Privacy-Preserving Search over Encrypted Images

MCS

By

Mehmood ul Hassan

A thesis submitted to the faculty of Information Security Department, Military College of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfilment of the requirements for the degree of MS in Information Security

August 2020

Thesis Acceptance Certificate

Certified that final copy of MS/MPhil thesis written by Mr. **Mehmood ul Hassan** student of **MSIS-17** Course Reg.No. **00000275180**, of **Military College of Signals** has been vetted by undersigned, found complete in all respect as per NUST Statutes/Regulations, is free of plagiarism, errors, and mistakes and is accepted as partial, fulfillment for the award of MS/MPhil degree. It is further certified that necessary amendments as pointed out by GEC members of the student have been also incorporated in the said thesis.

Signature: ______________________

Name of Supervisor: **Asst. Prof. Dr. Shahzaib Tahir**

Dated: ______________________

Signature (HoD): ______________________

Dated: ______________________

Signature (Dean): ______________________

Dated: ______________________

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Mehmood ul Hassan
July 2020

Dedication

This thesis is dedicated to my Family, Teachers, and Friends specially my elder Brother, my beloved Rani Behna, and Anna for their love, endless support, and encouragement.

Acknowledgement

All praises to Allah Almighty for the strengths and His blessing in completing this thesis. I would like to convey my gratitude to my supervisor Asst. Prof. Dr. Shahzaib Tahir and co-supervisor Asst. Prof. Dr. Fawad Khan, for their supervision and constant support. I would thank my committee members; Assoc. Prof. Dr. Faisal Amjad, Asst. Prof. Mian Muhammad Waseem Iqbal, and Asst. Prof. Dr. Hasan Tahir (SEECS, NUST) for their support, guidance and knowledge regarding this topic. Their invaluable help of constructive comments and suggestions throughout the thesis work are major contributions to the success of this research.

I would thank Ph.D. scholar Khwaja Mansoor ul Hassan for his support, knowledge, and continuous help in all aspects. I would extend my gratitude to my colleagues Aiman Sultan, Bilal Ahmed, Ali Raza, Talal Hassan, and all my family members for their valuable feedback, support, and suggestions. I am also thankful to research associate Sir Asim and student Saif ur Rehman from AIR University for their help during the implementation phase of proposed scheme. I am grateful to the National University of Sciences and Technology (NUST) and The Punjab Educational Endowment Fund (PEEF) for awarding me a scholarship opportunity and making it possible for me to complete my master's degree and this research work.

Abstract

The rapid development in the field of cloud computing, big data, and machine learning, motivates individuals and enterprises to outsource their multimedia and image data to the cloud. Although, outsourcing reduces the storage and computational overhead for resource constrained devices at client's side, these services are still not getting attention due to security and privacy concerns. Clients care about the privacy of their data that is being shared and stored outside their jurisdiction. Fortunately, image processing in encrypted domain can overcome this issue. Current available techniques do not provide full privacy of image content, owner/client related information or have high computational cost. While retrieving the images from the cloud service provider (CSP), the client sends the query request to the CSP. These queries are not well protected and/or not randomized. Therefore, they are prone to traceability issues and do not provide security from search pattern leakage attacks. We propose a novel searchable encryption technique, which provides image content-based ranked searching and client privacy along with the un-tractability of client's search queries. The scheme prevents against search pattern leakage attack as it is based on probabilistic trapdoors. Theoretical and experimental analysis shows that the proposed technique is more secure and efficient as compared to the state of the art schemes.

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

AES	Advanced Encryption Standard
AE	Application Encryption
BBT	Balanced Binary Tree
CS	Cloud Server
CSP	Cloud Service Provider
COCO	Common Objects in Context
CBIR	Content-based Image Retrieval
CSPRNG	Cryptographically Secure Pseudo Random Number Generator
DCT	Discrete Cosine Transform
DWT	Discrete Wavelet Transform
EHR	Electronic Health Records

E2EE	End-to-End Encryption
FHE	Fully Homomorphic Encryption
GDPR	General Data Protection Regulation
HM	Hahn Moment
HVE	Hidden Vector Encryption
HE	Homomorphic Encryption
IBE	Identity-based Encryption
IT	Index Table
IPE	Inner Product Encryption
IP	Internet Protocol
KDM	Key Distribution Management
KMS	Key Management Server
LBP	Linear Binary Pattern
MITM	Man-in-the-middle
MAC	Message Authentication Code
MFE	Multi-fractal Feature Extraction
MRSE	Multi-keyword Ranked Searchable Encryption
OPE	Order Preserving Encryption

OPH	Order Preserving Hashing
OC	Outcome
PHE	Partial Homomorphic Encryption
PE	Predicate Encryption
PIR	Private Information Retrieval
PKE	Public Key Encryption
PEKS	Public Key Encryption with Keyword Search
RF	Relevance Frequency
SIFT	Scale Invariant Feature Transform
SE	Searchable Encryption
SHE	Somewhat Homomorphic Encryption
SURF	Speeded Up Robust Features
SSE	Symmetric Searchable Encryption
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
YOLO	You Only Look Once

Symbols

$\mathscr{A}$	Set of polynomial time adversaries $\mathscr{A} = \{A_1, A_2, ...\}$
$\mathscr{B}$	Polynomial time adversary
$\mathscr{C}$	Challenger
$\mathscr{D}$	Polynomial time distinguisher
$\mathscr{W}$	Set of keywords $\mathscr{W} = \{W_1, W_2, ..., W_m\}$
$\mathscr{I}$	Set of images $\mathscr{I} = \{I_1, I_2, ..., I_n\}$
$b \xleftarrow{\$} \{0,1\}$	Sample a random element of $\{0,1\}$ into b independently
(k_{pub}, k_{pri})	Asymmetric key pairs
$\cap$	Intersection of sets
$\cup$	Union of sets
λ	Security parameter
$\subset$	A proper subset
$a \leftarrow b$	A contains the value of b
$H(.)$	Cryptographic hash function
$H_K(.)$	Key-based cryptographic hash function
$id(I)$	Image identifier corresponding to the image I

K	Master key
k_s	Session key
iv	Initialization vector
$O(.)$	Big oh notation representing upper bound complexity
$st_{\mathscr{A}}$	State of adversary
Q_W	Query trapdoor generated for the keyword W
$log(.)$	Logarithm
$\oplus$	XOR

Chapter 1

Introduction

1.1 Overview

In the cloud computing era, all processing, computations, and storage are outsourced by individuals and enterprises. The cloud service provider (CSP) provides huge processing and storage space through internet connectivity [1]. Services provided by the CSP can be used in all fields of technology. The number of users relying on the services provided by CSP are increasing. According to research, top service providers, and their clients/visitors per month are shown in Figure 1.1. Individuals and enterprises working in the image processing domain require high storage space and processing resources. This requires the use of cloud computing resources in the image processing domain. Data stored online is out of control from the users which gives rise to trust issues between the CSP and its clients. User privacy and confidentiality of data is the main hurdle which hinders individuals and enterprises to adopt the cloud services [2–5].

Many CSPs, especially social media services, make use of the client's data for adver-

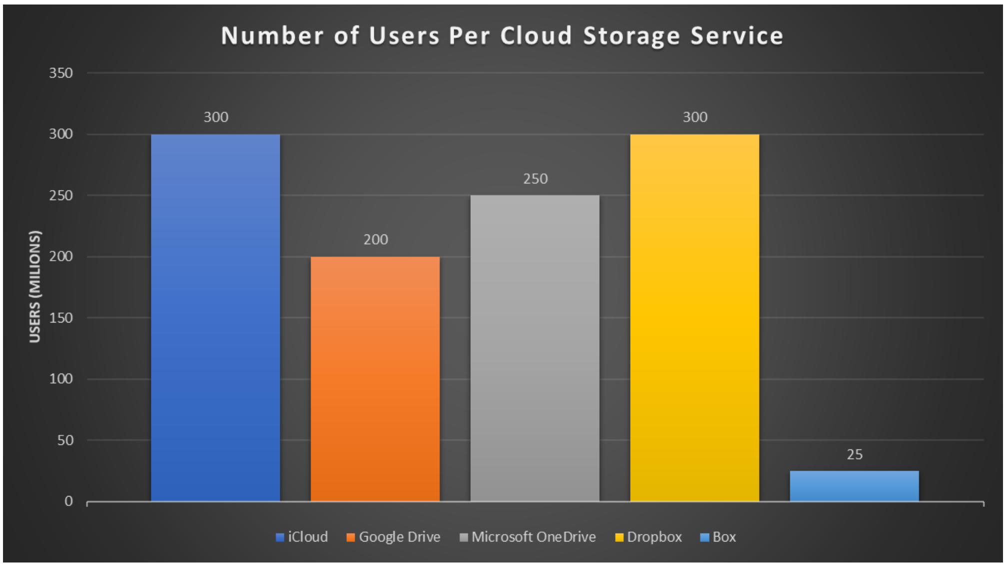


Figure 1.1: Number of Users per Cloud Storage Service Provider

tisement, promotions, and other business purposes. The person you meet in the morning will be showing in the "people you may know" list in the evening which makes serious security issues for the users in adoption of services provided by different CSPS to outsource personal and enterprise data. Online storage services provided by different CSPs are challenging for the users as CSPs can sell their user's data or expose it to third parties for maintenance, advertisements, or different purposes and company benefits [2, 6, 7]. To overcome this issue, users can make use of some third-party services or use encryption services provided by the CSPs [8].

While retrieving and searching over data, user can either download all the ciphertext, decrypt it locally, and can search over it. This way of searching over encrypted data is not feasible in real-time environments like in medical, media, army, and/or businesses. This type of searching required huge computing resources and hinders the basic use of outsourcing data. In the concept of smart cities and smart services, where we have huge data and it requires more time and resources to process the data. This way of processing also adds the computations on the client side which is not feasible. To overcome this problem, many researchers have proposed different techniques where required data can be searched over encrypted ciphertext.

Individuals and small organizations working in the image processing domain need high resources to process their data and to retrieve the results on time. To outsource this computations, privacy and security concerns are the main problem that restrict clients to rely on the CSP [9–11]. There is need of some machine learning algorithms and some user-end applications that cane solve this problem. This reason motivates organizations and individuals to outsource the data to the CSP. A semi-trusted CSP can be curious about user's personal data that violates the privacy and confidentiality of user data. The CSP can share the identity, profile information, interests, liking, and disliking *etc.* information to third party vendors for different business purposes.

To deal with such problems user needs to outsource encrypted data to the CSP. There are multiple issues associated with encryption as well *i.e.* key generation, key distribution, and encryption errors *etc.* In image processing, feature extraction over huge encrypted images data set is an open challenge for researchers from the past few years. To solve this problem, many researchers put forward their searchable encryption techniques in image processing domain. The outsourced images need to processed, searched, and provide required results to the user. Current available techniques provide users the ability to search over encrypted images and retrieve the most accurate results according to the user requirements and needs. Some application areas of SE techniques in image processing domain are briefly discussed here.

Applications:

Cryptography is the base of every field that involves IoT and the Internet. Searching over encrypted data is an important field nowadays. It has a vast variety of applications including businesses, education, entertainment, social networking, real estate, healthcare, finances, banking, music, media, and much more.

While working in the business domain, organizations working and controlling the personal data must be compliant to the GDPR across the European Union since 25 May 2018 [12]. Many small and large businesses are relying on cloud-based storage, they need a mechanism to secure their data over the cloud. Small businesses are relying more on the CSPs for storing and processing data. To do the business securely, businesses need to search their data securely. Searchable encryption is helping them to retrieve and process their outsourced data securely.

Searchable encryption has wide use cases in the education sector. The smart school system is an example of searchable encryption, where students and faculty attendance data is outsourced and is processed in encrypted form. Student enrollment, promotion, results, and progress tracks are stored over the cloud. In the smart school system, the data generated is stored, analyzed, searched, and processed in ciphertext and later decrypted when required [13].

Searchable encryption is also being used in sports. The selection of team members, voting for selection of team coach, financial funding related data, match schedules, salary and benefits details, all are stored on systems. This data is confidential and needs to be processed securely. Searchable encryption is guaranteeing the confidentiality of data along with the privacy of users and search results.

Social media services like WhatsApp [14] and Viber have been introduced and have been supplying their customers and users with end-to-end encryption (E2EE) services. This encryption makes the government authorities difficult to monitor internet traffic. Law enforcement authorities are trying to search for a way to monitor black sheep from participating in extremist and illegal activities. For E2EE authentication, both the sender and the recipient interpret the encrypted messages in social media applications because the password to decode the data is the end-user alone. No other person, including the vendor, may decode information even when the data is stored on the servers [15]. All the social media apps do not use E2EE, for example Facebook Messenger that encrypts the transit information only [16]. Some programs encrypt data but save decryption keys, that allow law enforcement agencies a chance for examination. Apps like Snapchat just encrypt information when it is in motion. When the receiver reads the message, data is then deleted [17].

1.2 Motivation

The global public cloud market is expected to expand by \$266.4 billion in 2020 and by \$260bin in 2023, at a yearly annual compound growth rate of 17 percent (CAGR) [18]. Despite increasing competition and demand, 75% of businesses have emphasized cloud computing security concerns. However, 60% of businesses have expressed questions about data protection. In fact, 2.6 billion important records were hacked globally since 2017 and this number is much more in 2018, culminating in 82 records breached in every second [19]. Such issues prevent people from benefiting from resource sharing and prohibit them from externalizing their private and confidential data over the cloud.

Enterprises and individuals are motivated to use cloud services due to multiple benefits associated with the usage of services provided by CSPs. There are some limitations as well. Like a semi-trusted CSP can make use of users' personal data by sharing the identity, profile information, interests, liking, and disliking, *etc.* information to third party vendors for their business purposes. This restricts the adoption and usage of cloud services.

To deal with such problems, the first step is to encrypt the data locally at the client end and then outsource it to the cloud. Many researchers have proposed their SE schemes which are not secure in terms of user privacy. These issues and challenges are briefly discussed in problem statement section.

1.3 Problem Statement

Nowadays every user is using smart devices connected to the cloud. With the increase of mobile usage and application development, a huge amount of image data is being generated each day. As the end-user devices are resource constraint in terms of storage capacity, the data to be outsourced to some external data storage medium. Cloud service providers offer storage, searching, and processing services over the internet without the restriction of geolocation or jurisdiction of the users. The data which is outsourced is out of control from the users. This puts the user's privacy in danger as most of the CSPs don't offer encryption as a service. Those CSPs which offer encryption as a service, have a lack of mechanism/technique which provides the privacy-preserving image retrieval process. The currently available techniques provide the deterministic searching which is still not secure where the user's privacy is a concern. Also, most of these image processing and retrieval schemes does not provide the image retrieval based on image content. Due to this reason, we need an image processing technique that can ensure the user's privacy in terms of batch queries and provides the probabilistic searching over the cloud and should support the image retrieval based on image content. This thesis focuses on the probabilistic searching and retrieval processes over encrypted image data based on image content.

