Members:

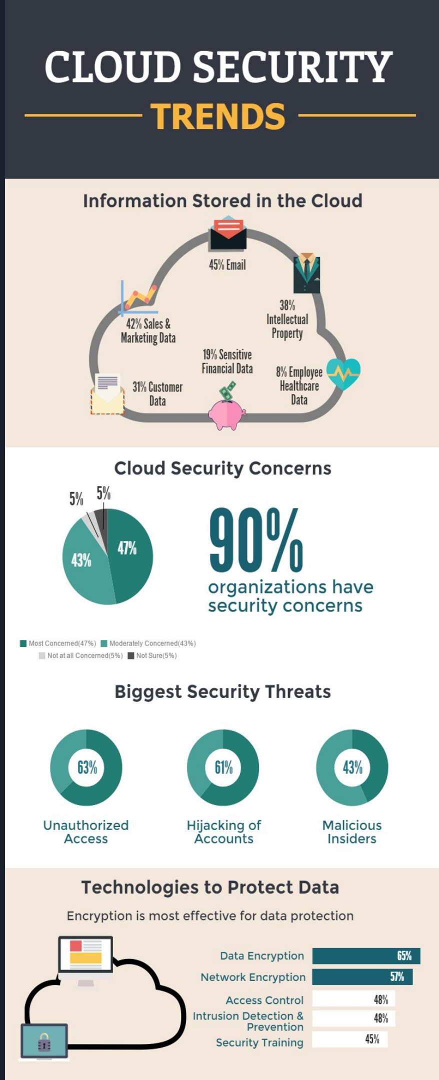
Jayant Jain	2017155
Karamveer Singh	2017156
Sagar Draveriya	2017186
Shiva Saini	2017190
Shubhankar Poundrik	2017194

Homomorphic Encryption for Cloud Computing : Images

Motivation

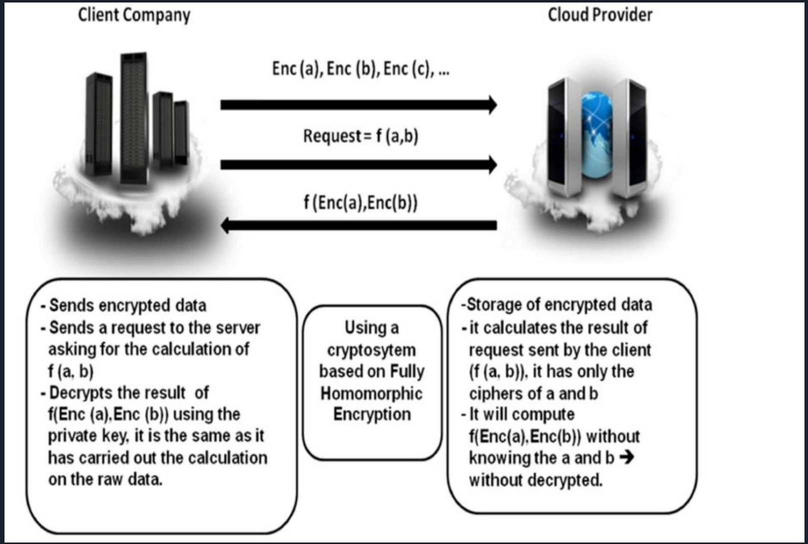
While doing computations in the cloud and keeping the data encrypted in the process can protect the data from attackers.

The mathematical structures are preserved in homomorphic encryption, that's why computation doesn't require the data to be decrypted which is a good point to keep the data secure.



Methodology

- Construction of three Homomorphic Encryption algorithms and verify their homomorphic properties.
- Extend the encryption algorithms for images.
- Construction of addition and constant multiplication functions for images to show that homomorphic image editing is possible.
- Analysis of Histogram and Correlation of the images.



