Invariant Key-Point Fully Homomorphic Encryption (IKP FHE)

by

Jonathan R. Anderson

Abstract: Fully Homomorphic Encryption is a cryptographic technique which allows for arbitrary computations to be performed on encrypted data without requiring access to the underlying, unencrypted data. It is a type of stream cipher. To date, stream ciphers have only been used for data in transit with the goal of not revealing the clear text. This paper shows how Homomorphic Encryption can be used as a block cipher, and thereby appear to be anything other than cipher text, expanding its usefulness in multiple potential scenarios.

Introduction: Homomorphic Encryption is an encryption scheme which is inherently much more secure than traditional encryption schemes, because with its use conclusions can be drawn from the data without knowing what the data actually is. It is a stream cipher and stream ciphers have traditionally been reserved for data in transit, such as in the case of two satellites exchanging information about their trajectories so they can adjust to prevent collision.

This paper introduces the concept of Invariant Key-Point Homomorphic Encryption, an encryption scheme which changes Homomorphic Encryption from a stream cipher to a block cipher. This is accomplished by leveraging the malleability of Homomorphic Encryption to generate new, alternative sets, of the same encrypted data but with different cipher text. By generating these alternative sets of cipher text in conjunction with machine learning algorithms one could search the domain of all sets of ciphers for ciphers which closely represent some arbitrary clear text. This means that you could generate cipher text that does not look like cipher text. Importantly, it should not be confused with steganongraphy, a technique in which metadata is hidden in empty spaces within data structures. Instead, Invariant Key-Point Homomorphic Encryption modifies the data itself and presents it in different forms, while retaining the original decryption key. As an example, data could be encrypted and the cipher text modified to represent a song or a picture. The song or picture could then be streamed over a radio wave. For those listening to the song or watching the video it would sound or look as though there was some “static” or interference in the connection. However, for individuals who need to extract the data from the music or image, the “static” is actually necessary for retaining the integrity of the original data. The broadcaster could exfiltrate data from an unfriendly region by broadcasting music over the radio or a image over television. The recipient, having the key, could then decrypt the image or song and retrieve the clear text.

Methodology: In traditional block ciphers there is a concept that forensic analysists observe in cipher text in which the blocks that are XOR’d with other blocks in a cipher form “spirals” which are more commonly referred to as artifacts [Re 2]. These spirals are a byproduct of overlaying blocks of encrypted data without previously encrypted data. This makes the cipher rigid because it is being XOR’d overtop of itself and requires knowledge of the initialization vector (the starting point of the cipher).

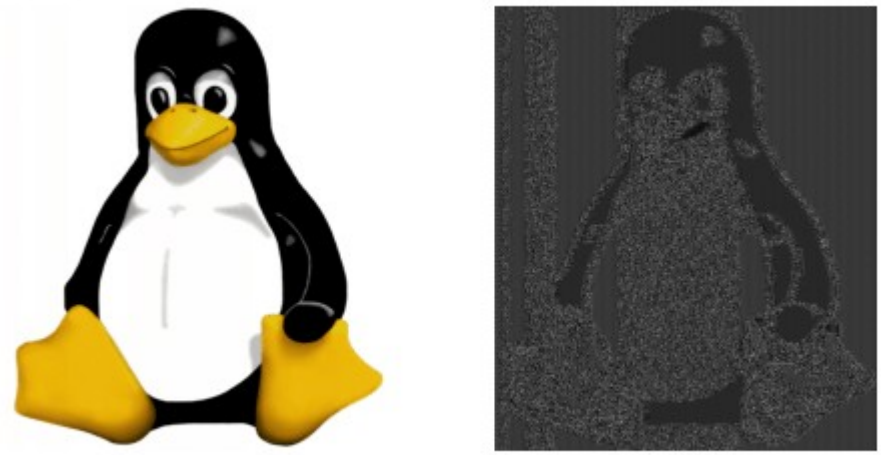
Traditional block ciphers require rigidity as a means of security, to avoid being weak and crackable. They attain rigidity by using XOR. Patterns or “spirals” in the cipher are a byproduct of this XOR operation. When blocks are XOR’d with other blocks in cipher form they require knowledge of some initial vector in order to maintain integrity of data, but even without such knowledge encryption can be broken by focusing on the “spiral” artifacts.