1.4 Research Objectives

The main objectives of the thesis are:

- Analyzing already proposed privacy-preserving image searchable encryption techniques.
- Propose a novel privacy-preserving image encryption technique that can-do probabilistic searching for batch queries in ranked order. Image retrieval is based on content of the images *i.e.* objects.
- A detailed security and performance analysis of the proposed scheme by implementing and testing it over an open source dataset.

1.5 Research Methodology

This thesis gives a detailed analysis to the domain of SE. A privacy-preserving SE technique is proposed in this research for secure retrieval of images from a trusted but curious CSP. Already proposed schemes have limitations and prone to security issues like access and patter leakage attacks (discussed in chapter 2). The attacker can launch passive attacks and can trace the users based on queries that are deterministic and causes distinguishability attacks. To overcome these issues and preserve the users' privacy, a secure searchable encryption scheme over encrypted images is proposed. This scheme enhances query effectiveness and supports the image retrieval based on content of the images.

1.6 Thesis Organization

In summary, this thesis proposes a secure SE scheme that can preserve the user's privacy in terms of traceability attacks. We have divided this thesis into six chapters as given below:

- **Chapter 1: Introduction:** This chapter introduces the topic, some application ares, describes research objectives, and highlights contributions of this research.
- **Chapter 2: Literature Review:** discusses the preliminaries, searchable encryption, types, and latest SE schemes over encrypted images are discussed. The advantages and disadvantages of different schemes are discussed in detail. The security and performance analysis of each scheme is presented and a problem

statement is explained in detail.

- **Chapter 3: Proposed Searchable Encryption Scheme:** This chapter presents a novel ranked searchable encryption scheme based on probabilistic trapdoors/-query object keywords. Correctness and soundness of proposed scheme is also presented in this chapter.
- **Chapter 4: Security Analysis:** This chapter focuses on the security analysis of proposed SE scheme in terms of leakage profiles. A formal security analysis and verification of proposed scheme in accordance with security definitions are also presented. This chapter gives a detailed security comparative analysis of the proposed scheme with other related schemes present in literature.
- **Chapter 5: Performance Analysis:** This chapter gives a detailed performance analysis of the proposed scheme in terms of algorithmic analysis, storage overhead analysis, computational overhead, and performance analysis of each phase separately. A detailed discussion about the results is also drawn in this chapter.
- **Chapter 6: Conclusions and Future Directions:** This chapter focuses on the conclusion of research carried out in thesis and highlights the shortcomings faced during research. This chapter also discusses the future directions where SE can be explored in image processing, computer vision, and other application areas.

Chapter 2

Literature Review

SE is an approach that allows customers to access and search the encrypted data that is outsourced to the cloud while protecting the user's privacy. Privacy preservation is a function that limits the volume of data exposed in the SE scheme to the adversary or the CSP while outsourcing the confidential data. Chapter 1 presented the outline of the current SE system and discussed the issues associated with the secure storage and retrieval of data by preserving the user's privacy.

Domain usability and the related cloud infrastructure affect the architecture of the SE system [20]. Therefore, before developing a SE system, several design primitive elements must be discussed. This chapter gives an insight into the critical and groundbreaking research in the SE domain. The unlinkability of the trapdoor and keywords privacy is essential. The current available security definitions are limited to the scenario or can not be applied to new SE schemes. In this chapter, currently available SE schemes and security definitions are discussed, analyzed, and shortcomings of SE schemes are briefly discussed.

This chapter provides the basis for the other chapters and this review aims to establish a structure that follows the security and privacy objectives as discussed in previous chapter. It is important to understand the design elements before we review the existing literature and address the pros and cons of existing SE schemes in the image processing domain. A review of searchable encryption schemes is given here.

2.1 Searchable Encryption

As we know, cryptography can achieve the confidentiality and integrity of data over the insecure channel. The traditional searching mechanism fail to perform its function over the normal encrypted text [21, 22]. Therefore we need a searching mechanism over ciphertext which is done by Searchable Encryption (SE) in a cloud-assisted environment [23]. The idea of SE scheme was first proposed by Song *et al.* in 2000 which solves the searching problem over the encrypted message [22]. With the evolution of technology, research, and challenges, new techniques were proposed by researchers. Currently, multiple different techniques are being used [24]. These includes Homomorphic Encryption (HE), Private Information Retrieval (PIR), Multi-keyword Rank Searchable Encryption (MRSE), Inner Product Encryption (IPE), Predicate Encryption (PE), Hidden Vector Encryption (HVE), Identity-Based Encryption (IBE), Public Key with Keyword Search (PEKS), and Searchable Symmetric Encryption (SSE) as shown in Figure 2.1. These techniques are briefly described in next section.

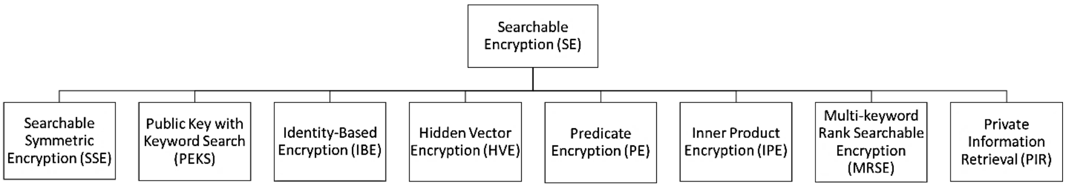


Figure 2.1: SE Techniques Classification

2.2 Types of Searchable Encryption

SE techniques are designed to provide secure, efficient, and reliable communication between the users and the cloud servers. These services are compatible with a single [21] and multi-user architecture [25]. These techniques support a single keyword search, multi-keyword ranked search, and fuzzy keyword search [21, 26–32]. We will briefly discuss some searchable encryption techniques here.

2.2.1 Symmetric Searchable Encryption (SSE)

SSE permits the client to access the cloud data with established anonymity through the delivery of secret and independent request. Secret requests or isolation queries require the cloud server to know only ciphertext. Data is searched through trapdoors that are generated securely. Searchable encryption involves 3 individuals in the whole process, including data owner O, authenticated user U, and semi-trusted or honest but curious cloud server CS [22]. Typically SSE involves four algorithms.

- $Keygen(1^k)$: The input arguments include a parameter k and gives a secret key K. This process is run by the data owner. It is a deterministic algorithm.
- $BuildIndex(K,I)$: This algorithm produces a secure keyword index *"I"* when the generated key K and image data is given input to the function. Data owner is

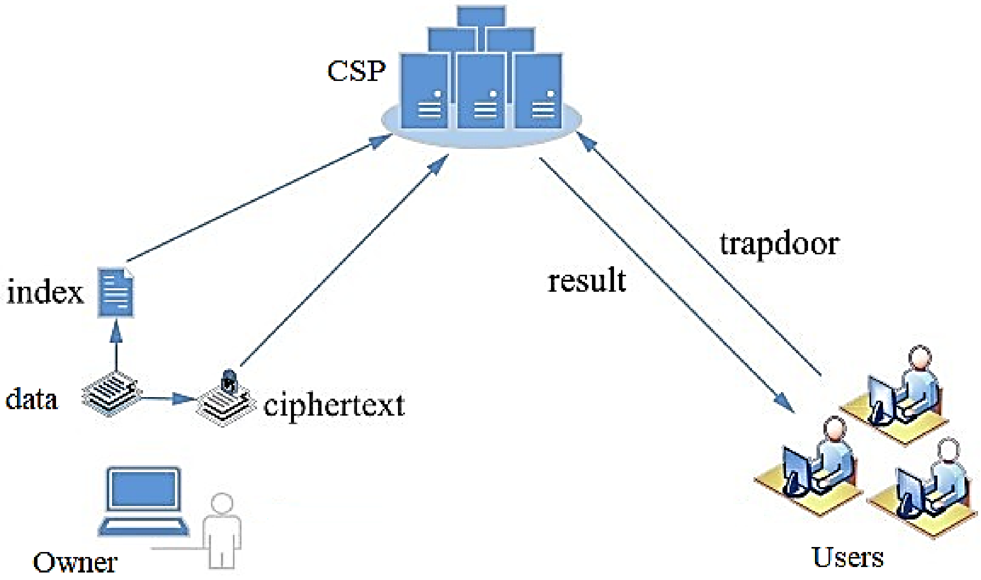


Figure 2.2: SSE technique model

responsible to run this phase. It is a deterministic process.

- $Trapdoor(K, w)$: This algorithm produces a bag of keywords, called trapdoors/search keywords. The secret key and a query word/feature is given as input to this algorithm and it produces trapdoors/search keywords T_w.

- $Search(I, T_w)$: This algorithm is processed by the CSP as it produces output by taking input the index table and trapdoors. It produces the output against each query value done by the user or the data owner.

A typical architecture of SSE is given in Figure 2.2.

Song *et al.* claimed that their scheme is reliable, efficient, and secure in terms of different security attacks like statistical attack for single keyword search [21]. In the context of SSE, however, this sort of safety is not robust enough and their scheme is vulnerable to information leakage for complex queey keywords. Also their scheme does not provide probabilistic searching. Goh introduced IND1-CKA and an improved IND2-CKA protection model to fix keyword index security [33]. The contents of data cannot be discovered by an attacker from the index table in both security models. There are multiple SSE schemes including single keyword, fuzzy keyword, conjunctive keyword, ranked and verifiable keyword search *etc,*.

2.2.2 Public Key Encryption with Keyword Search (PEKS)

PEKS was first introduced in 2004 [27]. In this technique, the data owner first encrypts the data & index table with his public key and outsources the encrypted data to the cloud. If user wishes to get the data from the CSP, he sends the ciphertext: $E_{A_{pub}}(M)$, $PEKS(A_{pub}, W_1)$, ..., $PEKS(A_{pub}, W_k)$. Here M is the data, A_{pub} is denoting the public key of owner, and $PEKS$ is a function that is providing searching functionality. The user generates the trapdoors T_w and send it to the CSP. The CSP returns the data which contains W in searching result to the user.

2.2.3 Identity-Based Encryption (IBE)

IBE was proposed for the first time in 1984 [34]. For encryption and decryption purposes, the key is generated from the client's identity. This identity key serves as a public key for encryption and only intended users having a valid private key can access and decrypt the ciphertext. As an example, a user can send an email to another person on his/her email address which is known to senders and the receiver can check the email by logging in his/her mailbox. PEKS is based on the IBE scheme. PEKS can handle chosen keyword attacks semantically secure in the random oracle model as proved by

Bilinear Diffie-Hellman (BDH) [27]. Two-level hierarchical identity-based encryption (HIBE) was presented by [35]. This scheme was designed to overcome the issues of encryption from one-to-many approaches, access control issues, and writing operations in cloud-assisted environments based on secure, efficient, and scaleable data collaboration scheme (SECO). During the encryption of data, many public keys of different users were used and only intended users with right private key is able to decrypt the ciphered data. To ensure the probabilistic and semantic security, SECO is banded with BDH.

2.2.4 Predicate encryption (PE)

It is a specialized type of searching over encrypted text which does allow the clients to search over the data without any private key corresponding to the public key. Instead of the private key, tokens are used and assigned to the query server. Query server, then, performs searching over the ciphertext based on that token. If searching produces any meaningful result, *i.e.* token finds an exact match, the ciphertext is then returned to the owner of a private key which further decrypts the ciphertext. This process is secure, and no information is leaked to the server [36]. In another scheme, access was controlled based on attributes of the users which was proposed by Goyal *et al.* [37]. To decrypt the ciphertext, private keys were shared with authenticated users. Special attribute-based features were added during the encryption process. This encryption involves a special type of mathematical relation along with multiple formulas with user attributes and private key of the user. According to different researchers [38], PE can be more efficient and reliable than traditional PEKS. Different attribute-based schemes are categorized under PE including attribute-based encryption (ABE), identity-based encryption (IBE), and anonymous identity-based encryption (AIBE) [36].

2.2.5 Hidden Vector Encryption (HVE)

It is a type of predicate encryption (PE) which supports the subset of search queries, comparative queries, and conjunctive combination of equality queries on a ciphertext [36]. It is a specialized form of PE as attribute-based two vectors related to token and ciphertext. During the encryption and decryption process, token matches the attributes to ciphertext if and only if the component of both are the same and equal. Moreover, it is possible to increase the basic equality rule so that conjunctive combinations of equality, subset predicates, and comparison can be increased. This allows better search queries over the encrypted text [39].

2.2.6 Inner Product Encryption (IPE)

IPE was introduced in 2013 by [36]. This cryptographic algorithm is specialized in achieving access control and special requirements and needs of the given task. IPE of IPC (inner product computation) is mostly used in HVE, IBE, and PE [40]. Another researcher [36], proposed different attribute hiding schemes which are different than payload hiding and are based on polynomial-time indications for disjunctions. To enhance the level of security, all secret information remains hidden until the attribute-based secret key is given to the algorithm for decryption. The purpose of attribute hiding and payload hiding is the same but is different in the working mechanism and the information it conceals from the ciphertext. Payload hiding only given the plaintext from the ciphertext while attribute hiding needs specific parameters that are associated with it during encryption process [41].

2.2.7 Multi-keyword Ranked Search Encryption (MRSE)

It was proposed in 2014 by [28]. With the help of the inner product similarity of keywords, documents are returned to the user when a searching algorithm is called. For better and accurate results, MRSE scheme was designed to choose the K nearest records from database (p_i) and query vectors (q). To ensure the communication secrecy over the cloud servers, secure inner products were implemented. This approach fulfills the privacy requirements of the users. Later, Li *et al.* cryptanalyzed the MRSE scheme and drawn three major security attacks [26]. MRSE is limited to the access frequency and keyword weight for the case when the documents are not at the top position in search outcome. To get the most relevant file from the outcome, it is difficult for the user to extract as search outcomes are not sorted and are presented out-of-order. MSRE uses the static dictionary for keywords which limits the searching efficiency as to add a new words in the list, construction of dictionary step is needed to perform again and again. To overcome the limitations of MRSE, [42] proposed a new scheme called multi-keyword query encryption (MKQE). In MKQE scheme, author have used the matrices partitioned approaches to overcome the limitations and issues of keyword dictionary expansion. To cope with the out of order searching and matching results, MKQE uses the index file along with the weights of keywords.

