DNA-BASED SECURE DATA HIDING SYSTEM

Ensuring Confidentiality, Integrity, and Authenticity

2024JCS2040

Introduction

DNA (Deoxyribonucleic Acid) is the long molecule that stores genetic information in living organisms, using a four-letter alphabet: A(Adenine), T (Thymine), G (Guanine), and C (Cytosine) (i.e.; four different nitrogenous bases). Generally, we store the genetic DNA of organism's digitally for different purposes like genomic research, medical diagnostics, conservation of endangered species, and also for forensic purpose (to identify individuals). For example, a short DNA sequence is "ATCGGACT". So, we are using the DNA as a medium to store secret messages in it.

As we know that Steganography is the art of hiding a secret message inside something (like image, audio, etc.). It is very hard to detect the existence of the message because it looks normal. Also, cryptography is the art of encrypting a secret message with a key to make it unreadable. It ensures that even if someone finds the message, they can't read it without the key. Therefore, we are using these two to make the message both invisible and secure. The medium we are using here is organism's DNA, because if we store the secret message inside DNA which will visible to everyone as a normal DNA only. If someone try to detect the existence of the message also it will be very hard because the DNA consists of approximately 3-billion nucleotides bases in it for each organism.

DNA-Based Secure Messaging System

Cryptography techniques will give confidentiality, integrity, non-repudiation and authentication to the system. The techniques are encryption, error correction and digital signatures. For encrypting a message. it is better use symmetric key algorithm than asymmetric key algorithm because asymmetric method will take more time to encrypt and decrypt. We can share the symmetric key using asymmetric key which makes faster to encrypt and decrypt the message. Error correction will fixes mistakes in the DNA sequence to ensure the message isn't corrupted (or tampered by attackers) or lost. Digital signatures gives the proof of the sender of the message who is genuine or fake sender and also proves the message hasn't been tampered. Since, we are hiding the secret message in DNA and storing in the data storage like cloud we are using these cryptographic techniques.

In our system, we are using Steganography methods to make it hidden by mapping binary message to DNA characters(bases)without any noticeable changes in it.

Used Methodologies

Compression

Compression is used to reduce the size of the data to save storage space or for faster data transmission. There are two main kinds of compression: lossless and lossy. Since we don't need any loss in the data, we are using lossless compression technique only which is Huffman coding. In this it assigns shorter codes to more frequent symbols and longer codes to less frequent symbols. Therefore, it counts each character appearance, builds a tree in a way that frequent characters will get short binary codes then replaces each character with its code that creates a shorter binary string data. The mapping of each character is stored separately in a codebook which is used at the time decompression.

Encryption

Encryption is like putting a secret message in a locked box that only opens with a special key. It scrambles the message so it looks like a random gibberish to anyone without the key. Here we are using AES (Advanced Encryption Standard) with a 256- bit key to protect the message before hiding it in a DNA sequence. It works on fixed size of data called blocks (256 bits or 32 bytes each). In our system we are implementing two kinds of modes: CBC and GCM. These modes decides how the blocks are handled to encrypt and decrypt the message securely.

In CBC mode, the message is split into blocks (256-bit) and then the first block is XORed with the IV (initialization Vector-256 bit) and the result is encrypted with AES using the 256-bit key, gives cipher block. For next blocks, the block is XORed with the previous encrypted block and the result is encrypted with AES to give the next cipher block. In this way each block is linked with previous block and the IV and the AES key need to stored.

In GCM mode, the message is encrypted like a stream (not like blocks chaining) and adds a tag to verify the message isn't changed. A random nonce(96-bit) like CBC's IV is generated to make the encryption unique. A counter value starts at a value derived from the nonce like nonce+1, nonce+2, … and each counter is encrypted with AES using 256-bit key, gives keystream bits. This keystream bits are XORed with the message to give ciphertext. In this way whole message is encrypted. For message integrity, a tag is generated by combining the ciphertext, nonce and a key derived value through Galois Field multiplication.