Homomorphic Encryption does not rely on rigidity for security, instead, it relies on malleability. In encryption malleability is a property where cipher text can be modified to produce a predictable outcome without knowing the encryption key. This is considered a bad property to have because it allows someone to manipulate encrypted data which can lead to attacks. In Homomorphic

Encryption malleability is considered beneficial because it allows computations to be performed directly on encrypted data without decrypting it. This is the main feature of Homomorphic Encryption and it is used in cases where data privacy is critical.

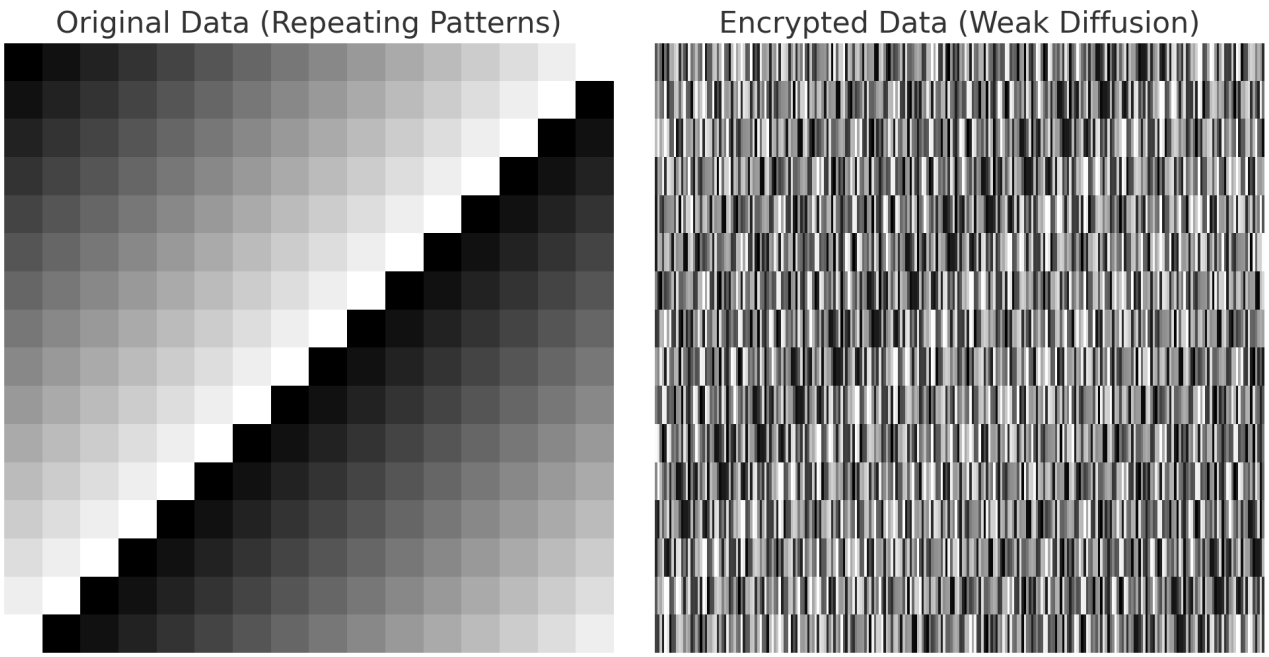
Block ciphers are limited because they do not allow for variation in the cipher rendering, they are deterministic. Homomorphic Encryption is not a block cipher, so the number of ways in which you could render the cipher from computations is limitless. What you get after a computation is performed is a new cipher where the integrity of the underlying clear text was preserved and the information was never revealed, but the answer to a computation can be extracted by the individual probing the data. The points at which blocks are overlayed with other blocks inherently creates patterns in the cipher. Traditionally, you could measure the entropy of the cipher by looking at these patterns. This is made possible because the set of encrypted bytes within the proximity of the overlayed blocks will be more tightly coupled in their values to the block as their value is deterministically defined by the overlay point. However as you extend outward, away from this point of rigidity, the number of values a byte can be become less dependent on that particular overlay point. This is not to imply that you can always arbitrarily replace old values with new values, but the set of values with which a byte can be represented does increase up until it approaches the proximity of another overlay point.

Different block ciphers create different patterns in the cipher text and are inherently influenced by the data being encrypted as well as other factors such as initialization vectors. The best example of this is ECB mode pattern leakage.



