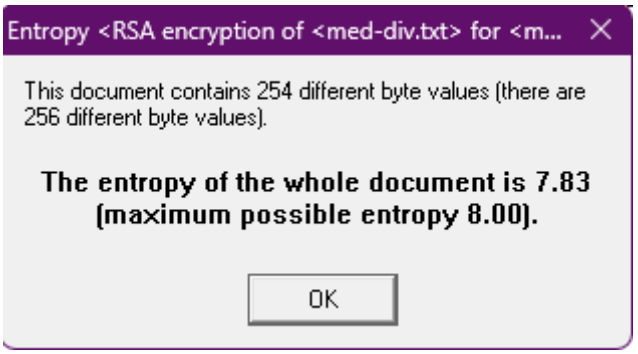
> homogenous, length 1024;



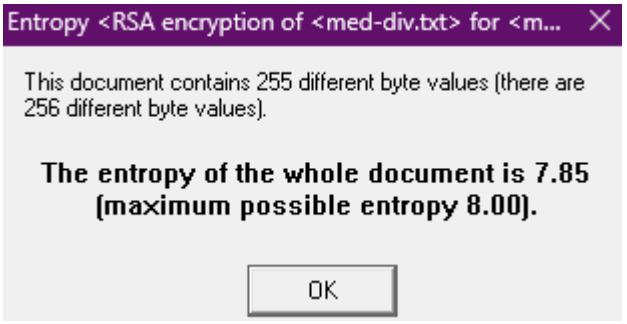
> homogenous, length 2048;



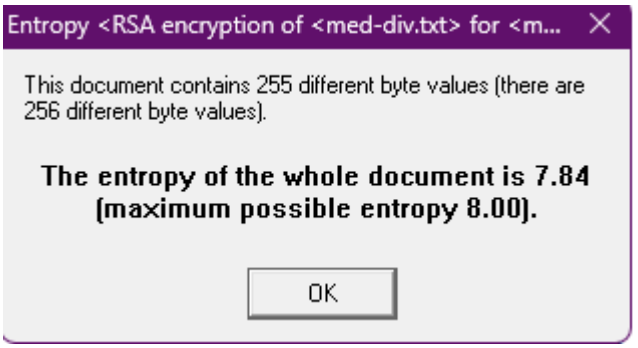
> mid-div, length 512;



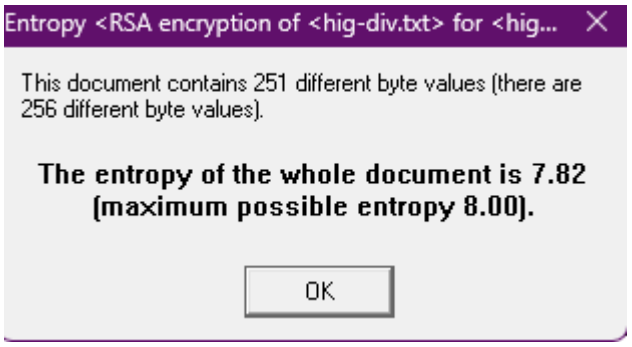
> mid-div, length 1024;



> mid-div, length 2048;

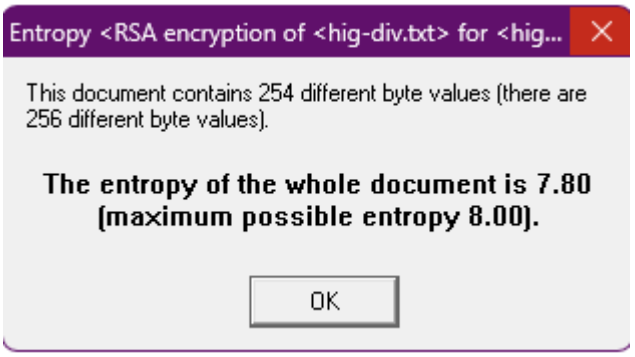


> hig-div, length 512;



