RESEARCH ARTICLE OPEN ACCESS

Secured Image Scaling and Cropping Using Encryption

Supreet Johar[1], Prof.A.S.Narote[2]
[1,2] Department of Information Technology, Smt. Kashibai Navale College of Engineering (SKNCOE),Pune, India.

--*************************-----------------------------

Abstract:

Images sometime contain highly sensitive and personal information. If not protected, sensitive information in the images (e.g., MRI scan of a patient or G.I.S. maps) might be subject to unauthorized accesses by cloud providers. A naive approach to protect confidentiality of outsourced images is to encrypt the images before they are stored in the cloud. However, once this is done, it may not be possible to perform basic image processing operations, such as scaling and cropping. For instance, a remote pathologist, accessing a large histopathology image, would require first to access a scaled-down version, and then perform scaling and cropping operations to get a proper resolution for the Region of Interest (ROI). With images that are encrypted using standard encryption techniques, such operations would require the client machine to download the full encrypted images, decrypt them on the local machine, and then perform the operations. This makes the workflow slow and inefficient because a huge amount of data is pre-fetched and processed. 2DCrypto, a cloud-based multi-user encrypted domain image scaling and cropping framework based on the modified Paillier cryptosystem. For Practical deployment, a novel space-efficient tiling scheme for tile-level encrypted domain scaling and cropping operations is implemented. In 2DCrypto, a number of pixels in a tile are put together and the tile is encrypted instead of encrypting each pixel independently and also optimizes the modified Paillier scheme to limit its storage requirement. Thus a space efficient and secured framework for image scaling and cropping using encryption is implemented.

Keywords — **Tile Encryption, Paillier cryptosystem, Image Outsourcing.**

------------------------------------*************************----------------------------

1. INTRODUCTION

Cloud computing is an attractive paradigm for accessing virtually unlimited storage and computational resources. With its pay-as-you-go model, clients access fast and reliable hardware, paying only for the resources they need to use without the risks of large upfront investments. Nowadays, building applications for multimedia content hosted in infrastructures managed by third-party cloud providers is common. Images might contain highly sensitive and personal information. If not protected, sensitive information in the images (e.g., MRI scan of a patient or G.I.S. maps) might be subject to unauthorized accesses by cloud providers.

A naive approach to protect confidentiality of outsourced images is to encrypt the images before they are stored in the cloud. However, once this is done, it may not be possible to perform basic image processing operations, such as scaling and cropping. For instance, a remote pathologist, accessing a large histopathology image, would require first to access a scaled-down version, and then perform scaling and cropping operations to get a proper resolution for the Region of Interest (ROI). With images that are encrypted using standard encryption techniques, such operations would require the client machine to download the full encrypted images, decrypt them on the local machine, and then perform the operations. This makes the workflow slow and inefficient because a huge amount of data is pre-fetched and processed.

In this scenario, cloud providers are honest-but-curious. It is assumed that they do not tamper with the applications deployed in the infrastructure, but data might be collected or leaked. A typical example is replacing old hard drives with new ones, where the data has not been properly wiped out. Also, a full-fledged multi-user access model[9], where several authorized user access and modify the data stored in the cloud is assumed. In order to take full advantage of the cloud model, operations are offloaded as much as possible to cloud servers. However, to preserve confidentiality, operations are performed over encrypted images. In this work, focus is on dynamic scaling and cropping operations on encrypted images. These two operations can be combined to implement zooming and panning operations, which are necessary to navigate through large images (such as maps). In this way, no information contained in the images can be leaked to the cloud servers, and at the same time, users can fully exploit the cloud model by delegating most of the computation to the cloud.

Hereby, we present a practical cloud-based Multi-user encrypted domain image scaling and cropping Framework based on the modified Paillier cryptosystem. For practical deployment, we propose a novel space-efficient tiling scheme for tile-level encrypted domain scaling and cropping operations. The main contributions of our work can be summarized as follows:

- A full-fledged multi-user scheme where users can view and process images without requiring any key sharing. Our key management approach suits the need of organizations having a dynamic workforce where managing shared keys is challenging.

- The modified Paillier-based cryptosystem scheme neither requires more than one datacenter nor assumes that an adversary cannot access more than certain number of datacenters at any time. Therefore, 2DCrypto is more suitable for practical scenarios and it provides stronger defense against colluding attacks.