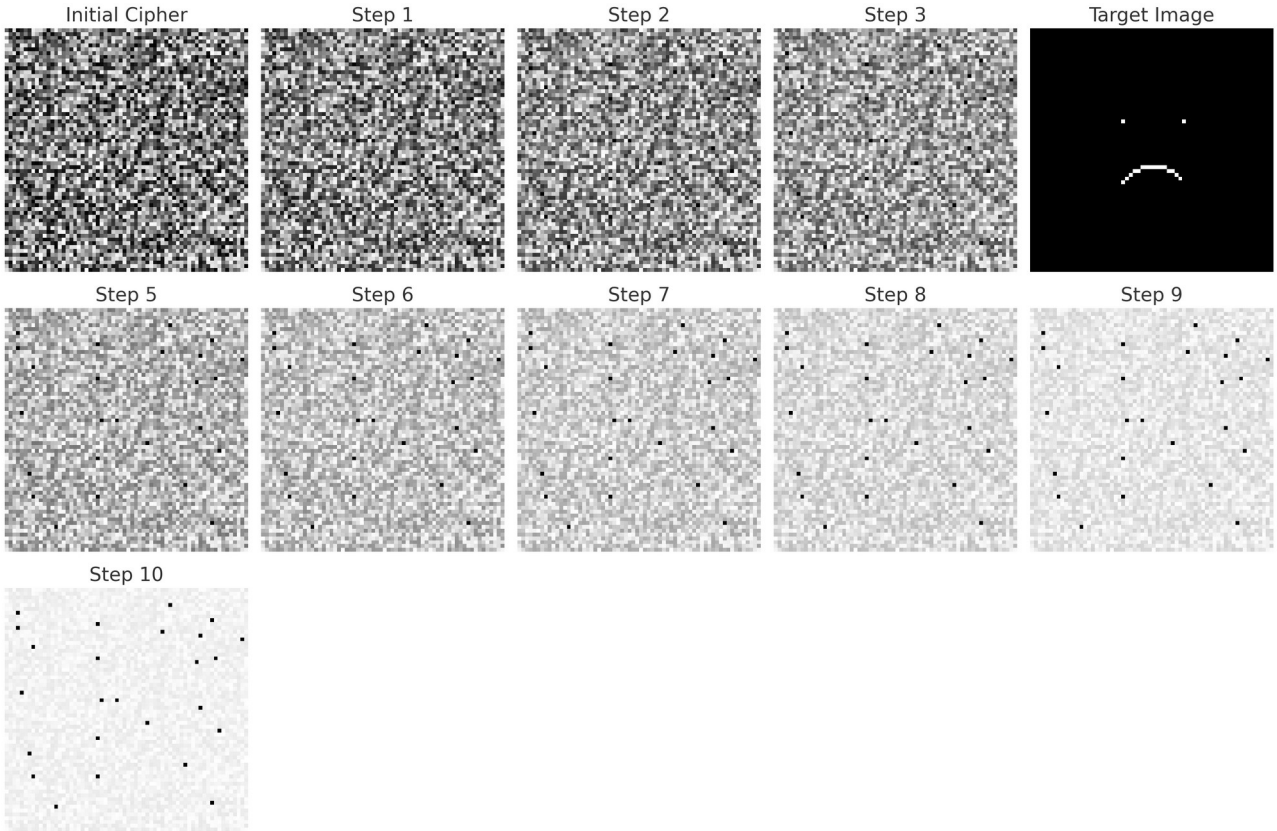
[Re 0: Example ECB mode pattern leakage]

This problem has already been solved by more sophisticated ciphers. Cipher text caused by low entropy or weak diffusion in a block cipher is a more relevant problem. This is exemplified below.



The image on the left is the original data, which contains repeating patterns generated using blocks of a specific size. The right image is the encrypted data using a simulated weak cipher with poor diffusion.