2.2.8 Private Information Retrieval (PIR)

PIR was proposed in 1995 by [43]. PIR protocol is the best approach to get the data from the CSP by keeping the information private and without revealing access patterns, search patterns, and query keywords to the CSP. A user can retrieve j^{th} of m^{th} bit data when multiple databases are stored on the cloud. This scheme is best for less computational communication environments where overall communication cost is less than the size of data itself. This scheme is limited to the keyword searching over un-encrypted text [44].

2.2.9 Homomorphic Encryption (HE)

HE is a sort of authentication system that permits the use of some computable functions on ciphertext by any third party (*e.g.* the CSP) while maintaining the basic usability and the layout of the encrypted data. As an example, we have m_1 and m_2 as two messages under the additively homomorphic encryption, one can get $E(m_1 + m_2)$ by performing the addition operation of $E(m_1)$ and $E(m_2)$ without getting any information about the messages m_1 and m_2. Confidentiality and privacy of messages are preserved with this encryption approach. HE has three types of encryption *i.e.* partial homomorphic encryption (PHE), somewhat homomorphic encryption (SHE), and fully homomorphic encryption (FHE) as shown in Figure 2.3.

All homomorphic encryption schemes can perform mathematical operations of addition and multiplication over the ciphertext. FHE can perform both operations at the same time. But it faced a memory usage issue while performing operations. SHE scheme was

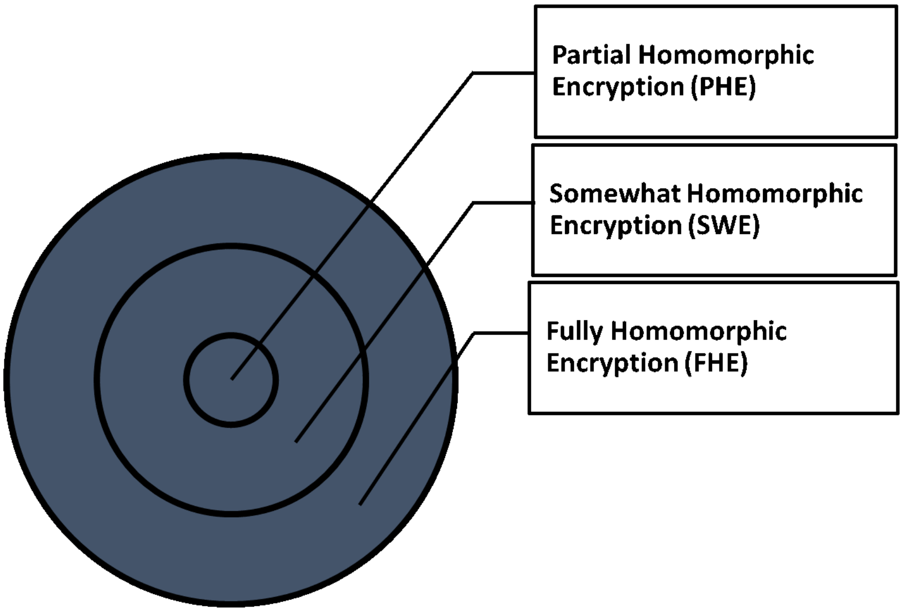


Figure 2.3: Homomorphic Encryption (HE)

the improvement over PHE but it was not efficient due to limited depth of circuits. PHE was only limited to the single operation of either addition or multiplication at one time and cannot perform both operations at the same time. The table 2.1 shows a comparison of all three schemes.

Table 2.1: Comparison of HE Schemes

Description	PHE	SHE	FHE
Computation on encrypted data	Yes	Yes	Yes
Limitation	One Computation Operation	Limited depth circuit	Large memory requirement
Example	Pallier, RSA, ElGamal	Gentry	Gentry, Fujitsu

We can see that FHE is not efficient in terms of memory usage. In the literature, many schemes were proposed to enhance and overcome this limitation [45–49].

2.3 Security Definitions

Searchable encryption got the attention of researchers back in 2000 when [21] proposed a searchable scheme to search over the encrypted data. There were no formal definitions present in the literature regarding searchable encryption. Few researchers come up with their definitions and assumptions, but they were based on scenarios and limited to some extent. In 2003 Goh [33], proposed the formal definitions of searching over ciphered data namely, Semantic Security Against Adaptive Chosen Keyword Attack (IND-CKA). He also proposed his searchable encryption scheme that satisfied these security definitions. He made some assumptions to prove his scheme practical *i.e.* to keep the indistinguishability intact, documents should have the same size and keywords should be of numbers. In that case, we don't need to keep the trapdoors encrypted and secure. These definitions were based on secure indices, but it lacks the probabilistic trapdoors. So, these definitions have limited scope in the SE context.

Later in 2005, [50] came up with the solution of limitations of definitions of Goh. Authors have proposed secure indexes based on bloom filters, called z-index which can overcome the limitation of the same size of documents. Later [51], highlighted the security issues associated with z-indexes and prove that their definition is not secure for searchable encryption schemes. To overcome the issues with SE schemes, Goh proposed enhanced and improved security definitions as IND1/2-CKA. According to IND1/2-CKA, documents were in no need of the same size and there was no need to keep the trapdoors secure. Curtmola R. *et al.* [51], claimed that all previously proposed definitions are lacking in the security of searchable encryption and can leak information about the users. He proposed 2 new security definitions for symmetric searchable encryption *i.e.* Adaptive/Non-Adaptive Indistinguishability Security for SSE. These definitions were also not in the fulfillment of security.

Later in 2017, [52] proposed new security definitions for ranked searchable encryption in terms of indistinguishability. They proposed a new searchable encryption scheme that fulfills the client's privacy requirements. This is scheme is only limited to searching over encrypted documents and requires more computations during query generation phase. We will be following the same security requirements and will analyse the proposed scheme against the security definitions proposed by [52].

2.4 Related Work

From the past few decades, researchers are trying to solve different mathematical and security problems associated with searchable encryption like privacy preservation and search pattern leakage issues [53–56]. Image processing in the encrypted image domain is a challenging task.

According to the paper [57], the first contribution towards searchable encryption using Scale Invariant Feature Transform (SIFT) by incorporating the homomorphic encryption was done by [53]. This scheme was not secure in terms of privacy-preserving and have high computational and storage overhead for the clients. In the paper [57], the authors has presented a protocol for outsourcing computations with privacy-preserving characteristics of image processing in an encrypted domain using SIFT. The author presented a scheme that can preserve the original characteristics of SIFT along with the preservation of user's privacy. They present a fresh and efficient SIFT exporting protocol to preserve its main features by randomly dividing the initial picture information, closely transferring the extraction tasks to two autonomous cloud servers. In terms of uniqueness and reliability compared to current alternatives, their protocol can maintain the significant features of the initial SIFT well. This scheme performs searching over encrypted images and can retrieve data from cloud server. There are some limitation with the scheme as this scheme does not support the image retrieval based on object. Another limitation of their scheme is the searching queries as they are deterministic in nature. This makes their scheme not secure as search patterns are the important part of the user privacy.

In the article [58], the author proposed the effective Privacy Preserving Linear Binary Pattern (LBP) system for retrieving LBP picture characteristics from encrypted pictures. The proposed model uses the MSB (Most Significant Bit) Image Plane encoding algorithm. All activities are done on encoded pictures without providing CSP with any information about images. The technique produces the same LBP function between encrypted and unencrypted pictures without extra communication overhead between the CSP and the users. The CSP is processing all necessary tasks for computations over the data. For the first time [58] proposed the LBP technique by dividing the image into a 3x3 matrix with center value as a binary number to secure the content of images. The proposed protocol reduces the communication overhead between the CSP and the client while performing feature extraction and less computational overhead at the CSP end. The user initially takes Bit plane randomization method with XOR based encryption on the most significant bit (MSB) plane to encrypt the pictures, and then outsources to the

CSP. The CSP then works on the encrypted images data. To ensure privacy and security, true random numbers are used. The proposed scheme is tested and employed over gray scale images. This scheme performs searching over encrypted images and can retrieve data from cloud server. The proposed scheme does not support content-based image retrieval and searching queries are not well protected.

In this article [53], authors have presented Privacy Preserving Hahn Moment (PPHM) with Somewhat Homomorphic Encryption (SHE). A mathematical model is presented for privacy preserving hahn moment by implementation and working in encrypted domain for reconstruction of images. The author has verified theoretically that plaintext HM can be utilizes along with plaintext image in the encrypted domain using PPHM. The confidentiality of image content is guaranteed by PPHM. Computational power and resources utilized by PPHM are more than normally used by DCT and DWT in encrypted domain. Discrete orthogonal Hahn moments have the benefits of having fewer computations to perform image encryption and searching. The noise-sensitive and outsourced image restoration capabilities are supported. This is used in multiple pattern identification applications [59–61]. In the encrypted domain, HM can be applied easily because Hahn's calculation includes additional processes and multiplication only. The implementation of PPHM with the help of SHE algorithm is proposed as a mathematical model. PPHM will increase the value of the orthogonal base function in relation to plaintext Hahn moments, and thus deduce the upper bound of Hahn moment so that after decryption, correct Hahn plaintext moments can be obtained. This scheme performs searching over encrypted images and can retrieve data from cloud server. This scheme is also not usable for real time environments because this scheme is not based on ranked searching. This scheme does not preserve the privacy of clients in terms of query randomness.

In this technique [54], secure modular hashing and K-means are used to preserve the user privacy in image outsourcing and image query results. K-means is a similarity evaluation clustering scheme which is based on distance between vectors. Similarity is higher with minimum distance of vectors and vice versa. A cluster is made up of all such closed vectors. With the help of K-mean, a secure index tree is constructed. Feature vector is extracted first from images and then encryption is applied *i.e.* AES or RSA similar to normal data. Then, a secure index tree is constructed from feature vector. Encrypted images and secure index tree both are stored on the cloud. When the user tries to retrieve the image, s/he will search an image from index tree and based on the relevance score, resultant image is sent back to the user. Image owner, then, shares the secret key of decryption with the user to decrypt the image. First, a secret key is generated as matrix M and string S. Matrix M is generated based on Gaussian distribution. The owner, then, shares this secret key (both matrix M and string S) to the users. To increase the accuracy, this index tree is generated with the help of K-means clustering function. To encrypt the images and index tree two different keys are used in the encryption process. To retrieve the data and search an image, the user first generates feature vector of query image, encrypt that vector with the same encryption key as was used initially to encrypt the original data *i.e.* M and S. The encrypted query then sent to the cloud server. The CSP starts searching from index tree from top to leaf approach. If a match is found, the CSP finds a most relevant image index based on hamming distance between encrypted non-leaf node and query feature vector. The CSP then forwards the encrypted image to the client. With the help of pre-shared private secret key the user decrypts the images. This scheme performs searching over encrypted images and can retrieve data from cloud server. There are some limitation with the scheme as this scheme does not support the image retrieval based on object. Another limitation of their scheme is the searching queries as they are deterministic in nature. This makes their scheme not secure as search patterns are the important part of the user privacy.

Another scheme was presented in [62]. This scheme was based on complex networks theory and SURF technique. Lu *et al.* [63] were the first who put forward the method of privacy preserving CBIR technique over encrypted image data. Jaccard similarity were applied on visual words after extraction of visual words from images. In this proposed technique, SURF algorithm with complex network is used to extract features from images. It provides single feature vector output for the input of single image and number of feature vectors for database images. To retrieve the image from the CSP, a feature vector is extracted from query image, applied classifier at database vectors and then based on Euclidean distance functions for feature matching function, array of matched images is extracted. Image sorting is applied on extracted array and query image is retrieved. This scheme performs searching over encrypted images and can retrieve data from cloud server. The proposed scheme does not support content-based image retrieval and searching queries are not well protected.

An efficient and privacy-preserving CBIR (EPCBIR) cloud-assisted scheme was proposed in [56]. For the representation of images, they have used Edge Histogram Descriptor (EHD) and Color Layout Descriptor (CLD). During the retrieval process, each query image was mapped to a unique feature. To build an index table, they used Locality Sensitive Hashing (LSH). These features were encryted and secured with the help of secure k-Nearest Neighbor (kNN) algorithm. As a result, top-k images are returned to the query client. This scheme provides feature-based image searching and retrieval while does not support the object-based image retrieval. Randomness added to the query features are not fully secure. Moreover, high computations are required during the search operations which makes this scheme less feasible for real time scenarios.

Another CBIR scheme was proposed in [55]. This scheme uses the same LSH and kNN algorithms for the encryption and security of images and their relevant features as used by [55]. This scheme also supports the detection of unauthorized query user by adding watermarks with retrieved images. If an illegal copy of image is found by data owner, the query client can be identified with the image watermark. This scheme have same limitations as of [56] *i.e.* query trapdoors are not fully random. High computations are required at the cloud server side and this scheme does not support the object-based image retrieval.

Another CBIR scheme was proposed using combined features [64]. This scheme utilizes the combination of new features extracted to use as a new descriptor. LSH algorithm was executed over the features and inverted file identifier vectors (IFV) calculated and stored in pre-filter tables. To improve the efficiency of proposed scheme, searching is performed over these tables. To increase the precision of retrieved images, these pre-filter tables are joined with IFVs. This scheme provides high efficiency in image retrieval process but have some limitations. This scheme does not support the object-based searching and limited to the image-based queries. Another limitation of this scheme is that query trapdoors generated are not fully random and can thus leak the information about queried images.

Based on Fisher vectors, [65] proposes a CBIR scheme by introducing K-mean algorithm. The authors have proposed this scheme namely SEISA by utilizing polynomial-based access control policy. This scheme is based on single writer single reader. Another scheme namely PIC was proposed by [66] to solve the problem of shared key during the query phase in CBIR. They introduced a new multi-level homomorphic encryption and CP-ABE method to address these issues. In shared key schemes, same key is used to encrypt the images, indexes, and query objects/features. PIC scheme utilizes different keys for query and index encryption. These schemes provide efficient searching over encrypted images but requires high computations during index generation and search outcome phases. Moreover, these schemes are limited to the feature-based image retrievals.

The researchers in [67] presented their secure EPIRM scheme that supports the image retrieval without splitting the feature vectors as done by KNN-CBIR schemes. This scheme returns the top-k images based on the features similarity with the query image. This scheme is supports the feature-based image retrieval and high computations are required during the index table generation and search outcome phases. These image retrieval schemes are mainly based on image features. There is a need of searchable encryption technique over encrypted images that can retrieve the images based on object keywords in ranked order and provide resistance to search pattern leakage attacks.