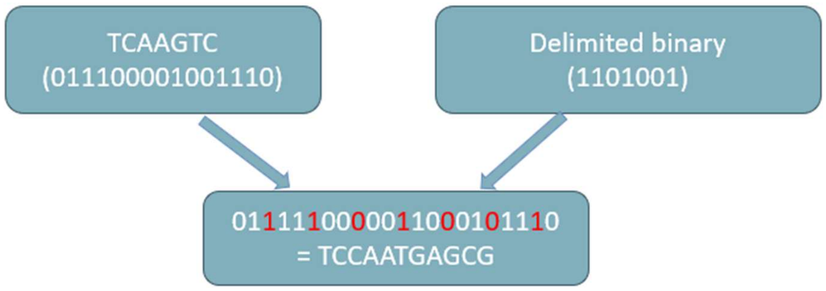
Error Correction

Error Correction is like having a backup plan to fix mistakes in the message so that the original message can be recovered. DNA sequences get errors during storage, transmission or sequencing. Error correction adds extra information to detect and fix these mistakes. Here, we are using two kinds of methods: Reed-Solomon and Hamming. For encrypted message we add error correcting bits. In Reed-Solomon, we add these bits in the starting of the encrypted message to correct burst errors, these bits are calculated using some mathematical formulae. In Hamming code, the error correcting bits are added in between the message to correct single bit errors. For each 8-bit byte we add 4 parity bits which give 12-bit codeword. Therefore, after adding parity bits the message length will be increased. At the time of decryption, after correcting the errors, we remove the parity bits and then do decryption.

DNA Encoding Methods

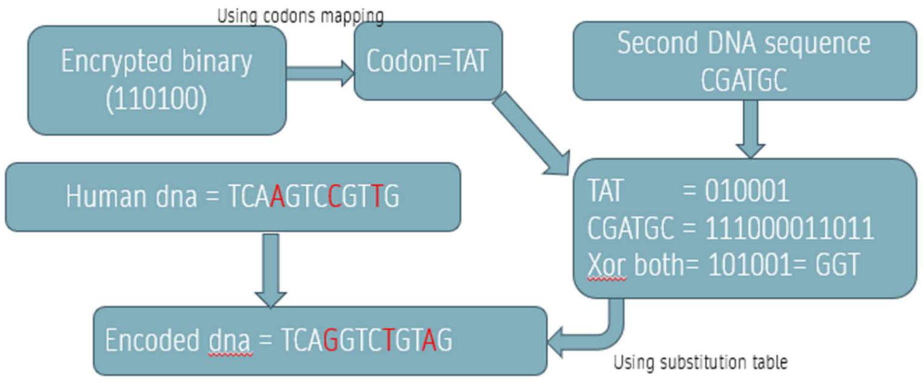
To embed this encrypted message into DNA sequence, we are using steganography technique. We are using two different approaches to embed the message into DNA sequence: Method1 and Method2. Before these methods we add delimiters starting and ending of this added parity message.

Method1



In this method, after adding delimiters we insert the each bit inside the DNA sequence depending on the key value. For example if the key value is 2 then we insert the message bit for every 2 bits in the DNA sequence. This will give us a binary string. We convert this into DNA sequences using DNA encoding. Now we get the encoded DNA. The length we is increased because we are inserting not replacing.

Mehtod2

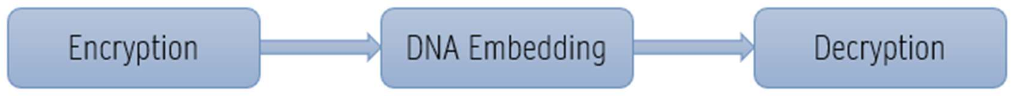


In this method, after adding the delimiters we convert these into codons(3 bases) for every 6-bits using codons mapping and then we convert it into bits using DNA decoding and do the XOR operation with second DNA sequence bits. These bits convert to DNA sequence using DNA encoding and then substitute each base in the DNA sequence (Human DNA) using the substitution table. Here we are replacing the contiguous repeated second base with other base using the substitution table. After this we get the encoded DNA. Here the length we remain same because we are replacing.