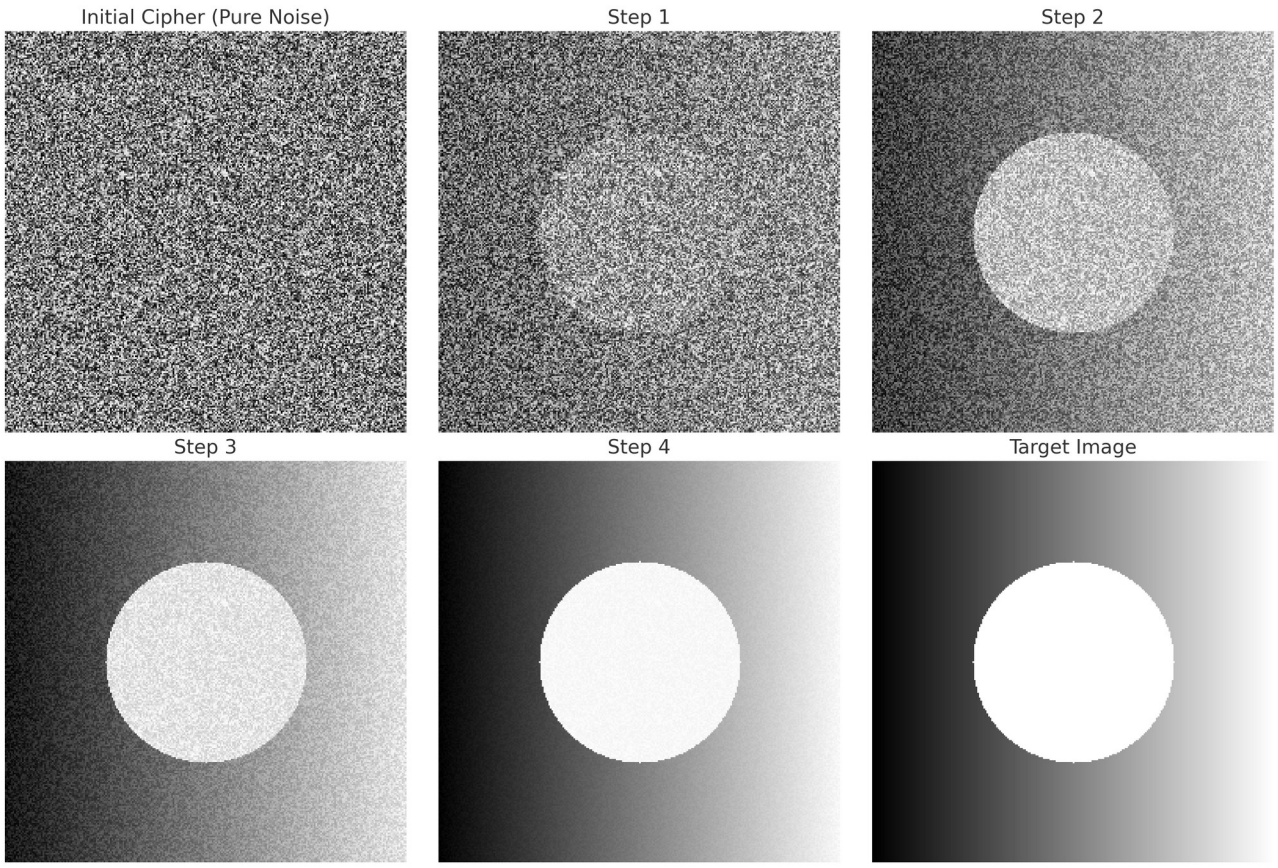
This demonstrates that the subsequent values of bytes are determined by previous bytes. If the problem is viewed through the lens of Homomorphic Encryption the problem is eliminated. It is possible to leverage the cipher to render different forms in which all generated cipher texts represent the same underlying clear text with the same key but they are all different in appearance.



The example images above show how the process would look like for approximating the target cipher. Notice that there are points in the image which do not change in the cipher over the various iterations. These points are referred to as points of invariance. In mathematics invariant points are ones which remain unchanged under a specific transformation or operation. The concept of invariance applies in various mathematical contexts, including geometry, algebra, and differential equations. It has applications in symmetry analysis, stability in dynamical systems, control theory, and computer graphics. The name “Invariant Key-Point” was originated in the SIFT algorithm from computer vision. The other points could be referred to as “cryptic sugar-points”, a term borrowed from syntatic-sugar in functional programming. Cryptic sugar-points simply refers to “extra” bytes which are data points that are generated in relation to the invariant key-points as well as their surrounding area. In the SIFT algorithm the invariant key-points are utilized for rendering 3D models from two or more images. The key-points in the SIFT algorithm are shown in the image below.