- To overcome high overheads resulted from encrypting an image, we propose a novel space-efficient tiling scheme that allows tile-level scaling and cropping operations. Using this scheme, we can encrypt a tile of pixels rather than encrypting each pixel independently. Furthermore, we optimize the cryptosystem to further limit its storage requirement. As a result, 2DCrypto requires approximately 40 times less storage than the naive per-pixel encryption

- 2DCrypto supports any factor scaling and cropping on encrypted images. These operations can be combined to support zooming and panning operations, which are two key features in image streaming. Compared to similar approaches, 2DCrypto does not create and store multiple copies of the same image. Moreover, from the cloud server to the user, only the requested processed part of the image is sent.

2. Literature Survey

The use of cryptosystems for hiding images is a well-studied area. A number of approaches, including but are not limited to, Public Key Cryptosystem (PKC)[7],watermarking[4], Shamir's secret sharing[5] and chaos-based encryption [7], have been proposed to protect images. These schemes provide confidentiality for cloud-based storage systems where a cloud datacenter does not perform any operation on the stored image.

To allow cloud datacenters to perform operations on the encrypted image, partial homomorphic cryptosystem-based solutions have been proposed. A partial homomorphic cryptosystem exclusively offers either addition or multiplication operations. Paillier, Goldwasser-Micali, Benaloh, Shamir's secret sharing[3] are among partially homomorphic cryptosystems that support addition. Whereas, examples of partially homomorphic cryptosystems that offer multiplication are RSA and Elgamal. Thus, the choice of a partial homomorphic scheme is heavily dependent on the type of operations to be performed in the encrypted domain.

Early works have focused on retrieving encrypted text documents. For instance, the first practical scheme for single keyword search on encrypted documents.

To improve performance, extended the encrypted search with indexing capability. Both works have been extended for searching using conjunctions of multiple keywords [7]. More recent works have focused on SQL-like queries supporting conjunctions and disjunctions[2]. Encrypted text-based search can also be applied to retrieval of encrypted images. However, the precision of the returned set is dependent on the quality of the keywords used for describing the content of an image. Few works have been proposed for searching encrypted images based on dynamic extraction of image features.

A more recent work[8] introduces a search scheme for encrypted images that is accurate and at the same time incurs computational overheads similar to a plaintext method. However, their scheme requires users to share the keys for accessing images.

Several works have been proposed for privacy-preserving face recognition where one party tries to match a face image with a dataset hosted by another party and both parties are interested in keeping their data secret from each other.

Shamir's secret sharing has been used for allowing encrypted domain scaling and cropping[4][5].Shamir's secret sharing-based schemes, however, can be infeasible for practical scenarios since they require n cloud servers. Moreover, these schemes are prone to collusion attack when k cloud servers collude. In contrast, 2DCrypto uses the Paillier-based cryptosystem that requires only one cloud datacenter and is more robust to collusion attacks. The Paillier cryptosystem is homomorphic to additions and scalar multiplications[6] and can be modified to a proxy-encryption-scheme.

3. PROPOSED SYSTEM

In this system, A distributed cloud-based image storage and processing system where a cloud server stores, scales, and crops an encrypted image on behalf of an image outsourcer is implemented. In the system model, the following entities are assumed:

- **Image Outsourcer:** This entity outsources the storing and processing (i.e., scaling and cropping) of images to a third-party cloud provider. It could be an individual or an organization, such as a hospital. In the latter case, several users can act as an Image Outsources. Typically, this entity owns the image. An Image Outsourcer is responsible for addressing security and privacy concerns attached to image outsourcing. To achieve this, the Image Outsourcer encrypts the image before sending it to the cloud datacenter. Further, the Image Outsourcer can store, delete and modify new images.

- **Cloud Server:** It is the part of infrastructure provided by a cloud service provider, such as Amazon S31, for storing and processing images. It stores encrypted images and access policies used to regulate access to the images. After making authorization checks, it retrieves a requested image from its image store. If the access request satisfies access policies, it scales and/or crops images in an encrypted manner, i.e., without decrypting them.

- **Image User:** it is authorized by the Image Outsourcer to access the requested image stored in an encrypted form on the Cloud Server. Depending on authorization, an Image User can issue either read request or process request (i.e., scaling and cropping operations). In both cases, the Image User decrypts the image returned by the request. Note that in a multi-user setting, (i) an Image User can modify an image that will be accessible by other Image Users, or (ii) an Image User can access images processed by other Image Users. In both cases, Image Users do not need to share any keying material.