To retrieve the images from the CSP based on image content, the distance between image features and/or objects are identified and analyzed. In present research, to the best of my knowledge, there is no scheme which can provide searching on the basis of visual content of the images *i.e.* based on the objects present in image and there are few schemes present in literature which provides secure image retrieval based on image features like histogram, DCT, and DWT *etc,* [64, 67–71].

In literature, multiple SE schemes are available that work in the domain of encrypted images. These schemes are limited to use case scenarios, requires high computations, or have security loopholes. A comparison table 2.2 of discussed techniques is given here. Some merits and demerits are mentioned.

Table 2.2: Privacy-Preserving Secure SE Techniques over Encrypted Images

Technique	**Type of Encryption**	**Merits**	**Demerits**
SIFT [57]	SHE	SIFT characteristics preservation, accommodate invariance, secure from brute-force and other attacks.	High computation at client end, Search Pattern Leakage Attack.
LBP [58]	XOR	Reduces communication overhead between CSP and user, prevents insider attacks	TRNG problem, Limited to gray scale images, error in encryption, Search Pattern Leakage Attack.

Continuation of Table 2.2			
Technique	**Type of Encryption**	**Merits**	**Demerits**
Hahn Moments [53]	SHE	Less computations over encrypted data compared to DWT & DCT, less noise sensitive, no communication is required between owner and CSP.	High computations, Search Pattern Leakage Attack.
CBIR & DCT [72]	Stream cipher, permutation, scrambling	Index construction and searching at CSP end. Less computations at user end.	Index generation at CSP end. Query Traceability issue. Search Pattern Leakage Attack.
KSMH [54]	AES & Secure Modular Hashing	Index based searching	High computations at user end, key sharing, Search Pattern Leakage Attack.
CBIR [62]	Standard encryption	SURF technique (Local feature detection) & complex network theory	Feature vector and complex network generation, Search Pattern Leakage Attack.

Continuation of Table 2.2			
Technique	**Type of Encryption**	**Merits**	**Demerits**
EPCBIR [56]	LSH & kNN	Ranked searchable encryption technique	Does not provide content-based searching. More computations are required
SCBIR [55]	LSH & kNN	Ranked searchable encryption technique, detection of unauthorized user	Does not provide content-based searching. More computations are required
SEISA [65]	LSH & kNN	Policy and access based searching over images	Does not provide content-based searching. More computations are required
End of Table			

2.5 Summary

The design primitives that contributed in providing an overview of the various domains involved in the area of SE were presented in this chapter. The design primitives can be described through literature as a triangle, where the triangle vertices represent query effectiveness, security & privacy, and efficiency. The current research was divided into different categories based on the feasibility of the application and the schemes examined separately. The existing security concepts and their drawbacks were also addressed. The existing SE schemes are discussed in detail. Their performance and security analysis are carried out and a comparison is given in the Table 2.2. The limitations of these schemes are briefly discussed.

The literature review shows that the current available techniques are prone to traceability issues as they provide deterministic searching and also does not provide the image retrieval based on image content (Chapter 3 discusses it in detail). These techniques are not feasible for real-word cloud deployment as they are not protecting users from search patter leakage attacks.

Chapter 3

Proposed Work

3.1 Overview

In this chapter, we will discuss about the image processing techniques that are already present in literature and are being used in approximately every field. Types of object detection and a comparison table of image processing techniques is given for understanding the background of object detection and recognition. Among different object detection and recognition techniques, YOLO v3 [73] is selected for testing purposes. Any image processing technique can be used with the proposed encryption technique depending upon the user requirements, computing resources, and time. The proposed scheme is generic and can be used with any image processing technique. For testing purposes, Microsoft COCO image dataset is used [74]. Following contributions are made in this research:

- A novel privacy preserving ranked searchable encryption scheme over encrypted images is presented in this chapter. The proposed cryptographic system is based on probabilistic encryption schemes for the generation of search queries. These probabilistic queries resist the traceability issue and ensure the indistinguishability of search queries. Probabilistic encryption also resists the passive attacks.

- Search keyword-trapdoor indistinguishability and trapdoor index indistinguishability are explained to define the definition of *"privacy preserving"*.

- The proposed cryptographic system is implemented over Ubuntu operating system. The effectiveness and performance of this protocol are tested and analyzed.

The detailed security analysis of proposed protocol is given in Chapter 4 while performance analysis and detailed results discussion is given in Chapter 5.

3.2 Image Processing Techniques

Deep learning requires processes such as the recognition of artifacts using an image, photo, or webcam feed in the sub-discipline named "Object Detection." The use cases of object detection are infinite, whether they are object tracking, video analysis, security surveillance, detection for irregularities, crowd detection & human counting, auto pilot airplanes, self-driving cars, drones, face detection, and the list of applications continues [75]. The identification of artifacts can be inter-class or intra-class detection. Two solutions to object detection in a single image exist: computer learning and in-depth research. Current algorithms of detection are focused on the methodology of machine learning (ML) while new algorithms are based on deep learning (DL).

3.2.1 Types of Object Detection Algorithms

There are multiple ML and DL based object detection algorithms present in literature. Some machine learning based algorithms include:

- VJ Det. Viola–Jones object detection framework based on Haar features
- (SIFT) Scale-invariant feature transform
- (HOG Det.) Histogram of oriented gradients features detection

And deep learning approach-based object detection algorithms include:

- R-CNN (Region based Convolution Neural Network)
- Fast R-CNN
- Faster R-CNN
- Masked R-CNN
- R-FCN (Region based Fully Convolutional Networks)
- YOLO (You Only Look Once)
- SSD (Single Shot MultiBox Detector)
- SPP-Net (Spatial Pyramid Pooling Network)
- FPN (Feature Pyramid Networks)
- RetinaNet (Focal loss)

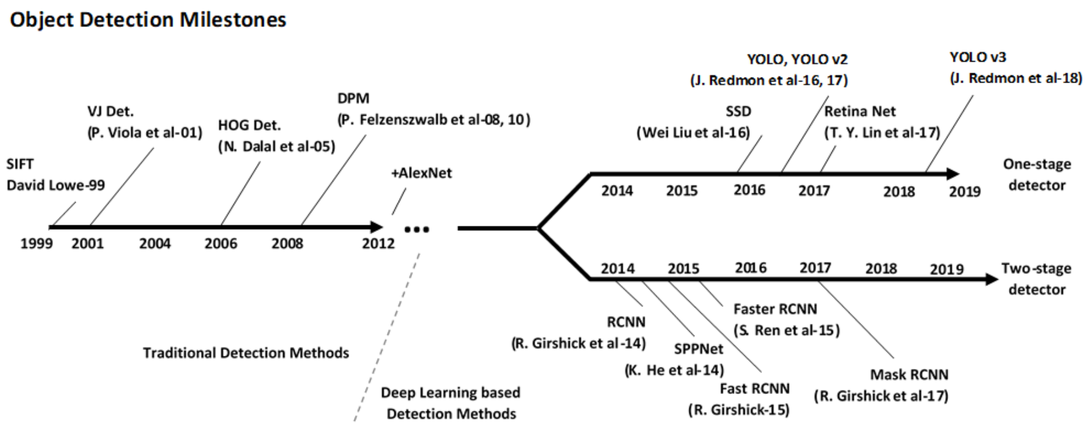


Figure 3.1: Milestone of object detection algorithms

The evaluation of object detection algorithms are shown in Figure 3.1 presented by [76]. Some algorithms are single stage while some are multi stage detection algorithms.

3.2.2 Single Stage vs Multi Stage Detectors

All region-based object detection algorithms are multi stage detectors. All object detection algorithm types that lay under R-CNN family are region based. These object detection algorithms work in two different phases: i) The algorithm finds the interested regions where an object is highlighted. Regions are selected based on selective or regional searching algorithms. ii) In second phase, these regions are processed and objects are identified based on confidence and highlighted regions.

For a single stage, the algorithms only process the image in a single phase. The regions are not selected for detection and recognition of images. The detection and recognition of objects is done in a single phase over dense sampling of possible objects locations in an image. From the comparative analysis of both single and multi stage object detection algorithms, we can clearly say that single stage detection algorithms are much faster than multi stage detection algorithms. Single stage object detection algorithms may have some limitations of not detection small objects or detection from a blur image [77]. A comparison of different object detection algorithms is drawn in Table 3.1.

Table 3.1: Performance comparison of different object detection algorithms

Year	**Algorithm**	**Stages (1/2)**	**Merits**	**De merits**
2014	R-CNN [78]	2	First to integrate CNN with RP methods; Performance improvement over previous state of the art protocols.	Multistage pipeline of sequentially trained (External RP computation, CNN fine tuning, training over BBR, SVM, and RP passing through CNN); Results extraction and running time is higher. Required more time and storage space.

Continuation of Table 3.1				
Year	**Algorithm**	**Stages (1/2)**	**Merits**	**De merits**
2014	SPP-Net [79]	2	A novel SPP in CNN family; Same performance as of CNN while takes much less time to execute the operations.	Gives same drawbacks as of RCNN; training requires more data; very slow in training and not feasible for real time detection.
2015	Fast R-CNN [80]	2	Enhanced version of RCNN; better performance and accuracy than RCNN and SPP; requires less computations and storage space.	Slow in training the model; very slow for real time applications.

Continuation of Table 3.1				
Year	**Algorithm**	**Stages (1/2)**	**Merits**	**De merits**
2015	Faster R-CNN [81]	2	Propose RPN for generating nearly cost-free and high quality RPs instead of selective search; Introduce translation invariant and multi-scale anchor boxes; much faster than previous relevant algorithms; requires less processing as it shares same conv. layers.	Training is complex, not a streamlined process; requires more time for training; slow for real time detection.
2016	YOLO [82]	1	Single stage detection algorithm; much faster than previous algorithms; requires less computational time and storage to process; no RP processing involves.	Small object detection is difficult; accuracy of detection objects is low; requires more time for training; slow for real time detection.

Continuation of Table 3.1				
Year	**Algorithm**	**Stages (1/2)**	**Merits**	**De merits**
2017	YOLO v2 [83]	1	Improves speed as compared to previous algorithm of this family; novel DarkNet19 based detection; higher speed and accuracy; good for real time detection; can detect more than 9K objects with YOLO9000.	Slow for real time detection; detection of small size objects and from blur images gives less accurate results.
2016	SSD [84]	1	Combination of features from YOLO and RPN; more accurate in detection; detection at multi-scale conv. layers; gives better performance than previous state-of-the-art algorithms.	Slow in real time detection; requires more time to process blur images; not efficient in detection small and blur objects.

Continuation of Table 3.1				
Year	**Algorithm**	**Stages (1/2)**	**Merits**	**De merits**
2017	FPN [85]	2	FPN works on basis of Faster RCNN; different scale images are processed easily; higher detection rate and more accurate than previous algorithms.	Training time and memory consumption increase rapidly; slow for real time detection; slow in construction of feature pyramid.
2017	Focal Loss (RetinaNet) [86]	1	High speed and simplicity, detector puts more focus on hard, misclassified examples during training with help of "focal loss" function. Focal loss gives high accuracy.	Lack of pool localization precision, unbalanced pos/neg data. slow for real time detection.
2016	R-FCN [87]	2	based on dependence and independence of RoI networks detection algorithm; performs better than previous relevant detection algorithms.	Requires high computations to process the image; slow for real time detection; slow in training the algorithm.

Continuation of Table 3.1				
Year	**Algorithm**	**Stages (1/2)**	**Merits**	**De merits**
2017	Mask R-CNN [88]	2	semantic segmentation masking is used in Mask R-CNN which solves the problem of instance segmentation; accuracy is higher.	The detection of mask is not efficient at pixel level; slow for real time detection; requires more time to process an image.
2018	YOLO V3 [73]	1	Faster object detection, good for real time environment.	Not good at detecting small objects; less accurate for less resolution images.
End of Table				

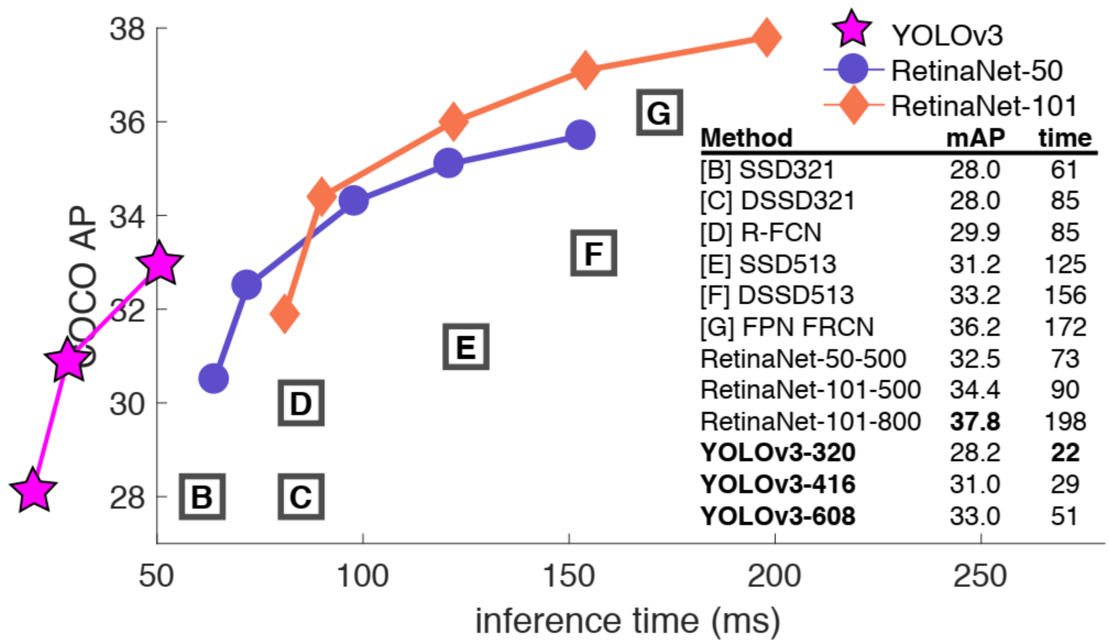
3.2.3 Speed vs Accuracy comparison

Object detection algorithms are compared based on their performance. The efficiency of object detection algorithm is measured by the number of objects detection and time required to process the image. This performance is measured in "mAP" called mean average precision ranging from 0 to 100. According to the authors [89], YOLO v3 given a great enhancement in speed as compared to the other object detection algorithms without compromising on the accuracy of the detection. All those algorithms which perform a detection in two stages are best in accuracy while algorithms with single stage detection takes much less time to complete the process of object detection in an image. A speed vs accuracy comparison is given in Figure 3.2 claimed by [90] and Figure 3.3 claimed by [89].