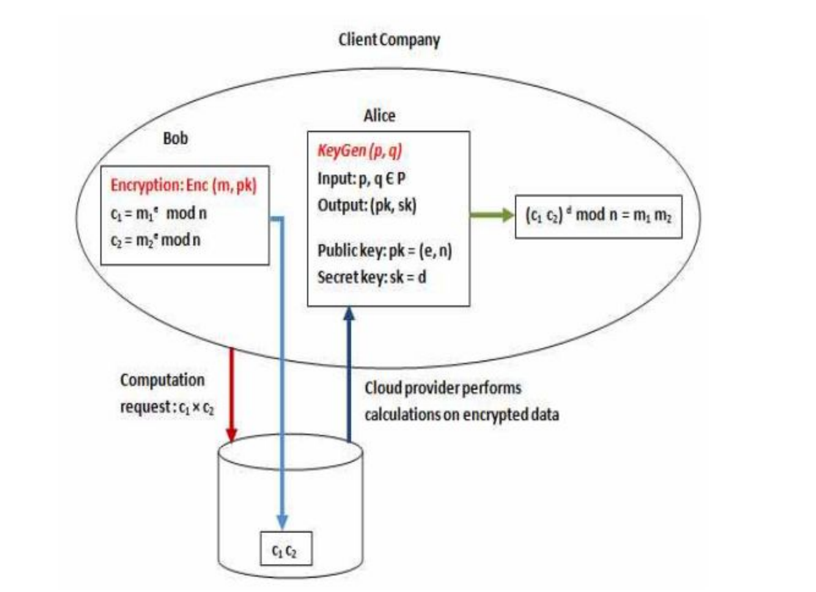
Homomorphic Encryption

- An encryption is homomorphic, if: from Enc(a) and Enc(b) it is possible to compute Enc(f (a, b)), where f can be: +, ×, ⊕ and without using the private key.
- The additive Homomorphic encryption (only additions of the raw data) is Pailler.
- Multiplicative Homomorphic encryption (only products on raw data) is the RSA.
- Additive Homomorphic:
  - Dec (Enc (A, r1). Enc (B, r2) mod n2) = (A+B) mod n, where A and B are real numbers
- Multiplicative Homomorphic:
  - Dec (Enc (A, r1) B mod n2) = A.B mod n.
  - Dec (Enc (B, r2) A mod n2) = A.B mod n.

Proposed Homomorphic Encryption Algorithms

- Multiplicative Homomorphic Encryption (RSA cryptosystem)
- Additive Homomorphic Encryption (Paillier Cryptosystem)
- Fully Homomorphic Encryption

Example



Paillier Cryptosystem (Additive Homomorphic Encryption)

Public key encryption has three stages: generate public-private key pair, encrypt a number, decrypt a number.

1. Selection of two large primes, *p* & *q*, such that $gcd(pq, (p-1)(q-1)) = 1$.
2. Compute $n = pq$ & λ = lcm($p-1$, $q-1$).
3. Select random integer *g* where $g \in Z_{n2}$
4. Ensure *n* divides the order of *g* by checking the existence of modular multiplicative inverse, μ where μ = (L (g^{λ} mod n^2))-1 mod n,

   where L(x) = (x-1)/n

   Public Key : (n , g)

   Private Key : (λ , μ)

Encryption

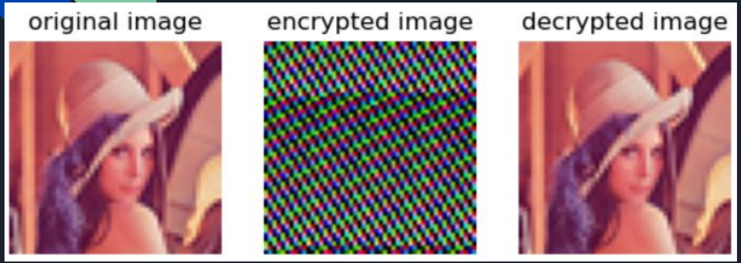
1. Let the message be m, $0 \leqslant m < n$, which has to be encrypted.
2. Select a random r , $0 < r < n$ & r $\in Z_n$ (also gcd(r,n) = 1)
3. Ciphertext, $c = g^m \,.\, r^n \bmod n^2$

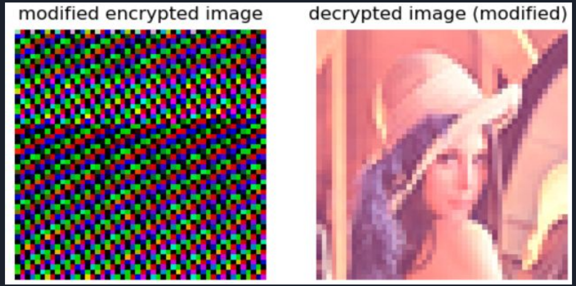
Decryption

The ciphertext, c can be decrypted by:

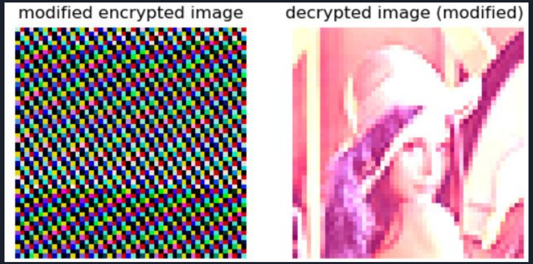
$$m = \left(L\left(c^{\lambda} \bmod n^2\right)\right) \mu \bmod n$$

Results of Paillier cryptosystem



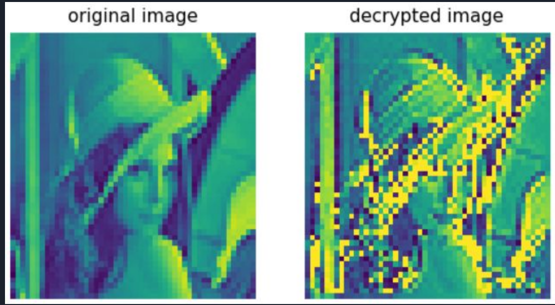


Decrypted image brightened by addition



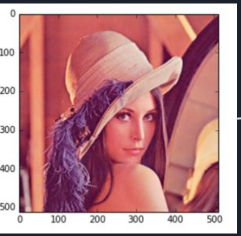
Decrypted image modified by Multiplication