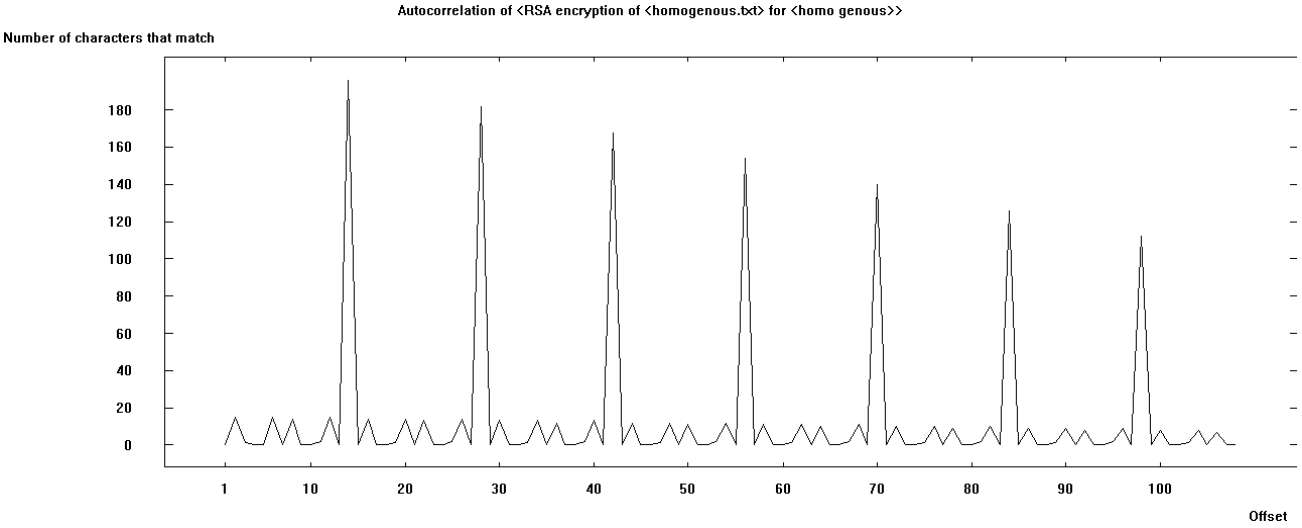
> hig-div, length 1024;

> hig-div, length 2048;

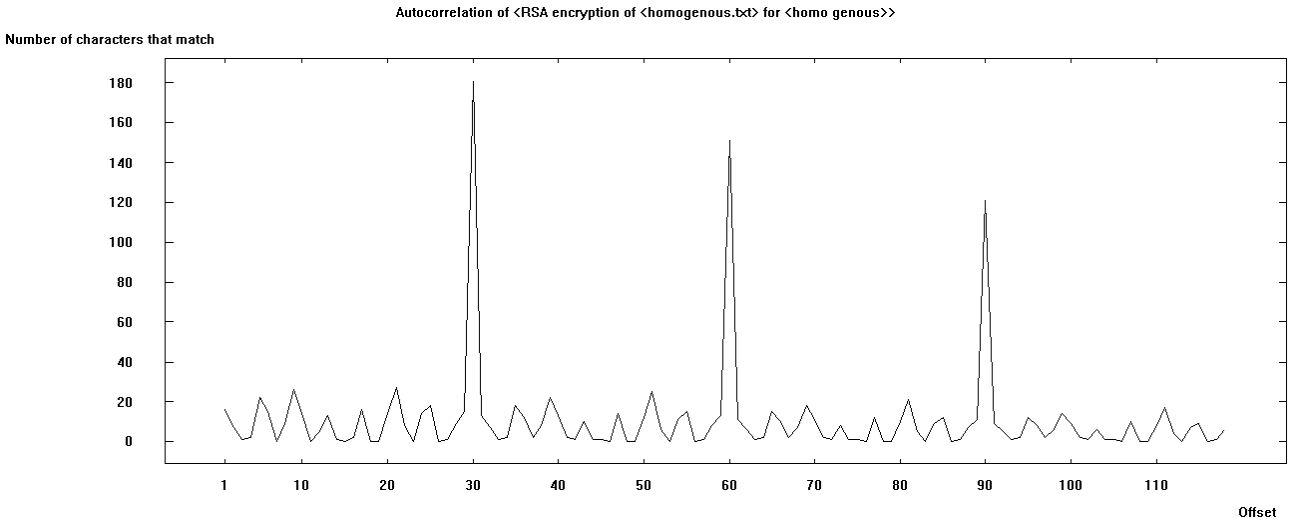


_Autocorrelation :