Figure 3.2: Speed vs Accuracy

For testing of our proposed SE scheme, we have used YOLO v3 [89] with MS COCO image dataset [74]. This dataset is most widely accepted and used feature rich image dataset with more than 330K images, 250K people with key points, five captions in each image, 91 categories, 80 object categories, 1.5m object instances, stuff and object segmentation, and context rich segmentation. As there are multiple benefits of using YOLO v3, it also have some limitations. Based on confidence percentage and bounding boxes this algorithm can detect objects in a better way. This algorithm works fine with limited training images and can work in both static images as well as real time videos. Model process time is much faster compared to other object detection algorithms and it takes much less power to give the output. While on the other hand, some drawbacks

Figure 3.3: YOLOv3 Performance



and limitations are also present there. The training of model can be slow if no dedicated GPU is present. The configuration and required model files are not present for Windows OS. It is hard to configure for Windows OS. Beside these limitations, YOLO v3 is best for small resource system.

3.3 Threat Model and Assumptions

To define the threat model, we are considering two entities for a SE scheme: the *CS* and the owner of the data. The owner of data encrypts the data that he want to outsource and then stores it to the remote *CS*. In our case, the data is the images that are being outsourced. The owner is assumed to be fully trusted and cause no security threat to the system. If the system is based on single owner multiple user, or multiple owner multiple user, the key distribution and security issues associated to that are not the part of this model. The main concern is with the *CS* as it acts as an adversary while performing security analysis of the SE scheme. The *CS* can launch different successful attacks. The adversary can have multiple characteristics as given below:

Honest but curious or trusted but curious server:

For the security analysis of proposed scheme, we are considering that the *CS* is honest but also curious or trusted but curious server. Trusted/honest ensures that the *CS* behaves in a recognized and approved way, but the *CS* is always happy and interested to get the complete or partial knowledge regarding the records and data that are submitted and kept to the CSP. The *CS* will only launch passive attacks to evaluate the data or to track network activity, to identify any data or details that could be correlated with encrypted content of images outsourced to the *CS*. We also assumes that the *CS* does not launch any active attack that could result in a service denial attack or any data modification.

Polynomial time adversary:

The polynomial time adversary means that the attacker is limited to execute the polynomially bounded number of operations only. These operations can be of encryption, decryption, or any passive attacks. The adversary is not allowed to perform unlimited number of operations in unlimited time to make a guess or deduce the actual plaintext.

Adaptive adversary:

The adversary can maintain the history about number of search queries, search results, and/or about the database *i.e.* the adversary can maintain the full history of all previous operations performed by the data owner/user. The security of SE scheme can be ensured if the adversary (*CS* in our case) can analyze the history and can choose any keyword adaptively while the SE scheme should perform securely.

Standard Model:

For the standard model, the *CS* *i.e.* the adversary is limited to the computational resources available and the time. The adversary is not assumed to have unlimited resources and time as the ideal adversary or replaced with the random oracle model. A novel privacy preserving technique is presented in this thesis based on mathematical hard/complex problems *i.e.* integer factorization problem which cannot be solved in a polynomial time and this scheme can provide a high level of security in this standard model.

3.4 The System Model

In the proposed scheme, we are considering a single reader single writer (S/S) model. A client-server model is visualized where two parties are involved *i.e.* Bob as a user and the *CS*. Bob wants to outsource all of his images collection $\mathscr{I} = \{I_1, I_2, ..., I_n\}$ to the remote *CS*. The *CS*, then, can perform searching, addition, and deletion operations over this database according to the user requirements. In this scenario. the *CS* is trusted-but-curious. Bob has performed an image object recognition and identification algorithm *i.e.* YOLO v3 [89] to extract the objects, called query keywords $\mathscr{W} = \{W_1, W_2, ..., W_m\}$, from each image. In case of MS COCO image dataset [74], number of object classes are 80 *i.e.* $m = 80$. Bob have calculated the relevance frequency (*RF*) of each keyword/object within the set of images. The *RF* is important while performing the ranked search and addition of new images to the dataset. The ranked searching is useful when a user wants to get multiple images with the same keyword present as the *CS* can find multiple images satisfying to the user object query. For the complex queries, the *CS* may find less number of images which satisfy the query. The relevance frequency *RF* is calculated from the output of YOLO v3 codded in Python language. The *RF* is calculated for each object and each image present in the dataset. This operation is performed by the data owner before outsourcing and encryption of images. Python code is given in Appendix A.

The relevance frequency is solely dependent on the object detection algorithm. Two stage object detection algorithms may find more number of objects including small and blur objects while one stage object detection algorithms may not perform well compare to two stage detection algorithms. Two stage detection algorithms are very resource intensive and hard to use them for real time object detection.

After finding the relevance frequency for each object in each image, an index table *IT* is generated by Bob. Later, he encrypts the index table *IT* and each image present in image dataset $\mathscr{I}$ and outsource both to the *CS*.

For searching over the encrypted images, Bob will enter a search keyword and an en-

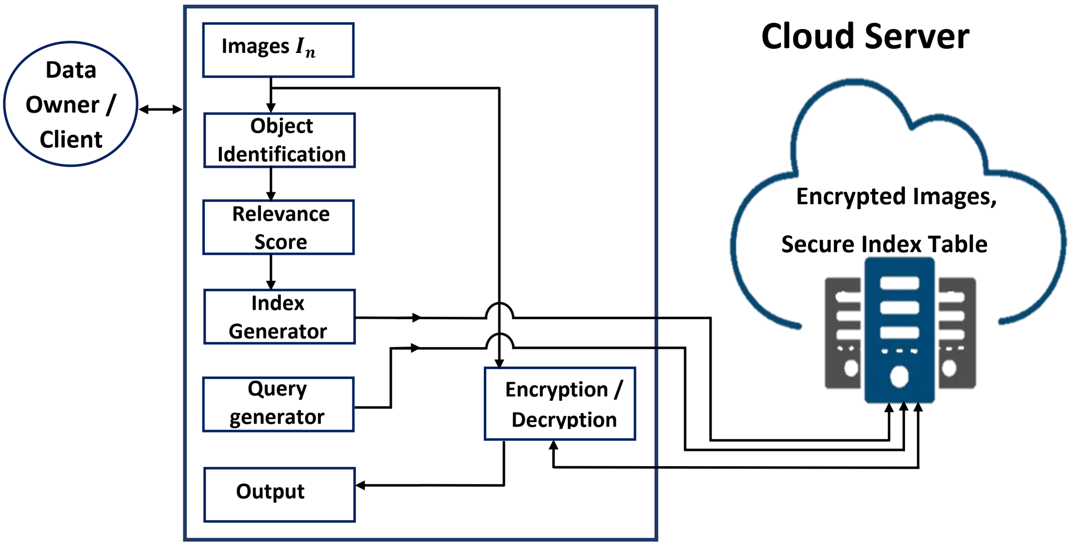


Figure 3.4: System Model Diagram for SE

crypted query is generated "Q_W" based on probabilistic encryption and send to the cloud server. The CS will search "Q_W" from index table IT and returns the number of images in ranked order. The system architecture and flow of events in proposed SE scheme is shown in Figure 3.4. Here, the user/the owner is interacting with a system that is performing all tasks and operations for that user. This system is interacting with the CS. From the system model, it can be seen that most of the operations are being performed at the user end while searching and storing of encrypted images along with the secure index are at the remote CS.

3.5 Probabilistic Encryption

To propose a probabilistic encryption SE scheme, we first revisit the definition of probabilistic encryption which is also termed as randomized encryption proposed by [91].

3.5.1 Probabilistic Encryption

It is a quadruple (M,K,C,Π), where K, M and C are the key, message and ciphertext space respectively, and Π is the relation as $\Pi \subseteq M \times K \times C$ so that: there is a message $m \in M$ for each $c \in C$ and $k \in K$ so that $(m,c,k) \in \Pi$ and there is a ciphertext $c \in C$ for each $m \in M$ and each key $k \in K$ so that $(m,c,k) \in \Pi$. For the probabilistic encryption, the search query keyword generation process should be probabilistic. If that is true, the result query keywords will not be deterministic and it will ensure the indistinguishability and resist the traceability attack. This indistinguishability is explained in scenario related to passive attacks in Chapter 4.

3.5.2 Design Goals

While constructing an SE scheme, there must be some security and performance goals. These includes but not limited to: *probabilistic query keyword, lightweight*, and *privacy preserving*. To resist the distinguishability attack, each query keyword must be generated differently even if the same object is searched repeatedly. This makes the scheme probabilistic and secure the users from traceability issues. The search pattern of each query should be kept secure and hidden while the access pattern can be exposed to the attacker (a.k.a the *CS* in our case). The scheme should be secure in know ciphertext model. This means that the *CS* should not know any information about the query keywords even knowing the index table and having the history of previously searcher keywords *i.e.* in case of an adaptive adversary. Also, the SE scheme should be computations friendly and can run at the *CS* as well as at the user end smoothly.

3.5.3 Security Definitions

For the proposed SE scheme, we are considering the definitions of indistinguishability, for query keywords (a.k.a trapdoors) and query index (a.k.a trapdoor index), same as defined by [52]. Chapter 4 shows that the proposed SE scheme complies with the following definitions.

3.5.3.1 Keyword-Trapdoor Indistinguishability for SE

Keyword-Trapdoor Indistinguishability is a process of performing search over ciphertext so that the encrypted query keyword should not reveal any relevant data about the unencrypted query keyword. If a same keyword is being search in plaintext repeatedly, the associated ciphertext or trapdoor should not be distinguishable even if an adaptive adversary (with keyword and trapdoor history) is considered. To predict a meaningful and a relevant information of query keyword, the adversary must perform a lot of operations in polynomial time and record the tremendous amount of data.

Description

The challenger $\mathscr{C}$ generates the encrypted table based on object keywords present in images from the image collection $\mathscr{I}$ called index table IT. The attacker $\mathscr{A}$ will choose an object keyword W to get the encrypted keyword and sends to the $\mathscr{C}$. The $\mathscr{C}$ generate the encrypted trapdoor of that object keyword. This encrypted trapdoor is forwarded back to the $\mathscr{A}$. The generation of keyword trapdoors continues until the $\mathscr{A}$ receives polynomial-many encrypted query keyword trapdoors. After the completion of this step, the $\mathscr{A}$ will send two distinct keywords *i.e.* W_0, W_1 and the $\mathscr{C}$ tosses a fair coin b. As a result, the $\mathscr{C}$ encrypts a keyword W_b and sends to the $\mathscr{A}$. The $\mathscr{A}$ is required to guess the output of b. At this point, if the $\mathscr{A}$ guess the correct output with probability great than 1/2 then the $\mathscr{A}$ wins otherwise the $\mathscr{C}$ wins. Here the security parameter λ is negligible.

Definition 3.1 Let SE be a searchable scheme with six phases (KeyGen, Obj_Det, Build_Obj_Index, Build_Query, Search_Outcome, Dec) with a security parameter λ, $\mathscr{I}$ be the set of images, $\mathscr{W}$ be the set of objects, and $\mathscr{A} = (\mathscr{A}_0, \mathscr{A}_1, ..., \mathscr{A}_{m+1})$ be the adversaries where $m \in \mathbb{N}$. Now we are considering to have a probabilistic experimental function $Index\_Trap_{SE,\mathscr{A}}(\lambda)$:

$$
\begin{array}{l}
Index\_Trap_{SE,\mathscr{A}}(\lambda) \\
\quad (K, k_s) \leftarrow KeyGen(\lambda) \\
\quad (IT) \leftarrow Build\_Obj\_Index(K, \mathscr{I}) \\
\quad for \;\; 1 \leq z \leq m \\
\quad\quad (st_{\mathscr{A}}, W_z) \leftarrow \mathscr{A}_z(st_{\mathscr{A}}, Q_{W_1}, ..., Q_{W_z}) \\
\quad\quad Q_{W_z} \leftarrow Build\_Query_K(W_z) \\
\quad b \xleftarrow{\$} \{0, 1\} \\
\quad (st_{\mathscr{A}}, W_0, W_1) \leftarrow \mathscr{A}_0(\lambda) \\
\quad (Q_{W_b}) \leftarrow Build\_Query(K, k_s, W_b, num) \\
\quad b' \leftarrow \mathscr{A}_{m+1}(st_{\mathscr{A}}, Q_{W_b}) \\
\quad (Q'_{W_z}) \leftarrow Build\_Query_K(W_p); \; p \in \mathbb{N} \\
\quad if \;\; b' = b, \; output \; 1 \\
\quad otherwise \;\; output \; 0
\end{array}
$$

To record the status of $\mathscr{A}$, $st_{\mathscr{A}}$ is used above. For the polynomial time $\mathscr{A}$, the keyword-trapdoor indistinguishability holds:

$$Pr[Keyword\_Trapdoor_{SE,\mathscr{A}}(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda) \tag{3.5.1}$$

Here the probability is dependent over b.

3.5.3.2 Trapdoor-Index Indistinguishability for SE

The complexity of an SE scheme is measured by the indistinguishability of trapdoor indexes. This means that the generated query trapdoors are random in such a way that no information is disclosed to the adversaries about keyword or index. The indistinguishability should remain in contact with the keyword even if the same keyword is searched repeatedly. The generated query trapdoor must preserve the users privacy and keyword security in terms of distinguishability and linkability with history (keyword, trapdoor, index) in adaptive adversarial model. Also, by changing a single bit or character in query keyword, the resultant Trapdoor and Index Table should be completely changed and same is true for reverse.

Description

The challenger $\mathscr{C}$ generates an encrypted index table IT based on object keywords present in images from the image collection $\mathscr{I}$. The $\mathscr{C}$ sends the entries of first row of IT, all object keywords W, and encrypted queries generated by W to the $\mathscr{A}$. The order of appearing keywords is maintained during sharing of keywords with $\mathscr{A}$. The

$\mathscr{A}$ chooses two keywords which he want to get the encrypted trapdoors *i.e.* W_0, W_1 and forwards to the $\mathscr{C}$. The $\mathscr{C}$ tosses a fair coin b and encrypt the object keyword based on the output of b. This encrypted keyword *i.e.* trapdoor is sent to the $\mathscr{A}$. The $\mathscr{A}$ is required to guess the correct object keyword *i.e.* the output of b. If the $\mathscr{A}$ guesses correctly with probability greater than 1/2 then $\mathscr{A}$ wins otherwise $\mathscr{C}$ wins and the SE scheme provides the trapdoor-index indistinguishability. Here the security parameter λ is considered small enough to be ignored.