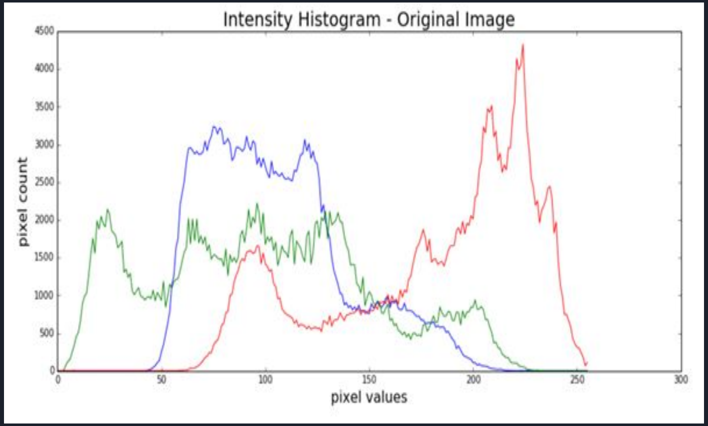
Original image brightened by addition

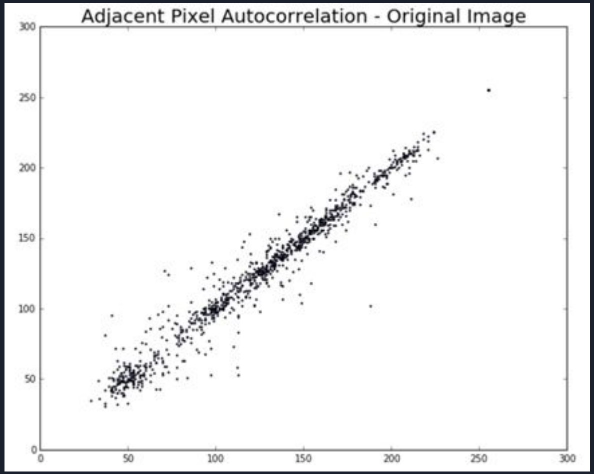


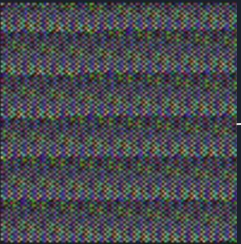
Unsharp masking

Paillier cryptosystem Analysis

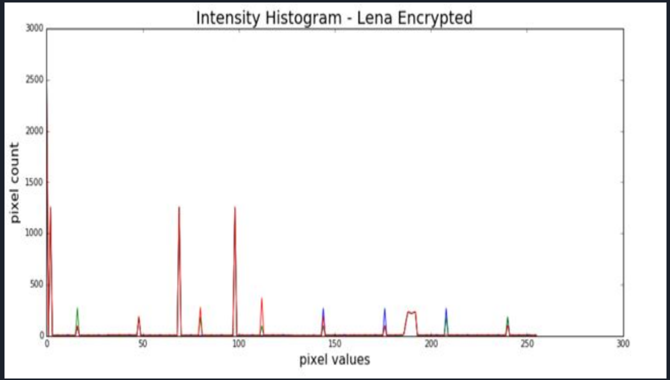


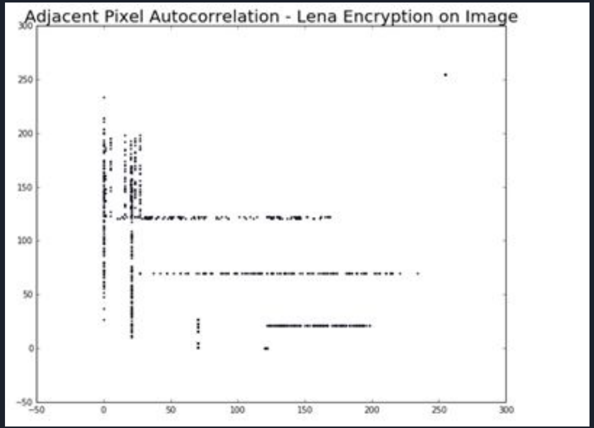






Encrypted





RSA Cryptosystem (Multiplicative Homomorphic Encryption)

The RSA algorithm involves following steps:

Key generation

- Choose two distinct prime numbers p and q
- Compute $n = pq$
- Compute $\lambda(n)$, where λ is Carmichael's totient function. Since $n = pq$, $\lambda(n) = \mathrm{lcm}(\lambda(p),\lambda(q))$, and since p and q are prime, $\lambda(p) = \varphi(p) = p - 1$ and likewise $\lambda(q) = q - 1$. Hence $\lambda(n) = \mathrm{lcm}(p - 1, q - 1)$.
- Choose an integer e such that $1 < e < \lambda(n)$ and $\gcd(e, \lambda(n)) = 1$; that is, e and $\lambda(n)$ are coprime
- Determine d as $d \equiv e^{-1} \pmod{\lambda(n)}$; that is, d is the modular multiplicative inverse of e modulo $\lambda(n)$.

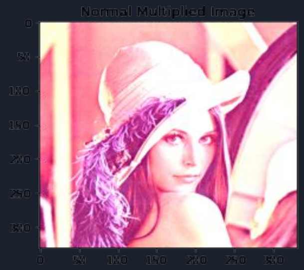
Encryption

- $C = m^e \bmod n$

Decryption

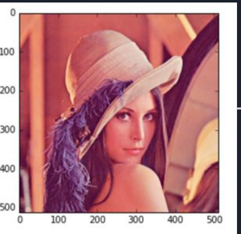
- $m = C^d \bmod n$

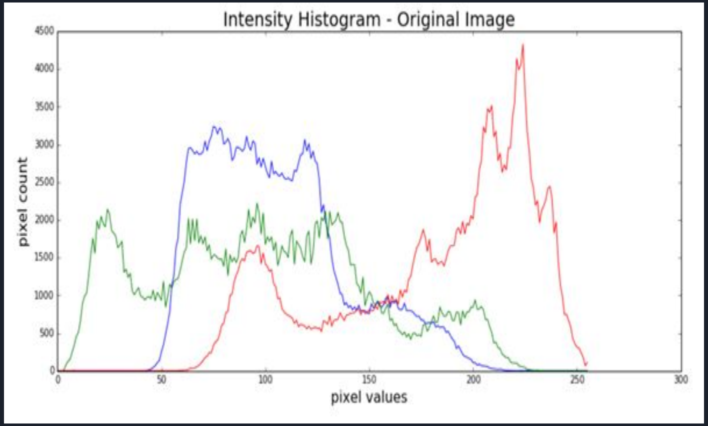
Results of RSA

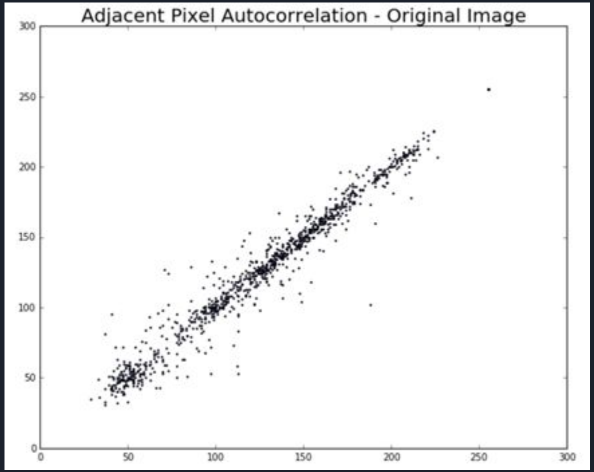


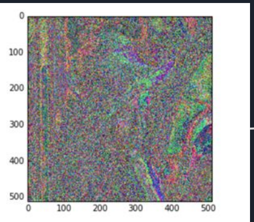
Original Image multiplication (x2)