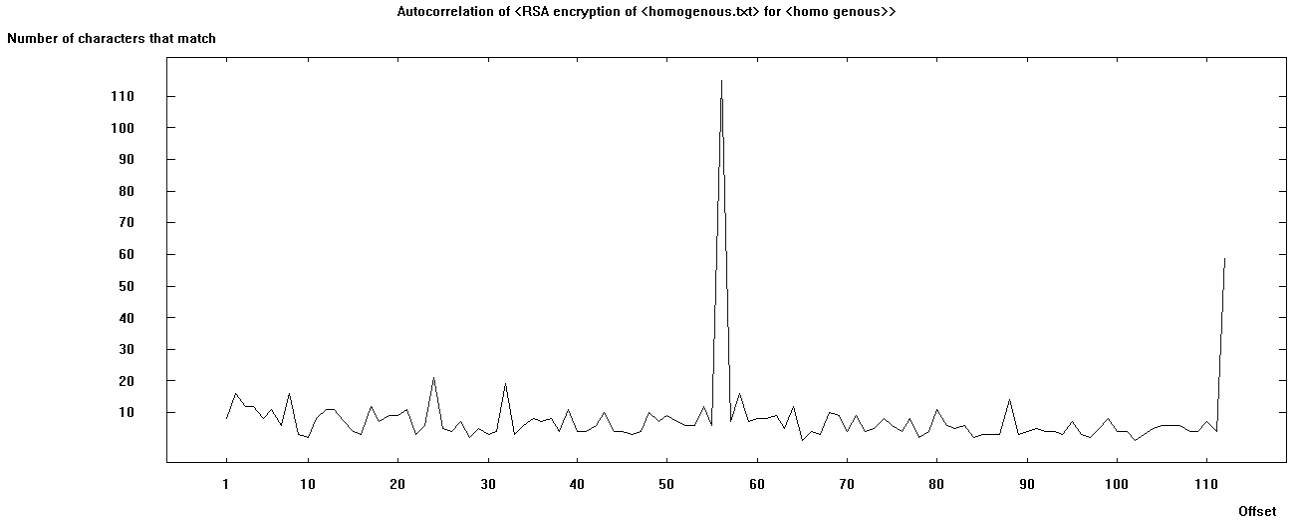
> homogenous, length 512;



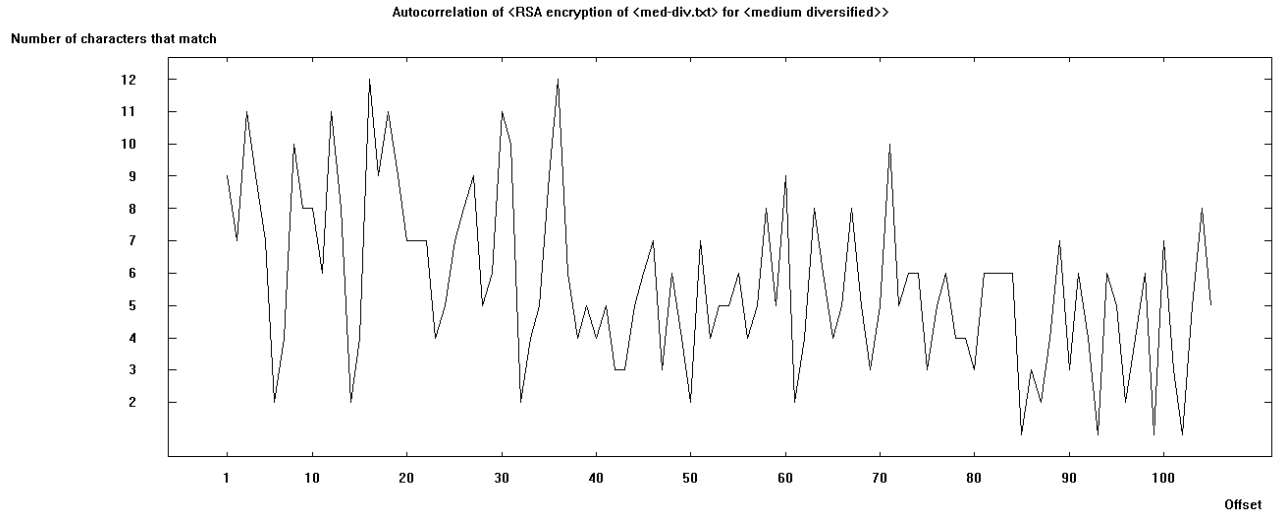
> homogenous, length 1024;