Digital Signatures

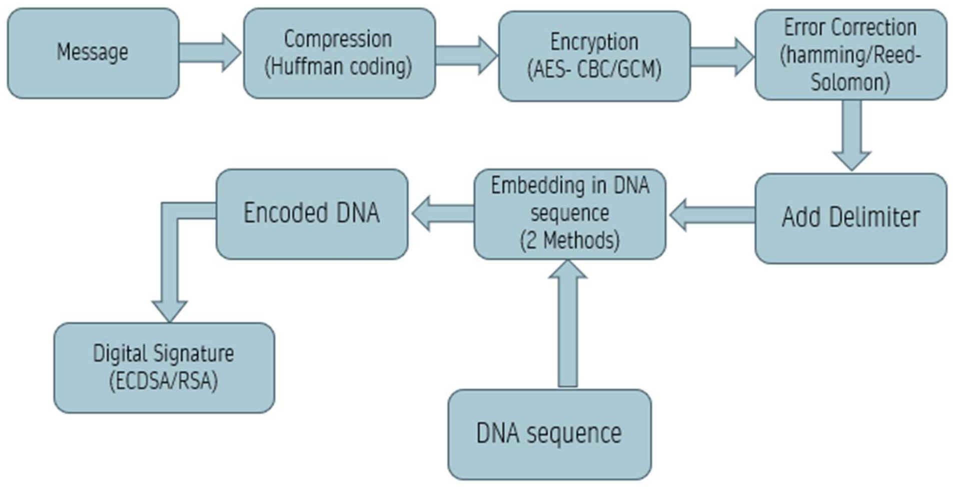
A digital signature is like a tamper-proof seal on a letter, proving that it comes from the right sender and hasn't opened or changed. It uses a private key to create the signature and a public key to verify. For this I used two signature types: RSA and ECDSA. Here we generate a public key and private key pair for choose signature type. The key for RSA is very large prime numbers that creates a 2048-bit signature. The key for ECDSA is based on the elliptic curve that creates a smaller ~512-bit signature. This ECDSA is faster compare to the RSA. First we will hash the encoded DNA using SHA-256 and then sign with the private key. In verification we use the public key. Here we are also embedding the hash of the signature in the DNA to ensure the DNA is authentic and unchanged.

Work flow



Starting we are doing the encrypting the message using encryption keys and then embedding into DNA (hiding in DNA sequence). Now from this DNA sequence, encrypted message is extracted and then do decryption.

Encryption


Message
Compression (Huffman coding)
Encryption (AES- CBC/GCM)
Error Correction (hamming/Reed-Solomon)
Add Delimiter
Embedding in DNA sequence (2 Methods)
DNA sequence
Encoded DNA
Digital Signature (ECDSA/RSA)

We can say that this is a multi-layered security combining cryptography, error correction and steganography.

Steps:

1) First, we compress the secret message using Huffman Compression which reduces the message size by 30-50%. Here we need to store the mapping of the characters.
2) Now, we encrypt the compressed message using AES-256 bit key in two different modes either CBC mode or GCM-mode. Using CBC-mode we get only encrypted message as output but using GCM mode we also get authentication tag for integrity.
3) For this encrypted message we add some extra parity bits for error correction. If we want burst error correction, then we use Reed-Solomon code otherwise for single error correction we use Hamming code.
4) Now for this we add delimiters of 7 Bytes length at starting and ending, to easily recognize the starting point and ending point of the encrypted message while decrypting.
5) We now embed this inside the DNA using any one of the two methods (method1 or method2). After embedding we get final encoded DNA. This encoded DNA we will store in the cloud.

6) Now for sender proof, we implement digital signature. First, find the hash of the encoded DNA using SHA-256 hashing method, then sign this hash with either ECDSA or RSA private key. Convert this signature into DNA bases and embed into the DNA sequence.
   Public Keys: Encoded DNA, signature
   Private Keys: AES key, codebook, embedding key

Decryption

Decryption is reverse of the encryption process and also here we do signature verification of the sender.

Steps:

1) Extracts the embedded signature from encoded DNA and verifies using the public key of the signed signature. If the signature is not verified then we stop the decryption process otherwise we continue.
2) We now extract the data from the encoded DNA doing reverse of method1 or method2 depending on the which we used for encryption.
3) Now we find the delimiters and then extract the encrypted data which is between the delimiters.
4) For the extracted encrypted data, if there are any error we correct it using error correction decoding method depending on the which we used (Reed-Solomon/ Hamming).
5) After fixing the errors, we decrypt the encrypted message using AES key in either CBC mode or GCM mode decryption which we used.
6) Now decompress this decrypted message using the codebook to rebuild the original message.