RSA cryptosystem Analysis

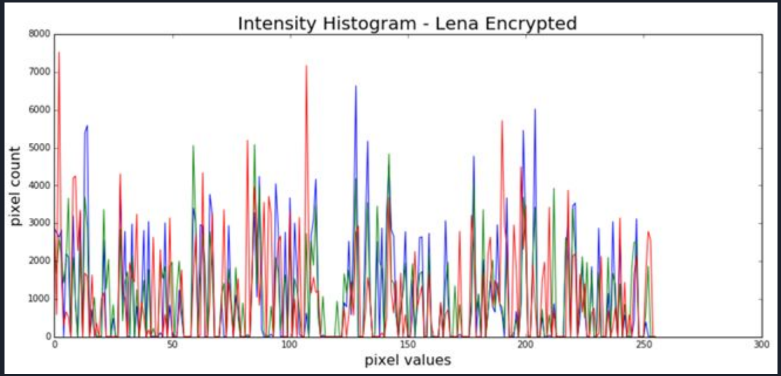


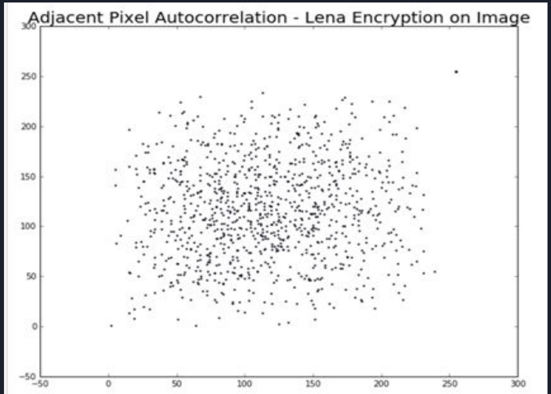






Encrypted





Fully Homomorphic (Gentry's)

This involves three steps: KeyGen, Encryption and Decryption.

- KeyGen returns a security parameter λ that specifies the bit-length of the keys.
- P= λ **2
- Q= λ **5
- The key is a random P-bit odd integer p.
- Encrypt E(key, m): Ciphertext c=m0 + key*q, where q is a random Q-bit number and m0 = m mod 2.
- Decrypt E(key, c): Output (c mod key) mod 2.

Conclusions

Homomorphic Encryption Cryptosystems

Characteristics	**Paillier**	**RSA**	**Gentry(Full)**
Platform	Cloud Computing	Cloud Computing	Cloud Computing
Homo Encryption Type	Additive,Multiplication	Multiplicative	Full operations
Privacy of data	ensured in communication and storage processes	ensured in communication and storage processes	ensured in communication and storage processes
Security applied to	Cloud Provider Server	Cloud Provider Server	Cloud Provider Server
Key Used by	The client (Different keys are used for encryption and decryption)	The client (Different keys are used for encryption and decryption)	The client (Different keys are used for encryption and decryption)

Future Work

The current proposed solution is computationally expensive and takes some time for large images which make its practice implementation very difficult.

We need to find some less rigorous Homomorphic encryption and decryption. This can explored in chaos map based algorithms.Eg- Arnold Cat, Bakermap

In practical purposes though, the calculation speed is scaled up by using GPU’s for parallel encryption/decryption purposes of an image.

References:

- Homomorphic Image Encryption Sachin Rana1, Om Jadhav1 , Shivam Rajput 1, Pranjal Bhansali1, Varshapriya Jyotinagar2
- Secure Cloud Computing through Homomorphic Encryption 1Maha TEBAA, Laboratory of Mathematics, Computer and Applications,, Faculty of Science, Rabat-Morocco
- Computing Arbitrary Functions of Encrypted Data Craig Gentry IBM T.J. Watson Research Center