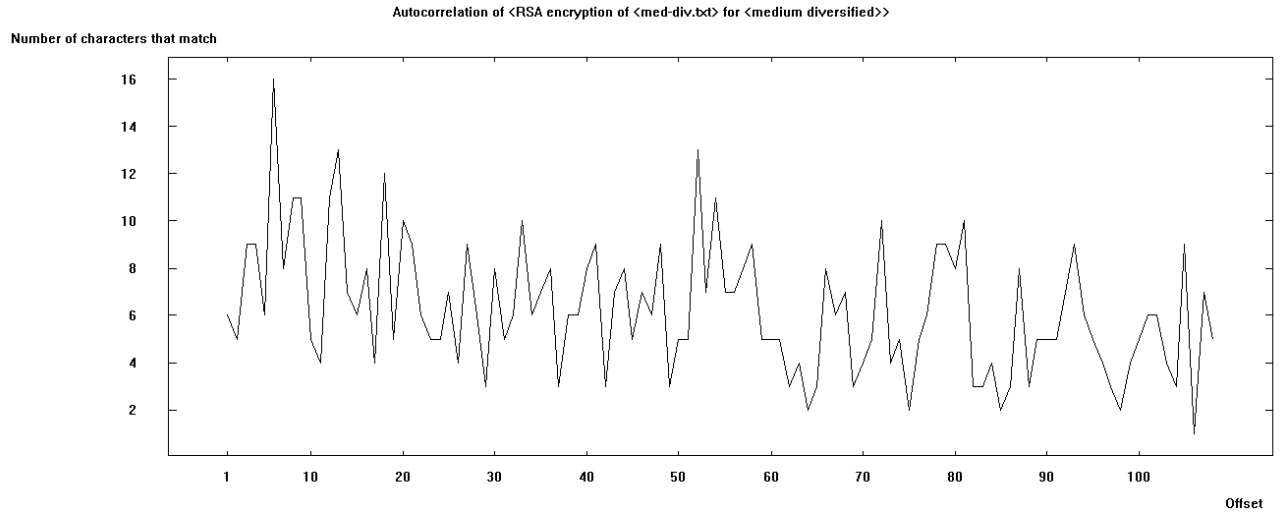
> homogenous, length 2048;



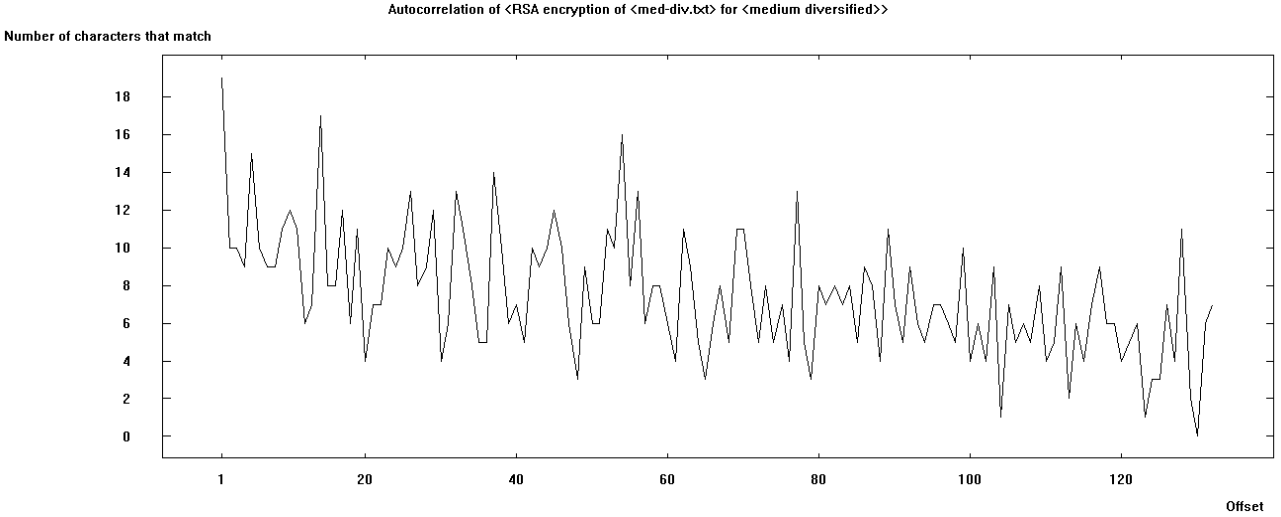
> mid-div, length 512;