- **Key Management Authority (KMA):** It generates and revokes keys. It generates a client and server key pair for each user, be it an Image Outsourcer or Image User. The client and the server side keys are securely transmitted to the user and the Cloud Server, respectively. Whenever required (say in key lost or stolen cases), the KMA revokes the keys from the system with the support of the Cloud Server.

- **Threat Model:** It's assumed that the KMA is a fully trusted entity. Typically, the Image Outsourcer directly controls by the KMA. Since the KMA deals with the small amount of data, it can be easily managed and secured. Also the KMA securely communicates the key sets to the Cloud Server and the Image User. This is achieved by establishing a secure channel. Except for managing keys, the KMA does not need to be involved in any operations. Therefore, it can be kept offline most of the times.

A honest but curious Cloud Server is considered. That is, the Cloud Server is trusted to honestly perform the operations on an image as requested by the Image User. However, it is not trusted to guarantee data confidentiality. The adversary can be either an outsider or even an insider, such as unfaithful employees serving the cloud service provider. Furthermore there are mechanisms to deal with the image integrity and availability of the Cloud Server.

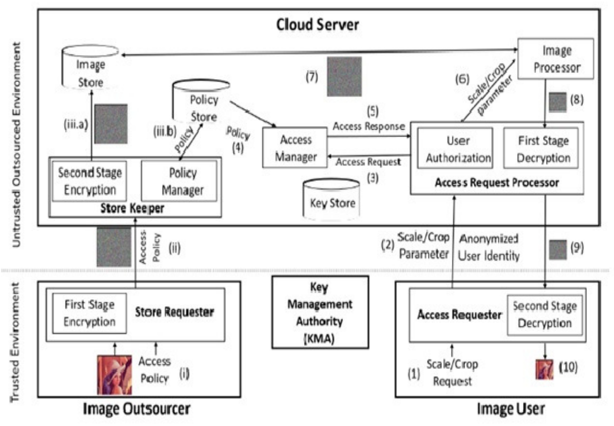

Figure 1: Proposed system Architechture

User login

In this Module User maintained various types of images can share to our friends and family with encryption. Here algorithm can used for encryption and cropping and scaling the images. In 2DCrypto, the number of decryptions, although dependent on the scaling and cropping factors, never exceeds the number of decryptions of the naive scheme. The worst case happens when one pixel in a tile is required by the scaling/cropping requester. Figure 1 explains the proposed system architecture.

Image Encryption

Although proposed tile-level encryption scheme 2DCrypto can have less computational and storage overheads than the naive per-pixel encryption, the flexibility of selecting an individual pixel is lost. Since 2DCrypto encrypts a tile of pixels, it requires a fewer number of encryptions (and hence encryption cost) than the naive scheme. In 2DCrypto, the number of decryptions, although dependent on the scaling and cropping factors, never exceeds the number of decryptions of the naive scheme. The worst case happens when one pixel in a tile is required by the scaling/cropping requester. On average, 2DCrypto requires a fewer number of decryptions as typically multiple pixels are put together in a tile. As an obvious consequence, computational (as fewer encryptions and decryptions are required) and storage overheads (as fewer encrypted values are stored and communicated) of 2DCrypto is lower than the conventional encryption. For instance, in tiling example of 4x4 size tiles, the encryption of RGB values of 64 pixels of a super-tile results in maximum 36864 bits, requiring maximum 24 bytes of cipher text to get one byte of plaintext color. In this way, 2DCrypto has more than 21 times improvement than the naive approach that independently encrypts each color component of a pixel. Figure 2 summarizes the process of tile-level encryption.