Definition 3.2 Let SE be a searchable scheme with six phases (KeyGen, Obj_Det, Build_Obj_Index, Build_Query, Search_Outcome, Dec) with a security parameter λ, $\mathscr{I}$ be the set of images, $\mathscr{W}$ be the set of objects, and $\mathscr{A} = (\mathscr{A}_0, \mathscr{A}_1)$ be the adversaries. Now we are considering to have a probabilistic experimental function $Index\_Trap_{SE,\mathscr{A}}(\lambda)$:

$$
\begin{aligned}
& Index\_Trap_{SE,\mathscr{A}}(\lambda) \\
& \quad (K, k_s) \leftarrow KeyGen(\lambda) \\
& \quad (IT) \leftarrow Build\_Obj\_Index(K, \mathscr{I}) \\
& \quad for \;\; 1 \le z \le m; \text{ where } m \in \mathbb{N} \\
& \qquad let \;\; IT' = IT[0][z] \\
& \qquad let \;\; W = (W_1, W_2, ..., W_z) \\
& \qquad Q_{W_z} \leftarrow Build\_Query(K, k_s, W_z, num) \\
& \quad b \xleftarrow{\$} \{0,1\} \\
& \quad (st_{\mathscr{A}}, W_0, W_1) \leftarrow \mathscr{A}_0(st_{\mathscr{A}}, \lambda, W_m, IT', Q_{W_m}) \\
& \quad (Q_{W_b}) \leftarrow Build\_Query(K, k_s, W_b, num) \\
& \quad b' \leftarrow \mathscr{A}_1(st_{\mathscr{A}}, IT_{W_b}) \\
& \quad (Q_W') \leftarrow Build\_Query_K(W_p); \;\; p \in \mathbb{N}
\end{aligned}
$$

$$if \;\; b' = b, \; output \; 1$$

$$otherwise \;\; output \; 0$$

To record the status of $\mathscr{A}$, $st_{\mathscr{A}}$ is used above. For the polynomial time $\mathscr{A}$, the trapdoor-index indistinguishability holds:

$$Pr[Trapdoor\_Index_{SE,\mathscr{A}}(\lambda) = 1] \leq \frac{1}{2} + negl(\lambda) \tag{3.5.2}$$

Here the probability is dependent over b.

Theorem 3.1: The Privacy Preserving Search Over Encrypted Images (PPSEI) SE Scheme is secure if the generated index table IT is secure and query object trapdoors are probabilistic. The proposed PPSEI scheme is secure and exhibits the Keyword-Trapdoor Indistinguishability and Trapdoor-Index Indistinguishability.

3.6 Proposed Scheme

This section presents the Privacy Preserving Search Over Encrypted Images Searchable Encryption Scheme that consists of 6 phases. A detailed explanation and representation of each phase is given in this section. Following are the six phases involved in the proposed SE scheme:

1. Key Generation Phase **(KeyGen (λ) Phase)**
2. Object Detection and Identification Phase **(Obj_Det($\mathscr{I}$) Phase)**
3. Object (class based) Index Table Generation Phase **(Build_Obj_Index($K, \mathscr{I}, \mathscr{W}$) Phase)**

4. Query Generation Phase **(Build_Query(**K, k_s, W, num**) Phase)**

5. Searching Phase **(Search_Outcome (**IT, Q_W**) Phase)**

6. Decryption Phase **(Dec(**K, A**) Phase)**

3.6.1 Scheme Construction

These phases are explained in details here.

- **Phase 1-[**K, k_s**]←KeyGen(**λ**):** In key generation phase, master and session keys are generated by giving a security parameter as input argument to the key generation algorithm such that $K, k_s \leftarrow \{0,1\}^{\lambda}$. Furthermore, for single writer multiple reader (S/M) model *i.e.* multiple query users with a single owner of data, the session key for each user will be different.

- **Phase 2-**$\mathscr{W}$**←Obj_Det(**$\mathscr{I}$**):** In object detection and identification phase, images are given as input to the object detection algorithm *e.g.* YOLOv3, SSD, RetinaNet, *etc.* and get the array of objects $\mathscr{W}$, also called keywords, for each object in each image. If the data owner want to add more images to the database, this phase will be repeated each time. Manual entries can also be done by the data owner. In case of manual entries, this phase will not be processed.

- **Phase 3-**IT **←Build_Obj_Index(**$K, \mathscr{I}, \mathscr{W}$**):** In index table generation phase, the images, objects extracted from these images, and the frequencies of these objects are listed in this table. The encryption of each entity is also carried out in this phase.

- Initialize a 2D array IT of size $n \times m$; where m is the number of objects and n is the number of images in the database.
- For $1 \leq x \leq m$ and For $1 \leq y \leq n$
  * Compute and store: $IT[1][x] = H_K(W_x)$
  * Compute and store: $IT[y][1] = Enc_K(id(I_y))$
  * Calculate RF for each W_m in $\mathscr{I}$ with a Python code as given in Appendix A.
- For masking of RF, following equation is used:
  * For $1 \leq z \leq number\,of\,columns\,in\,IT$
    · $IT[:][z+1] = \log(IT[:][z+1]+2) \times R_1 + R_2$
    · R_1 and R_2 are random values. For each column, each random value must be unique to achieve the masking in relevance frequencies.
  * By using log and random values, original object frequencies are masked and it limits the leakage information about frequencies while keeping the ranking intact. The index table "IT" is generated and outsourced to the CS along with the encrypted images.

• **Phase 4-**Q_W **←Build_Query**(K, k_s, W, num)**:** For the query generation, following steps are performed. *num* is the number of images desired by the user.
- Compute $a = H_K(W)$
- Compute $b = Enc_{k_s}(W)$
- Compute $c = a \oplus b$ and $d = H(b)$

– Set query trapdoor $Q_W \leftarrow (c, d, num)$

- **Phase 5-**$A[\,]$ $\leftarrow$**Search_Outcome (**IT, Q_W**):** In the searching phase, following operations are performed. Later, the output is stored in the array $A[\,]$ as search outcomes.

  – Initialize a dynamic 2D array $A[\,]$.

  – For $1 < y \leq number\ of\ columns\ in\ IT$

    1. Set $a = IT[1][y]$;
    2. if $(d == H(a \oplus c))$:

       (a) For $1 \leq r \leq num$

       * Find the highest RF, return $Enc_K(id(I))$;

  – $A[\,] \leftarrow Enc_K(id(I_i))$ and return A to the client.

- **Phase 6-Dec(**K, A**):** Using the master key K, each image identifier in the array A can be decrypted. This will give the image identifiers in plaintext.

3.7 Discussion about Proposed Scheme

This sections gives the detailed analysis of each phase involved in proposed PPSIE scheme.

KeyGen Phase

In this phase, a master and session keys are generated by the user. The input argument of this phase include a security parameter λ and produces two keys; a session key k_s as

$k_s \in \{0,1\}^\lambda$ and a master key K as $K \in \{0,1\}^\lambda$. Both master and session keys are to be kept secret at the user end and there is no need to share any key with the *CS*. As, at the *CS* end, all operations are being performed without master and/or session key. Master key generation is a deterministic function while session key generation is a probabilistic algorithm. Both phases are executed by the data owner.

Obj_Det Phase

This phase performs the object detection operation. Any good object detection algorithm can be used depending upon the user requirements, computational resources, time, and budget. The object detection algorithm must be trained on images from user's customized image dataset or datesets available online. The user can use pre-trained object detection algorithms and can test his/her own images for object detection. For example, for the custom image dataset, the detection of most cancer effected body area, the object detection algorithm can be trained and images related to cancer effected areas can be used. Also, for military operation areas, the object detection algorithm can be configured and trained as to identify the areas with more number of troops, military vehicles, and air crafts *etc.* In simple words, this phase is generic and can be modified according to the use case. This phase will give objects present in images. The data owner executes this phase which is deterministic in nature.

Build_Obj_Index Phase

In this phase a dynamic 2D array IT is initialized. The cryptographic hash function is used by the client as:

$$H : \{0,1\}^{\lambda} \times W \rightarrow \mathbb{Z}_p$$

The object keyword is hashed using the master key K. The index table IT can be categorised in to 3 sections. The first row of IT is the keyed hash values of all objects as identified in phase 2. The first column of IT contains the encrypted identifiers of images *i.e.* $Enc_K(id(I_n))$. And the remaining entries of IT are the relevance frequencies of objects $\mathscr{W}$ in images $\mathscr{I}$. Each column of IT is the associated frequency of object W which are calculated with a Python code. Later, we took the log of RF, multiplied, and added random values R_1 and R_2 respectively. The random numbers are obtained from CSPRNG to mask the original values. This way of masking the frequencies maintains the order of frequencies across the same column but values across multiple columns are different. In IT, many entries can be zero. We have added a constant value "2" to the RF before taking log and then added a random number also multiplied with second random to reduce the frequency analysis attack. This will also resist the information disclosure about size of images and number of objects present in each image. The data owner executes this phase which is deterministic in nature.

Build_Query Phase

In this phase, the user generates the encrypted query to search for images based on image content *i.e.* objects. To generate a query Q_W, the user first encrypts the query keyword by computing the cryptographic keyed hash function and store it to a parameter a. Also, the user computes another parameter b by encrypting the same keyword with probabilistic symmetric encryption by using AES in CBC mod and a session key k_s. Another parameter c is computed by computing the XOR operation of parameters a and b. Then, another parameter d is computed by taking a simple cryptographic hash of parameter b. The computed parameters are then set to a user query parameter Q_W which contains the parameters c, d and the desired number of images denoted as num. This query parameter Q_W is then sent to the CS. The data owner executes this phase each time when a query is made. This phase is probabilistic in nature.

Search_Outcome Phase

When the CS receives the query request Q_W from the user containing three parameters c, d, num, it will try to find the column entry in index table IT based on the condition when $d == H(a \oplus c)$. Where a is the values of first row of index table IT as encrypted object keywords. When the condition holds true, as mentioned before, the CS returns the num number of encrypted image identifiers based on relevance frequency in ranked order. The CS executes this phase which is deterministic in nature.

Dec Phase

When the user receives the encrypted image identifiers in ranked order, he will use his master key K to decrypt these image identifiers. The data owner executes this phase which is deterministic in nature.

3.8 Dynamic Database

To update the database, the user will need to perform few steps. Index table entries can be modified by adding or deleting image identifiers from it. The deletion of images is relatively easy and quick process while the addition of new images requires more time and resources than deletion of images. These processes are explained in detail below:

3.8.1 Addition of new Images

If a user is interested to add n number of new images to the database, the CS will only append N new rows at the end of existing IT. While at the user end, phase 2 and few steps from phase 3 are repeated *i.e.* the phase 2 will be repeated entirely to identify the objects present in images and object frequencies (RF), are calculated. Later, in phase 3, a 2D array AIT (Addition in Index Table Array) of size $n \times m$ is initialized. The data owner is interested to add n new images to the database while m represents the columns equals to the object keywords present in $\mathscr{W}$.

The user will encrypt the image identifiers as $Enc_K(id(I_n))$ and stores in the first column of the AIT. The relevance frequencies are masked with the same random numbers as was used to generate the original IT. These masked values are stored in the rest of the respective locations of AIT. At this point, we do not need to recompute the hashes of object keywords as was calculated and stored in first row of IT. The random values for masking of RF must be the same as was used earlier. This will keep the ranking in right order. If the object detection algorithm in phase 2 is changed, the user need to regenerate the original index table IT from scratch as it may change the number of columns, keywords, and RF values.

Remarks: To avoid the frequency analysis attack, we masked the original relevance frequencies by taking a log of original frequency value, adding and multiplying the random values. This step ensures and keeps the effective and efficient ranked searching intact. From the results shown in Chapter 4, it is clear that this type of masking can helps the users to maintain the privacy in terms of size of images and exact number of objects in those images. The attacker can not predict the original frequency value by analyzing the masked values. To increase the security of these indexes, there are alternative options are available in literature *i.e.* Order Preserving Encryption (OPE) [92] or Order Preserving Hashing (OPH) [93] can be used.

3.8.2 Deletion of Images

For the deletion of one or multiple images, only few steps of phase 3 are required *i.e.* only encryption of image identifiers $Enc_K(id(I_n))$ will be performed at the user end. These encrypted image identifiers are then sent to the CS. The CS will delete all those rows in IT containing the image identifiers sent by the user and also deletes the images from the database. It is assumed that the CS will perform this operation honestly. According to the threat model, the server is trusted but curious which means the server is interested for content of the images and the index table IT while it performs its operations successfully.

3.9 Dynamic Queries

The proposed scheme supports 4 types of search queries. These includes single object keyword query, multi object keyword query, single object picture query, and multi object picture query.

Single object keyword query includes one keyword at a time. The user enters an object keyword and the encrypted query is generated using the probabilistic encryption through Build_Query phase. This encrypted query is then sent to the *CS*. Upon receiving the request from the user, the *CS* will execute the searching algorithm for query keyword from the *IT* and if query keyword is found, required number of image identifiers are returned to the user.

For multi-object keyword query, user enters more than one keyword and enters the number of images required. Each query keyword is processed separately by the Build_Query phase. The encrypted query trapdoors are generated probabilistically and sent to the *CS*. The *CS* performs searching for each query keyword and try to fetch the images that contain all objects in images. As the index table entries are either encrypted, hashed, and/or masked, the cloud server cannot predict and search accurately. The *CS* will identify the image identifiers for each query object keyword and store them in an array. The best way to identify an image identifier, with all query keywords, is to find the sum of all *RF* values and return the image identifiers with maximum value of *RF* to the user. For example, if the user have set the value for $num = 3$ and there are 3 keywords in the multi-object query, 3 encrypted query keywords will be sent to the *CS*. The *CS* will process each query keyword separately and store the *num* number of image identifiers for each keyword in a new array. At this stage the *CS* will have the array of size 9×4.

Where 9 is the number of rows with first column of image identifiers and 4 is the number of columns with 3 columns of *RF* values. The *CS* will calculate the sum of these 3 columns and store them in 5th column. The *CS* will return the *num* number of image identifiers based on the maximum sum of *RF* values.