[Re 1: Invariant Key-Points from the SIFT algorithm]

Applying this concept to cryptography allows for cipher text exemplified by the stepwise pictures presented below.

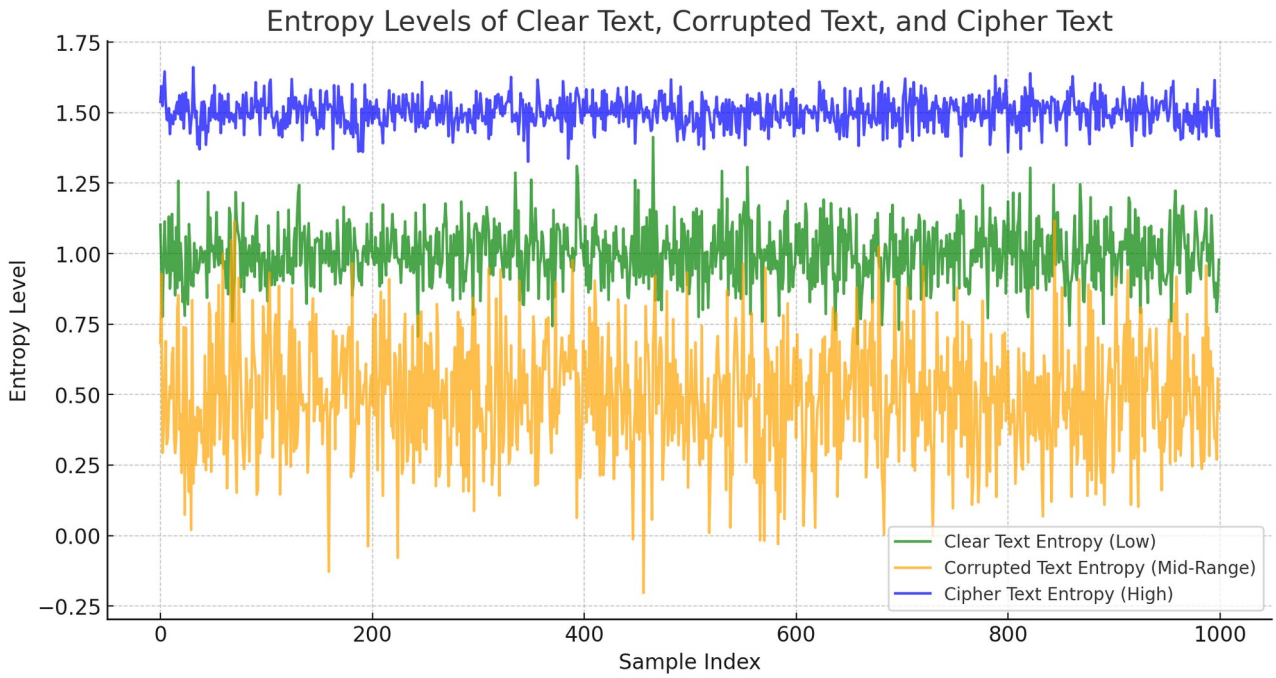


An iterative process is used to identify the invariant key-points. All other points become candidates for replacement as cryptic sugar-points. Substitution of these points is repeated until you get an assortment of pixels which most closely resembles the target image. Programmatically, the process should be terminated when the return value from a given iteration falls below two standard deviations from the average number of pixels substituted over the iterations of change in the cryptic sugar-points. So long as significant modifications are detected in the cipher, then the process will continue. When the trend changes such that there are no significant improvements or there is corruption in the invariant key-points, the process stops. In the case of corruption the process moves back one iteration.

Significance of the invariant key-points approach:

The reason this approach is important is that it allows for further probing of the cipher for answers. You could probe it with buffers or empty strings, and force it to render the original data as different sets of ciphers. This means that the cipher can be rendered in a seemingly infinite number of ways if you XOR it with buffers of varying lengths. The cipher could continue to be XOR’d until the cipher is rendered in a way that approximates some secondary set of clear text, such as an image or a song. This would allow for the safe transmission of the encrypted data, which as a byproduct of it being a cipher that imitates clear text, and can look like low-quality clear text data of a different set. The data itself could be analyzed forensically and pass as being authentically clear text.