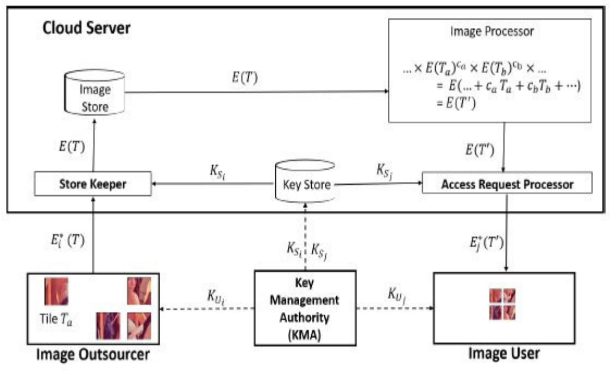

Figure 2: Overview of Overview of encryption and decryption processes

Mathematical details of encryption and decryption

For each user i (either acting as an Image Outsourcer or Image User), key pairs KUi and KSi generated by the KMA are securely transmitted to the user and the Key Store of the Cloud Server, respectively. During the uploading phase, the Image Outsourcer splits the image into tiles. Next, the Image Outsourcer encrypts each tile T using the user-side key KUi of the user i. The Image Outsourcer yields the ciphertext E i (T), which is sent to the Store Keeper. The Store Keeper retrieves KSi, corresponding to the user i, and re-encrypts the ciphertext E i (T). The re encrypted ciphertext E(T) is finally stored in the Image Store. This is a one-time-only operation performed when the image is stored for the first time on the Cloud Server. Upon request from the user, the Image Processor retrieves encrypted tiles from the Image Store, and performs bilinear scaling and/or cropping operation without decrypting them. These operations are shown in Equations 3 and 4, respectively.

E(T)c0 = E(c0 T) (3)

and

E(T1) E(T2) = E(T1 + T2) (4)

where c0 is the integer version of interpolating factor, and T, T1 and T2 denote the tiles. The Image Processor returns each processed tile E(T0) to the Access Request Processor. The Access Request Processor pre-decrypts the processed tile E(T0), using key KSj corresponding to the Image User j. The pre-decrypted tile E

j (T0) can only be accessed by the Image User j. The Image User client receives each pre-decrypted tile E

j (T0) and finally decrypts the processed tile T0, using the user-side key KUj

Diffie Hellman Algorithm

Diffie–Hellman key exchange (D–H) is a specific method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols as originally conceptualized by Ralph Merkle and named after Whitfield Diffie and Martin Hellman. D–H is one of the earliest practical examples of public key exchange implemented within the field of cryptography.

Algorithm:

The simplest and the original implementation of the protocol uses the multiplicative group of integers modulo p, where p is prime, and g is a primitive root modulo p. These two values are chosen in this way to ensure that the resulting shared secret can take on any value from 1 to p–1. Here is an example of the protocol, with non-secret values in blue, and secret values in red.

1. Alice and Bob agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).
2. Alice chooses a secret integer $\boldsymbol{a} = \mathbf{6}$, then sends Bob $A = g^a \bmod p$
   - $A = 5^6 \bmod 23 = 8$
3. Bob chooses a secret integer $\boldsymbol{b} = \mathbf{15}$, then sends Alice $B = g^b \bmod p$
   - $B = 5^{15} \bmod 23 = 19$
4. Alice computes $\boldsymbol{s} = B^{\boldsymbol{a}} \bmod p$
   - $\boldsymbol{s} = 19^6 \bmod 23 = \mathbf{2}$
5. Bob computes $\boldsymbol{s} = A^b \bmod p$
   - $\boldsymbol{s} = 8^{15} \bmod 23 = \mathbf{2}$
6. Alice and Bob now share a secret (the number **2**).

Both Alice and Bob have arrived at the same value s, because, under mod p, More specifically, Note that only a, b, and $(g^{ab} \bmod p = g^{ba} \bmod p)$ are kept secret. All the other values – p, g, $g^a \bmod p$, and $g^b \bmod p$ – are sent in the clear. Once Alice and Bob compute the shared secret they can use it as an encryption key, known only to them, for sending messages across the same open communications channel.

4. Result And Discussion

In implementation, input is as 250 X 250 size non-overlapping image. From each super-tile, nine 4x4 size overlapping tiles are created by implementing the space efficient tiling method. Then, each tile is encrypted and stored in a file. For a scaling or cropping request, required tiles are determined (at the Cloud Server end) based on scaling and cropping parameters. These tiles are then fetched from the file for the scaling and cropping operations. For example, for scaling by a factor of two, only the first four tiles of a super-tile are fetched and processed. After decrypting the processed tiles, the required pixels (at the Image User end) of a tile are determined from the scaling and cropping parameters, and the decrypted pixels are grouped to render the requested scaled/cropped image.

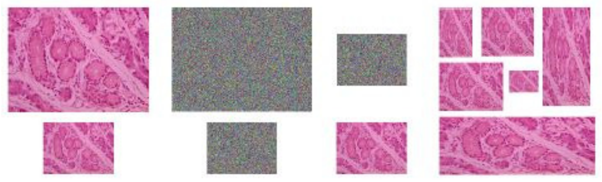
(a) (b)

Figure 3: Encrypted scaling.
(a) 1st column illustrates the user perspective - the original image (1st row) and the scaled image (2nd row) desired by the Image User; 2nd column shows the Cloud Server perspective - the encrypted image (1st row) and its scaled version (2nd row); 3rd column again shows the Image User side - the scaled image received, but before decryption (1st row) and after decryption by the Image User (2nd row). (b) 2DCrypto supporting multiple scaling factors.