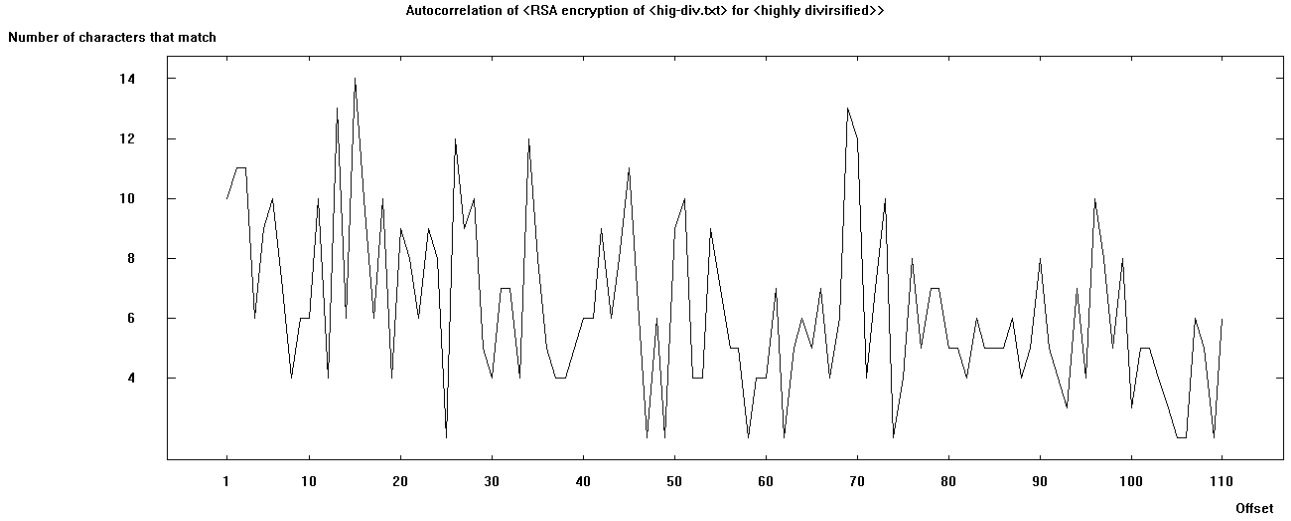
> mid-div, length 1024;



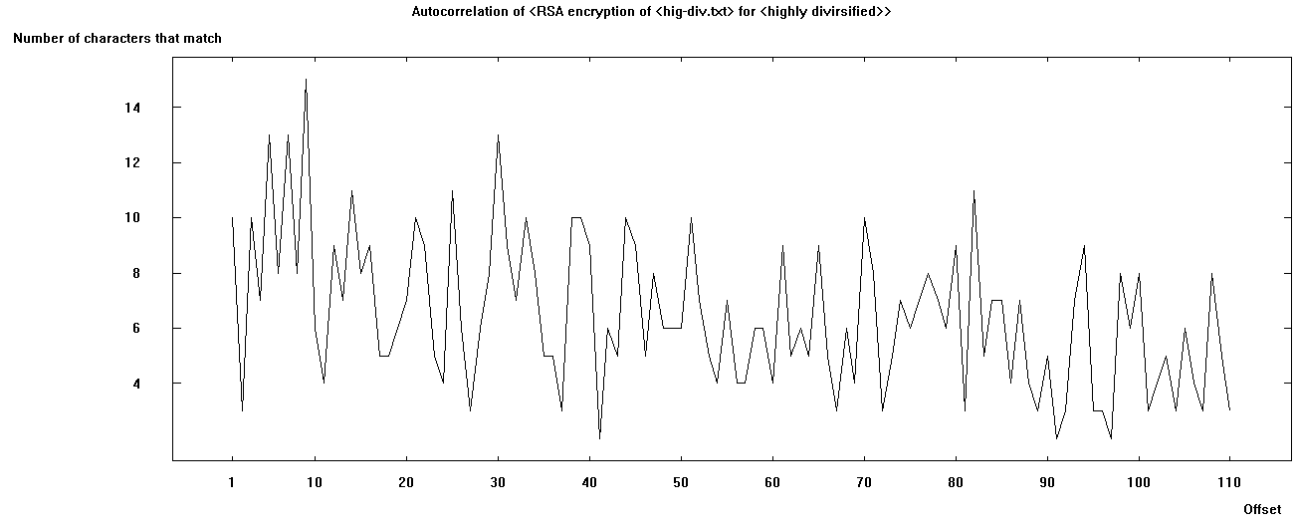
> mid-div, length 2048;