For single object picture query, user inputs an image with single object to the algorithm. The algorithm first identifies the object class and then encrypts the query probabilistically. The object keyword trapdoor is forwarded to the *CS*. The server deals this query as normal single object query and finds the best relevant image identifiers based on the object frequency and returns required number of identifiers to the user.

For multi-object picture query, the user enters an image with multiple objects to the algorithm. The algorithm identifies all the objects present in image and query generation function is called. Each keyword is encrypted separately and probabilistically. The cloud server deals this query in the same way as multi-object keyword query and returns required number of images to the user. The user decrypts the image identifiers by the help of Dec function and gets the required image identifiers in plain text.

3.10 Correctness and Soundness

This section gives the proof of correctness and soundness of proposed SE scheme. Section 3.5.2 defines the design goals of an SE scheme. We will go through the proposed scheme and will verify the correctness and soundness. The PPSEI technique is comprises of six polynomial time phases *i.e.* Π = (KeyGen, Obj_Det, Build_Obj_Index, Build_Query, Search_Outcome, Dec).

Correctness: The proposed PPSEI is correct if for the security parameter λ, the master key K and the session key k_s generated by $KeyGen(\lambda)$, for the *objects* detected and identified by $Obj\_Det(\mathscr{I})$, for IT generated by $Build\_Obj\_Index(K,\mathscr{I},\mathscr{W})$, the search using the query keyword Q_W generated in $Build\_Query(K,k_s,W,num)$ and search query processed by $Search\_Outcome$ always gives the correct encrypted image identifiers set $A = Enc_K(id(I_n))$ in ranked order. The proposed PPSEI technique is correct for the following equation:

- The following results hods true if $W \in I_i$ with the overwhelming probability:

$$Search\_Outcome(IT,Q_W) = \mathscr{I} \cap Dec(K,A) = I_i,\ where\ 1 \leq i \leq n \qquad (3.10.1)$$

- The following results hods true if $W \notin I_i$ with the overwhelming probability:

$$Search\_Outcome(IT,Q_W) = \mathscr{I} \cap Dec(K,A) = 0 \qquad (3.10.2)$$

Soundness: The proposed PPSEI is sound if for the security parameter λ, the master key K and the session key k_s generated by $KeyGen(\lambda)$, f or the *objects* detected and identified by $Obj\_Det(\mathscr{I})$, for IT generated by $Build\_Obj\_Index(K,\mathscr{I},\mathscr{W})$, the search using the query keyword Q_W generated in $Build\_Query(K,k_s,W,num)$ and search query processed by $Search\_Outcome$ should not generate the false positives and always gives the sound results. The proposed PPSEI technique is sound for the following equation:

- The following results hods true if $W \in I_i$ with the overwhelming probability:

$$Search\_Outcome(IT, Q_W) = 1 \tag{3.10.3}$$

- The following results hods true if $W \notin I_i$ with the overwhelming probability:

$$Search\_Outcome(IT, Q_W) = 0 \tag{3.10.4}$$

Lets we have master key K and a session key k_s such that $K, k_s \in \{0,1\}^{\lambda}$ generated by $KeyGen(\lambda)$ phase. Given $W, W^{'} \in \mathscr{W}$, the following should holds true:

- Given $Q_W = Build\_Query(K, k_s, W, num)$, the following equality holds true with an overwhelming probability:

$$Q_W = \left\{ \begin{matrix} (H_K(W)) \oplus (Enc_{k_s}(W)), \\ H(Enc_{k_s}(W)), num \end{matrix} \right\} \tag{3.10.5}$$

- Given $Q_W = Build\_Query(K, k_s, W^{'}, num)$, and $W \neq W^{'}$, with an overwhelming probability the following equality holds true:

$$Q_W \neq \left\{ \begin{matrix} (H_K(W^{'})) \oplus (Enc_{k_s}(W^{'})), \\ H(Enc_{k_s}(W^{'})), num \end{matrix} \right\} \tag{3.10.6}$$

We can clearly state that this inequality condition holds true with only if $H_K(W) = H_K(W^{'})$ which is having a negligible probability.

From the above results, we can clearly state that if the object query keyword is unique then it will result a distinct object keyword in the IT. As, the IT contains the encrypted image identifiers $Enc_K(id(I_n))$ for each image in database $\mathscr{I}$, the result computed by the $Search\_Outcome$ Phase comply with the correctness and soundness as explained in equations 3.10.1, 3.10.2, 3.10.3, and 3.10.4. Hence, the proposed SE scheme is correct and sound.

3.11 Summary

In this chapter, we briefly discussed about the current available image processing techniques, threat model and assumptions about the proposed SE scheme. System model for the proposed scheme is presented in this chapter. Probabilistic encryption and security definitions are explained in detail. We have presented a novel Privacy Preserving Search over Encrypted Images (PPSEI) scheme that provides ranked searching and preservers the clients privacy in terms of search pattern leakage attacks. Dynamic database and dynamic query scenarios are explained. The correctness and soundness of the proposed scheme is discussed in this chapter. Chapter 4 discusses the security analysis of the proposed PPSEI scheme.

Chapter 4

Security Analysis

As explained in Chapter 2, previous image-based SE schemes are supporting deterministic search queries and leak information about the user activity, repeated keyword search frequency, and leads to the traceability attacks. The proposed SE scheme is based on probabilistic query generation which resists the traceability issue. We will review the leakage profiles that will show the amount of information leaked in the proposed SE scheme. We will map the proposed scheme with the security definitions as defined in Section 3.5.3. Later, we will review the formal game based security analysis. The security analysis of proposed scheme with relevant techniques are given at the end of this chapter.

4.1 Security Evaluation of Proposed Scheme

This section analyses the information leakage profiles of the proposed SE technique. Ideally, the SE scheme should not reveal any data or the information that leaks the user privacy, outsourced data and the statistical data about the number of search queries made over data. A cryptographic protocol is considered secure if it preserve the user privacy about data, access and search patterns. The proposed PPSEI is secure in terms of search pattern and preserving user's privacy. Some leakage profiles are given here.

4.1.1 Leakage Profiles

Leakage profile defines the amount of information leaked to the adversary. This information can be encrypted or unencrypted, significant or insignificant. We will analyze the information obtained from the six polynomial time algorithms of proposed SE scheme to the *CS* *i.e.* index table IT, query trapdoor Q_W, and search outcome. The adversary $\mathscr{A}$ can launch any possible attack in standard model. This is because we are not restricting $\mathscr{A}$ to replace the SE technique with different weak construction. The only restriction applies to the time of execution of different process and steps by the $\mathscr{A}$ *i.e.* the $\mathscr{A}$ is restricted to polynomial time and the following outcomes are observed by the security analysis of PPSEI scheme:

Leakage $L_{4.1}$

Description: This leakage profile $L_{4.1}$ is linked with IT. This IT is shared and revealed to all parties including the user, the CS, and the attacker $\mathscr{A}$. $L_{4.1}$ is defined as:

$$L_{4.1}(IT) = \left\{ \begin{array}{c} (H_K(W_m)), Enc_K(id(I_n)), Mask(RF) \\ Enc_K(\mathscr{W}) \in Enc_K(\mathscr{I}) \vee Enc_K(\mathscr{W}) \notin Enc_K(\mathscr{I}) \end{array} \right\} \tag{4.1.1}$$

Leakage $L_{4.2}$

Description: The leakage profile $L_{4.2}$ is linked with the keyword query Q_W that is generated for a specific object keyword W and is shared to all parties including the user, the CS, and the attacker $\mathscr{A}$. Q_W is generated by the user. $L_{4.2}$ is defined as:

$$L_{4.2}(Q_W) = \begin{Bmatrix} a \leftarrow H_K(W_i) \oplus Enc_{k_s}(W_i), \\ b \leftarrow H(Enc_{k_s}(W_i)), num \end{Bmatrix} \tag{4.1.2}$$

Leakage $L_{4.3}$

Description: This leakage profile $L_{4.3}$ is linked with the outcome of search for a specific query keyword generated against specific object W. The search outcomes (SO) are generated at the CS and it is assumed that these outcomes are revealed to all parties including the user, the CS, and the attacker $\mathscr{A}$. Let OC be the outcomes corresponding to the search object W. $L_{4.3}$ is denoted as:

$$L_{4.3}(SO) = \Big\{ OC(W), Enc_K(id(I_i)) \forall Q_W \in \mathscr{I} \Big\} \tag{4.1.3}$$

Discussion on Leakage:

In the proposed scheme, the search query are generated probabilistically. Furthermore these random queries are hashed with the user's master key. Therefore, the leakage associated with search query is meaningless and we can ignore it. During the query generation phase, the algorithm uses session key for encrypting the object keyword and the session key is generated randomly each time a key generation function is called.

This makes the query probabilistic. In other words, if somehow the attacker has access to the query generation process accidentally, the queries generated in future with this function are still secure. This is because each generated query is independent. This way, the PPSEI scheme is secure in terms of search pattern. The access pattern are not prevented by our scheme. The access pattern tells about the images that are accessed as a result of a successful search.

Regarding the relevance frequencies of objects present in each image are masked with random numbers. As discussed earlier, to achieve a high and better security of relevance frequencies, order preserving hashing (OPH) can be employed. The masking technique either by random numbers or OPH may reveal the information about the presence or absence of a keyword in images. This leakage about relevance frequencies is only related to the relevance frequencies and does not affect the query trapdoor unlinkability and indistinguishability.

From the above analysis, we can say that leakages $L_{4.1}$ (given in equation 4.1.1) and $L_{4.3}$ (given in equation 4.1.3) might lead to the user's privacy and data security issues. However, through the formal security analysis, we have explained that such leakages do not leak any data related to outsourced data to the cloud. The assumptions and leakages discussed here are interrelated and interdependent to each others. Therefore, to achieve the best level of security, it is assumed that all the security assumptions are followed strictly.

4.2 Formal Security Analysis

Through the game-based formal security analysis of proposed content-based searchable encryption scheme over encrypted images (PPSEI) is given in this section.

Lemma 4.1. The proposed Privacy Preserving Searchable Scheme over Encrypted Images presented in Chapter 3 is "privacy preserving". According to the security definitions presented in 3.5.3.1 and 3.5.3.2 the proposed scheme is $L_{4.1}, L_{4.2}$ and $L_{4.3}$ secure. Here $L_{4.1}$ represents the leakage associated with index table IT which can leak the information about hashed object keywords, encrypted image identifiers, masked RF values, and the presence / absence of an encrypted object keywords within an encrypted image. While $L_{4.2}$ is representing the leakage linked to the query object keyword trapdoor that leaks the information about parameters a, b and required number of images. $L_{4.3}$ is representing the search outcomes from the SE scheme and is associated with the outcomes of Search_Outcome Phase. This leaks the information of encrypted image identifiers resulting from the search outcomes.

Proof:

As stated in Theorem 3.5.3.2, the proposed PPSEI technique is resistive to multiple attacks and secure if the generated index table IT is secure and query object trapdoor Q_W is probabilistic. To prove the stated lemma and the SE scheme is accordance with Theorem 3.1, simulation of security definitions *i.e.* keyword-trapdoor indistinguishability and trapdoor-index indistinguishability definitions is done. Two entities *i.e.* the attacker $\mathscr{A}$ and the challenger $\mathscr{C}$ are required for this poof. If the attacker $\mathscr{A}$ is unable to distinguish between object keywords, their object query trapdoors and the search outcomes from the algorithm, then the proposed SE scheme is secure in terms of privacy-preserving for the user's.

To verify the proposed PPSEI scheme, we have used the game-based approach same as used by [52]. The security proof is categorises into three different parts including setup, challenge and outcome phase. We will revisit the security definitions in terms of game-based security analysis.

4.2.1 Keyword-Trapdoor Indistinguishability in PPSEI Scheme

For the security analysis of proposed PPSEI technique and to check the keyword-trapdoor indistinguishability, the game is played between the challenger $\mathscr{C}$ and the attacker $\mathscr{A}$. Let we have "n" images in the dataset as $\mathscr{I} = \{I_1, I_2, ..., I_n\}$ with "m" number of object in images as $\mathscr{W} = \{W_1, W_2, ..., W_m\}$, where $n, m \in \mathbb{N}$ are associated with the index table IT. This game is further categorises in to three different phases as given below:

1. **Setup Phase:** This step starts from the attacker $\mathscr{A}$. $\mathscr{A}$ forwards the object keyword to the challenger $\mathscr{C}$. In response $\mathscr{C}$ sends the encrypted query trapdoor to $\mathscr{A}$. This phase continues till $\mathscr{A}$ get the responses of all object query keywords and stores the history of all encrypted queried trapdoors with respect to their object keywords.

2. **Challenge Phase:** This step initiated by the attacker $\mathscr{A}$. $\mathscr{A}$ chooses two object query keywords $W_0^{'}, W_1^{'} \in \mathscr{W}$ and forwards to $\mathscr{C}$. The attacker $\mathscr{A}$ can choose these

object keywords as if the $\mathscr{A}$ want to search for a unique object keyword such that $W_0^{'} \neq W_1^{'}$. In response to this, the challenger $\mathscr{C}$ selects a queried object keyword based on a fair coin toss such that $b \leftarrow \{0,1\}$. $\mathscr{C}$ then generates the encrypted keyword trapdoor and sends back to $\mathscr{A}$. This query trapdoor is generated with the selection of value of b *i.e.* $Q_{W_b^{'}}$. After this challenge phase, the $\mathscr{A}$ can query more object keywords and can rerun the previous phase *i.e.* setup phase. $\mathscr{A}$ can make queries for the same keywords again, as done in challenge phase, if interested.

3. **Outcome Phase:** After the completion of challenge phase, the attacker have to guess the the output of $b^{'} \in \{0,1\}$ based on the object trapdoor $Q^{'}_{W_b^{'}}$. If $\mathscr{A}$ guess the output as $b^{'} = b$ then the attacker wins. We can rephrase this in other words as in the polynomial time the attacker must have to return the object keyword $W_b^{'}$ with respect to the query trapdoor $Q^{'}_{W_b^{'}}$ to the challenger $\mathscr{C}$. If the attacker $\mathscr{A}$ guesses the correct keyword then $\mathscr{A}$ wins. If not, the challenger $\mathscr{C}$ wins and the proposed SE scheme provides keyword-trapdoor indistinguishability.