Figures 3 shows 2DCrypto provides perceptual security in the cloud: in both figures the images on the Cloud. A server does not provide any information about the original image. Similarly, the images obtained from the Cloud Server side encrypted tiles also do not reveal any information about the original image. To retrieve an image, the Cloud Server performs the first round of decryption for those tiles sufficient to address the Image User's request. Figure 3 illustrates our encrypted domain scaling scheme .In 2DCrypto, the Cloud Server, which performs scaling and cropping, learns no information about the processed image. The Image User recovers the image by decrypting processed images.

5. Conclusion And Future Scope

Cloud-based image processing has data confidentiality issues, which can lead to privacy loss. Hereby this issue is addressed by implementing 2DCrypto, a modified Paillier cryptosystem-based scheme[9] that allows a cloud server to perform scaling and cropping operations without learning the image content. In 2DCrypto, users do not need to share keys for accessing the image stored in the cloud. Therefore, 2DCrypto is suitable for scenarios where it is not desirable for the image user to maintain per-image keys. Furthermore, 2DCrypto is more practical than existing schemes based on Shamir's secret sharing because it neither employs more than one datacenter nor assumes that multiple adversaries could collude by accessing a certain number of datacenters.

To make 2DCrypto practical, some improvements to decrease overheads resulted from the application of the modified Paillier cryptosystem are suggested. First, a space efficient tiling scheme that allows the cloud to perform per-tile operations. In 2DCrypto, a number of pixels in a tile are clubbed, and encrypt the tile instead of encrypting each pixel independently. Furthermore, optimized the modified Paillier scheme to limit its storage requirement. Due to these improvements, 2DCrypto requires approximately 40 times less cloud storage than the naive per-pixel encryption. The computational overhead is also significantly reduced because of fewer encryptions and decryptions rounds. The exact computational overhead and the data required by the image user, however, are dependent on the image size and the user's scaling and cropping parameters. For example, when a 512 * 512 image is scaled by a factor of two, the user needs approximately 5:3 times more data and works 2:3 seconds more than the conventional processing.

An obvious direction is to extend this work for compressed images. Another approach can be using our idea for addressing security issues in more specialized images, such as histopathology images and G.I.S maps. It will be interesting to investigate the properties of these specialized images to further decrease overheads. Another possible future work can be extending our work to video processing in encrypted domains.

Acknowledgements

I am incredibly thankful to my guide Prof. A. S. Narote for suggesting the topic for survey and providing all the assistance needed to complete the work. She inspired me greatly to work in this area. Her guidance and discussions with him are valuable in the realization of this work.

References

[1] C. Gentry, *A fully homomorphic encryption scheme*,"Ph.D. dissertation, Stanford University, Stanford, USA, 2009.

[2] M. Naehrig, K. Lauter, and V. Vaikuntanathan, *Can homomorphic encryption be practical?* in Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop, 2011, pp. 113–124.

[3] A. Shamir, *How to share a secret* Communications of the ACM, vol. 22, pp. 612–613, November 1979.

[4] M. Mohanty, W. T. Ooi, and P. K. Atrey, *Scale me, crop me, know me not: supporting scaling and cropping in secret image sharing* in Proceedings of the 2013 IEEE International Conference on Multimedia & Expo, San Jose, USA, 2013.

[5] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, *Public key encryption with keyword search* in Advances in Cryptology-Eurocrypt, 2004, pp. 506–522.

[6] M. R. Asghar, G. Russello, B. Crispo, and M. Ion, *Supporting complex queries and access policies for multi-user encrypted databases* in Proceedings of the ACM Workshop on Cloud Computing Security Workshop, 2013, pp. 77–88.

[7] T. Bianchi, A. Piva, and M. Barni, *Encrypted domain DCT based on homomorphic cryptosystems* EURASIP Journal on Multimedia and Information Security, vol. 2009, pp. 1:1–1:12, January 2009

[8] J. Yuan, S. Yu, and L. Guo, *SEISA: Secure and efficient encrypted image search with access control* in IEEE Conference on Computer Communications, 2015, pp. 2083–2091

[9] E. Ayday, J. L. Raisaro, J.-P. Hubaux, and J. Rougemont, *Protecting and evaluating genomic privacy in medical tests and personalized medicine*, in Proceedings of the 12th ACM Workshop on Privacy in the Electronic Society, 2013, pp. 95–106.