Results

Generally, Human DNA sequence is of 3-billion characters (bases-A, T, G, C which is of 2bits each) and it very fare enough to embed a huge amount of information in it. But practically to use that 3-billion characters is very hard for evaluation. So, I am using 15000 characters (30000-bits) DNA sequence and the DNA encoding used is A=00, T=01, G=10 and C=11. The secret message I used is having 614 characters (in ASCII, 8bits per character => 4912bits). After compression it reduces to ~50% (that is ~ 2560 bits). For in method2 second DNA sequence is shorter only (up to 3500 bits).

After encryption and error correction the bits will increased with IV or nonce, tag and padding in CBC mode and also parity bits, delimiters (the final output is ~3000 bits). These bits are embedded in the DNA sequence bits using method1 or method2. The encoded DNA sequence bits length will remain same for method2 but increased in method1 (that is same 15000 bases for method2 and greater then 15000 bases for method1.

Evaluation Factors and Security analysis

The quality of a DNA steganography technique depends on several factors like cracking probability, layer of security, blindness, modification rate, preserving original DNA functionality, hiding capacity, speediness, etc

1) Cracking probability is very less if we are taking the long DNA sequence with more than 1-billion bases in it. Also we used AES 256-bit key encryption, the probability to crack this key is 1/(2^256) which is very hard to crack.
2) Layer of security means number of DNA sequences used in data embedding methods. In method2 it is more secured than method1 because we are using double layer.
3) Blindness means sender need not to send the receiver original DNA sequence. In this, for method1 has more blindness compare to method2 because in method2 the receiver need to use the original DNA sequence while decrypting the secret message.
4) Modification rate means the percentage of the original DNA modified or altered. In both the methods there is less modification rate only. Because when we are using the long DNA sequences, the modification rate very much less.
5) Payload for method1 is more compared to method2. Method1 increases the length of the DNA sequence but in method2 it remains same.
6) Preserving original DNA functionality is that function of producing proteins is not affected. In our system we are preserving it.
7) Hiding capacity for method1 has more than method2. For method2 we can only hide limited data depending on the repeated number of bases in the sequence. If the percentage of repeated number of bases in the DNA sequence is 2% only, then we can hide small amount of data.
8) Speediness means the system is faster or not.
9) Confidentiality, in our system if the attacker extracts the encrypted message from the DNA like by finding delimiters or guessing the index in method2, still it remains unreadable without the key.
10) Authentication and Integrity, Since we are using the digital signatures overall system is ensuring integrity and authenticity. If we no need for authentication also the integrity will be there with GCM mode.
11) Correctness, since we are using the error correction methods, the system will recovers the original message without any errors. But hamming code will only able to correct single bit errors. Reed-Solomon code will correct burst errors(I tested this).
12) Avalanche effect, this is depending on the ratio of DNA sequence bits and the message bits. If DNA sequence bits has very high percentage than message bits then the avalanche effect is very less (nearly below 5%). That is a slight change in the input, the results are significant and unpredictable in the output.

Challenges faced : It is very much time taking to test our algorithm with larger DNA sequence (1.6 billion). We need more computing power to do this test.

Comparative analysis

Factors	Method1	Method2
Cracking Probability	low (delimiters detectable, AES-256 secure)	Extremely low (multiple dependencies, AES-256 secure)
Security Layer	Single-layer (cover DNA)	Double-layer (cover + second DNA)
Blindness	Fully blind (only DNA and delimiters needed)	Partially blind (requires indices, table, second DNA)
Payload	Yes	No payload
Encrypting Secret Data	AES-256 (CBC/GCM, 2^256 attempts)	AES-256 (CBC/GCM, 2^256 attempts)
High Capacity	Nearly unlimited	Limited
Easy to Apply	Simpler, faster	Complex, slower

Conclusion

Our system provides excellent confidentiality, strong integrity, and high correctness for hiding a secret message in a DNA sequence. Method1 ensures high security and accuracy with simpler steganography and also Method2 offers superior confidentiality and integrity due to its double-layer, with equally high correctness. Both methods uses AES-256 encryption, RSA/ECDSA signatures, and Reed-Solomon/Hamming error correction to protect the message against unauthorized access, tampering, and errors. The system is well-suited for secure messaging, DNA data storage, and other applications. The 1.6 billion-base DNA sequence's vast size gives exceptionally low modification rates, enhancing secret and supporting high-capacity data hiding, while the small message size minimizes processing overhead. Also the system is highly sensitive to input changes, enhancing security by making it difficult for attackers to predict or reverse-engineer the output based on small input modifications.