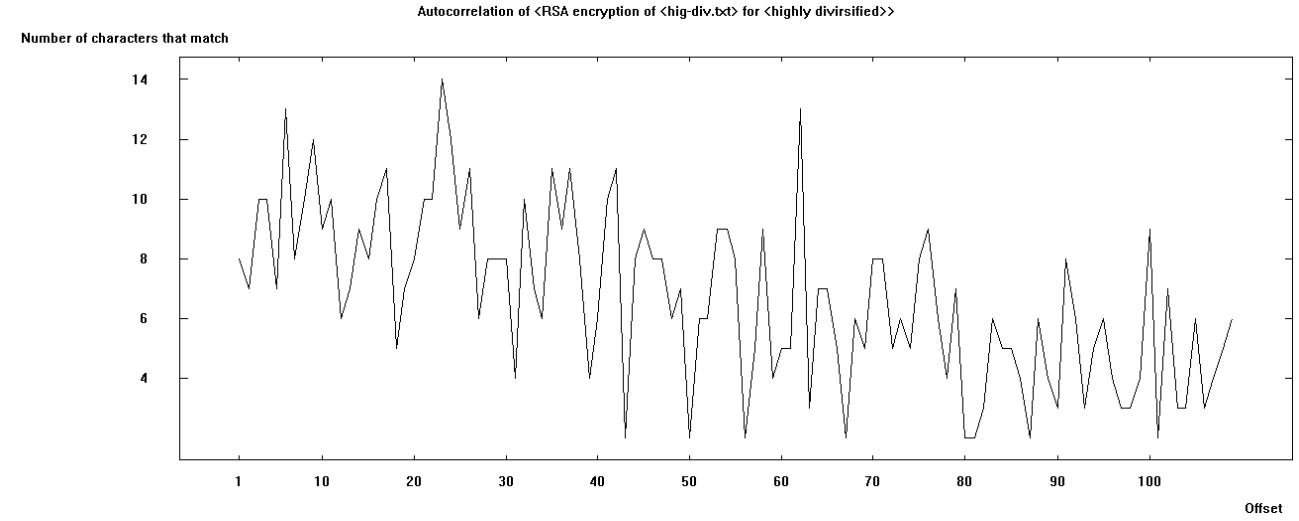
> hig-div, length 512;



> hig-div, length 1024;



> hig-div, length 2048;



Looking at the entropy values and Histograms we can clearly see the difference between changes.

II. Questions:

1. What is the purpose of the RSA-AES algorithm using a symmetrical and an asymmetrical cipher?

- AES is a symmetric encryption algorithm - one key can be used to encrypt, and then decrypt the message. You need to securely share that key with the system you're exchanging encrypted data with, otherwise other people can decrypt your data, or pretend to create encrypted data on your behalf.RSA is an asymmetric encryption algorithm - a pair of keys is used, one you keep to yourself (private), and one you share with the rest of the world (public).If you want to exchange data with RSA, you use the other party's public key to encrypt the data, which can only then be decrypted with the private key that only they have.The two algorithms are used for different purposes - RSA is really helpful for key exchange but it's slow to use.AES is really fast, but suffers from the security risks of key exchange (which can be solved using RSA)

2. Does the key length affect the ciphertext entropy? If so, how?

- **Yes, the bit complexity increases but the change is minor. It can be more or less.**

3. Does the length of the key affect the autocorrelation of the ciphertext? If so, how?

- **Yes, although the change is hardly eye-catching.**

4. Does the ciphertext entropy depend on the plaintext entropy? If so, how?

- **Yes & not by much compared to the RSA algorithm.**

5. What can be observed comparing the entropy of the ciphertext from section 2 (after RSA encryption) with the ciphertext from this point (after RSA-AES encryption)?

- **There are differences - not by a lot.**

6. What should the application of RSA-AES hybrid or related encryption be?

- **RSA-AES has both the advantages of symmetric and asymmetric algorithms; it is best used in mobile chat systems and SSL/TLS.**