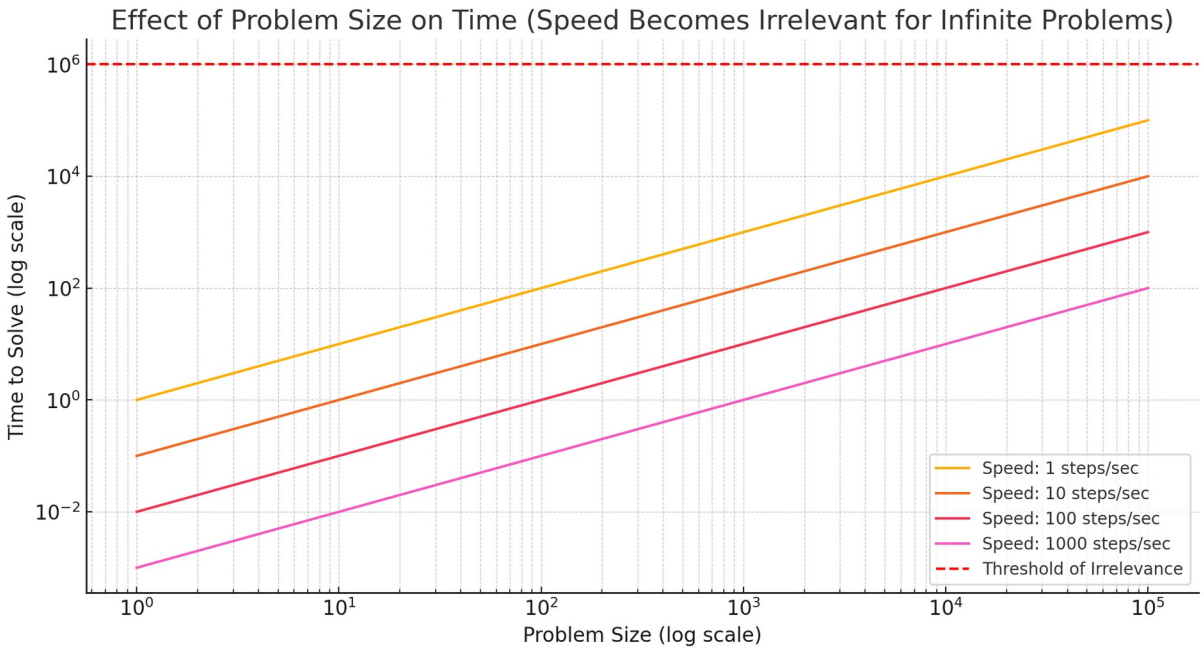
In the era of quantum computing traditional encryption will be rendered useless. They are easily identifiable as an increase in entropy. However this algorithm allows for a circumstance in which the only way to crack the ciphers would be to first distinguish legitimate clear text from ciphers. Using invariant key-point homomorphic encryption the entropy can be made indistinguishable from clear text even with quantum computing, since speed in terms of cracking ciphers becomes irrelevant as the domain of possible starting points becomes infinite.



In the graph above, the green line displays clear text with medium entropy and with minor variations, representing readable and structured data. The corrupted text in orange shows low entropy with larger variations, simulating clear text that is partially corrupted or degraded. The cipher text in blue maintains consistently high entropy with minimal variation, characteristic of well-encrypted data.

The challenge arises when the entropy level of corrupted text closely overlaps with clear text, making it difficult to distinguish between the two. This results in an infinite number of possible starting points, as corrupted data may appear indistinguishable from clear text data. In such cases,

brute-forcing or traditional analysis becomes infeasible due to the vast possibilities as shown by the graph below.



This graph illustrates how the time required to solve a problem grows with the size of a problem, even as computational speed increases. For an infinitely large problem set, no finite speed can overcome the magnitude of the task, rendering speed irrelevant in such scenarios.

Summary:

This application of Homomorphic Encryption has not been attempted or researched and is unlikely to be attempted because Homomorphic Encryption is a stream cipher and concepts utilized in it do not entirely overlap with concepts in block ciphers. However Homomorphic Encryption could be implemented as a block cipher and would likely provide a dramatically increased level of security.

References:

Re 0: https://crypto.stackexchange.com/questions/20941/why-shouldnt-i-use-ecb-encryption
Re 1: https://en.wikipedia.org/wiki/Scale-invariant_feature_transform
Re 2: https://en.wikipedia.org/wiki/Differential_cryptanalysis