As the object keyword trapdoors are generated based on probabilistic encryption and every time the encrypted trapdoor is unique, the probability for guessing the correct outcome of $\mathscr{A}$ is 1/2. As with this probability of 1/2 and it is inline with the security definitions 3.5.3.1 and equation 3.5.1. From these results the challenger wins and the proposed PPSEI technique provides the query keyword indistinguishability.

4.2.2 Trapdoor-Index Indistinguishability in PPSEI Scheme

For the security analysis of proposed PPSEI technique and to check the trapdoor-index indistinguishability, the game is played between the challenger $\mathscr{C}$ and the attacker $\mathscr{A}$. Let we have "n" images in the dataset as $\mathscr{I} = \{I_1, I_2, ..., I_n\}$ with "m" number of object in images as $\mathscr{W} = \{W_1, W_2, ..., W_m\}$, where $n, m \in \mathbb{N}$ are associated with the index table IT. This game is further categorises in to three different phases as given below:

1. **Setup Phase:** In this phase, an index table IT is generated by the challenger $\mathscr{C}$ corresponding to the images. $\mathscr{C}$ sends all the relevant information to $\mathscr{A}$ including the IT, all query keyword trapdoors, and entries of IT corresponding to query keywords along with the keywords.

2. **Challenge Phase:** This step is initiated by the attacker $\mathscr{A}$. $\mathscr{A}$ chooses two object query keywords $W_0^{'}, W_1^{'} \in \mathscr{W}$ and forwards to $\mathscr{C}$. The attacker $\mathscr{A}$ can choose these object keywords as if the $\mathscr{A}$ want to search for a unique object keyword such that $W_0^{'} \neq W_1^{'}$. In response to this, the challenger $\mathscr{C}$ selects a queried object keyword based on a fair coin toss such that $b \leftarrow \{0, 1\}$. $\mathscr{C}$ then generates the encrypted keyword trapdoor and sends back to $\mathscr{A}$. This query trapdoor is generated with the selection of value of b *i.e.* $Q_{W_b^{'}}$. After this challenge phase, the $\mathscr{A}$ is gives access to the data provided in setup phase *i.e.* the previous generated history.

3. **Outcome Phase:** After the successful completion of challenge phase, the $\mathscr{A}$ have access to the generated keyword trapdoor $Q^{'}_{W_b^{'}}$. $\mathscr{A}$ is now required to provide the entry of IT corresponding to the $Q^{'}_{W_b^{'}}$ in polynomial time. The $\mathscr{C}$ wins if the $\mathscr{A}$ can not make correct guess in polynomial time. If the $\mathscr{C}$ wins then the proposed

SE scheme provides trapdoor-index indistinguishability.

As the query keyword trapdoors are being generated with probabilistic encryption and it is random each time, therefore, the $\mathscr{A}$ having the history and current keyword trapdoor, can not make a successful guess or can be successful with the probability of 1/2. Therefore, we can conclude that this is inline with the definition of trapdoor-index indistinguishability as defined in 3.5.3.2 and given in equation 3.5.2.

To prove the defined theorem 3.5.3.2 we follow the corollary defined by [52]:

Corollary 4.1: From the above analysis, we can state that the keyword-trapdoor and trapdoor-index indistinguishability leads us to the result to the proposed privacy preserving search over encrypted images (PPSEI) technique.

Proof: Let we have an SE scheme with six phases *i.e.* KeyGen Phase (for key generation), Obj_Det Phase (for detecting objects in images), Build_Obj_Index Phase (for generation of index table from identified objects), Build_Query Phase (for query generation to search an image(s)), Search_Outcome Phase (searching functionality provided by CSPs), and Dec Phase (conversion of ciphered identifiers in to plaintext image identifiers).

To prove the proposed PPSEI scheme as *"Privacy Preserving"*, we say that the PPSEI provides trapdoor-index & keyword-trapdoor indistinguishability as it is based on probabilistic encryption of query keywords. In the index table, the image identifiers and object keywords are secure with cryptographic functions and probabilistic query trapdoors points to a specific index location each time, this process maintains privacy. The probabilistic encryption of object keywords leads to the privacy preservation of all entities involved in SE scheme. As a result the client's privacy is preserved.

To verify the security of PPSEI technique against $L_{4.1}, L_{4.2}$, and $L_{4.3}$ leakage profiles, as presented in equations 4.1.1, 4.1.2, and 4.1.3 respectively, we take the probabilistic nature of query trapdoors in to account. As already discussed, due to the probabilistic query generation these leakages are not affecting or reducing the security of PPSEI technique and these leakages are meaningless. The PPSEI technique is based on secure cryptographic primitives *i.e.* one way hash function, AES encryption, XOR, *etc.* therefore, our scheme provides a good security. The proposed scheme, given in Section 3.6, consists of six phases as KeyGen Phase produces two keys $(K, k_s) \leftarrow KeyGen(\lambda)$. The Obj_Det($\mathscr{I}$) Phase will extracts the objects present in images, the Build_Obj_Index($K, \mathscr{I}, \mathscr{W}$) Phase produces an encrypted index table IT based on the objects extracted in Obj_Det phase. The Build_Query(K, k_s, W, num) Phase generates the probabilistic query keyword trapdoors of keyword W. The Search_Outcome (IT, Q_W) Phase performed at the cloud server, returns the outcomes of the search. As discussed already, the PPSEI technique is based on probabilistically generated object queries that results the indeterminisitic trapdoors even if the same object keyword is searched repeatedly. The attacker can not make a relation or difficult to link the query object keyword and trapdoor or create a connection among the keyword, trapdoor, and IT before searching. This also applies to an attacker who maintains a search and results history. Definitions 3.5.3.1 and 3.5.3.2 are therefore met.

Now, if we look at the leakages, we can say that either these leakages are encrypted, hashed, or masked. If the user's master key K is kept secure, the adversary can not recompute the keyed hash. In other words, given the hash value, the attacker is unable to extract the plaintext from that hashed value. In polynomial time, there is no chance of getting any information from the probabilistically encrypted object query trapdoors. This encryption leads to the problem of integer factorization problem and it is a hard problem. As a result, the PPSEI technique is secure against all leakages ($L_{4.1}, L_{4.2}, L_{4.3}$) for adaptive/non-adaptive adversaries. The proposed PPSEI technique provides the keyword-trapdoor indistinguishability and trapdoor-index indistinguishability.

4.2.3 Randomization Testing of Repeated Query Keyword

As discussed, the proposed PPSEI technique is based on probabilistic generation of query keyword trapdoors even if the same keyword is being searched multiple times. The query generation involves the encryption and hashing of query keywords that is employed with HMAC and AES in CBC mode. The AES in CBC mode needs 3 input parameters including the user's private key, initialization vector (*iv*) and the keyword. By keeping the *iv* and key random each time, the generated keyword is random. To test the probabilistic encrypted experimentally, we have generated trapdoors for keyword *"person"*. Due to the limitation of *pyexcel* library, only 65000 rows were allowed to write in excel file. We applied filter on duplicate values in excel file and found that there is no repeated query trapdoor was found. These results shows that the generated query trapdoors are random even if the same keyword is being searched multiple times.

4.2.4 Comparative Analysis

The security comparative analysis of PPSEI technique with other scheme present in literature are given in this section. Chapter 2 discusses many SE techniques over encrypted images that are present in literature [53, 54, 57, 58, 62, 94–96]. These schemes do not provide enough security to the user to prevent from security attacks. Almost all of the schemes discussed are prone to search pattern leakage attacks that leads to the user traceability issue. The security comparison is given in Table 2.2.

The proposed PPSEI technique is based on the indeterministic generation of object query trapdoors which resists the users from traceability issues. As, the query generation involves the keyed hashed value of query keyword and probabilistic encryption of the same keyword is utilized. The AES in CBC mode provides this functionality of randomness. If the same object keyword is being searched repeatedly, the generated encrypted query trapdoors are random each time. This resists the search pattern leakage attack and preserves the user's privacy in terms of user's query trapdoors traceability attacks. User can search any keyword without being traced.

4.3 Summary

This chapter focused on the security evaluation and analysis of the proposed privacy preserving searchable encryption technique for encrypted images (PPSEI). Security leakages are discussed and formally verified that the proposed SE scheme resists the all knows security attacks. Furthermore, the proposed scheme is verified against the security definitions 3.5.3.1 and 3.5.3.2. The comparative analysis given in Table 2.2

represents the security of proposed technique against search pattern leakage attacks. In Chapter 5 we will discuss the performance analysis of proposed scheme in details.

Chapter 5

Performance Analysis

5.1 Overview

This chapter discusses the performance and computational analysis of the proposed PPSEI technique. The performance analysis is divided in different parts *i.e.* first the algorithmic analysis of PPSEI is presented. Then the storage overhead and computational analysis is given in detail. The implementation of proposed scheme, system and dataset specifications are given in computation analysis section. Furthermore, the implementation is done in Python programming language and pseudo codes are given in Appendix A. The computational overhead of each phase of the proposed PPSEI scheme is given in details.

5.2 Algorithmic Analysis

To check the performance of the proposed PPSEI scheme, multiple analysis are presented and discussed. The time complexity analysis of PPSEI is drawn in this section. The complexity is based on the number of images present in collection of image dataset $\mathscr{I}$ denoted by n, number of object keywords present in images as denoted by m. The number of object keywords depends on the Object Detection algorithm and the trained model. For testing purposes, we have used YOLO v3 object detection algorithm trained on COCO image dataset. There are 80 common object keywords in COCO image dataset. The complexity of hash function is represented by "h" and complexity of encryption function is represented by "e". The proposed scheme is content-based image searching and retrieval scheme and consists of 6 phases including KeyGen, Obj_Det, Build_Obj_Index, Build_Query, Search_Outcome, and Dec phase. The complexity analysis of these phases are given as follows:

The complexity of the schemes are denoted by $O()$, read as *"big oh"*. This is called the asymptotic upper bound. It tells the time complexity required to run the code when the size and number of input parameters increases. It is a relationship among input parameters and the required time to process those input parameters. In the case of proposed scheme, KeyGen and Dec phases are fairly constant functions and requires the same amount of time. Therefore, the time complexity for KeyGen and Dec functions is $O(1)$

In the Obj_Det phase, each image is fed to the object detection algorithm and it detects the objects present in that image. As, our scheme is independent of the selection of object detection algorithm (*i.e.* the user can choose the object detection algorithm according to their requirements and use case scenarios) therefore the complexity can not be clearly defined for this phase. However, for the case of YOLO v3 object detection algorithm, the function involves the initialization of pre-trained model and coco names. The initialization phase is linear and independent from number of images. The object detection function's complexity depends on the loops used. For n number of images, the algorithm takes almost linear time to detect objects. Therefore, its complexity is $O(n)$.

For the Build_Obj_Index algorithm, the function takes an index table as input parameter and gives the ciphered index table IT. The generation of index table depends on the Obj_Det phase and the dataset used. For the case of COCO image dataset, there are 80 object classes and each image is analyzed against those 80 objects. Each image can contain objects from these 80 classes. Therefore, the number of columns for IT are always fixed while the number of rows are equal to the number of images present in dataset at the user end. The IT generation algorithm involves the AES encryption and $Hash$ functions which gives the linear time complexity. In conclusion, if the number of images increased, time complexity increases linearly. Therefore, the complexity of Build_Obj_Index phase is $O(n)$.

In the Build_Query function, the user gives an object keyword, to the query generation algorithm, to search for images. Another input parameter includes the number of required images. The Build_Query function uses 2 $Hash$ functions and one AES encryption function along with one XOR function. The time complexity for each is constant individually. If the number of keywords increases *i.e.* in the case of batch query or image query, the number of objects can be more than one, the time complexity will be constant *i.e.* $O(2h+e)$.

For the Search_Outcome function, simple XOR and one $Hash$ function is involved. As the keyword is searched from 80 columns (in case of COCO trained model), the time complexity will be the same for each encrypted query. Therefore, we can conclude that the time complexity for Search_Outcome function is $O(n_q)$, here "n_q" denotes the number of query keywords. Higher the number of query keywords, more time it will take to process each query. As a result, the function returns the *num* number of images based on the object frequency.

The algorithmic complexity of proposed scheme along with other schemes present in literature are presented in Table 5.1.

Table 5.1: Algorithmic Analysis of Proposed Scheme

Schemes	Feature extraction / Object detection	$Build\_Obj\_Index$	$Build\_Query$	$Search\_Outcome$
EPCBIR [56]	$O(n)$	$O(n)$	$O(1)$	$O(2n)$
PPCBIR [55]	$O(n)$	$O(n)$	$O(1)$	$O(2n)$
SCBIR [67]	$O(n)$	$O(2n)$	$O(1)$	$O(n)$
SEISA [65]	$O(n)$	$O(mn)$	$O(1)$	$O(n)$
PIC [66]	$O(n)$	$O(mn)$	$O(n)$	$O(2mn)$
EPIRM [97]	$O(n)$	$O(2mn)$	$O(1)$	$O(2mn)$
Our	$O(n)$	$O(n)$	$O(2h+e)$	$O(n_q)$

5.3 Storage Overhead

The proposed PPSEI technique presented in Section 3.6 is consists of 6 phases. The KeyGen phase generated 2 keys *i.e.* the masker key K and a session key k_s. Session key is generated at run time during Build_Query phase and the user do not need to store session key. Only master key is stored. Master key is 256 bit in length. So, the client stores (256 bits) / 8 = 32 Bytes. In the object detection phase, user need to store all the object detection algorithm related files. For the case of YOLO v3 pre-trained algorithm, user need to store three files named "coco.names" of size 705 bytes, "yolov3.cfg" of size 8,140 Bytes, and "yolov3.weights" of size 236,000,000 Bytes. For different object detection algorithm, these files can be of different sizes and names depending upon the selection of algorithm. During the index table generation phase, user need to mask the relevance frequencies. The random numbers through CSPRNGs are generated and stored at the user end. For the testing purposes, we have generated 180 random numbers and stored in an .xls file. The size of random number file is 5,500 Bytes. In case of a .txt file, this space is only 962 Bytes. Thus, the client requires total storage space as: 32 + 705 + 8,140 + 236,000,000 + 962 = 236,009,839 Bytes or 236.009839 Mega Bytes.

At the *CS* side, the *CS* stores encrypted index table *IT* and encrypted images in the database. Let we have the average size of an encrypted image as I_{avg} and n number of images, then the storage overhead for images would be $n.I_{avg}$. The size of encrypted images is the same as of plain images. Furthermore, the *CS* stores *IT* of size mn, here m represents the columns and n represents the rows of IT. For the case of YOLO v3 trained on COCO image dataset have $m = 80$. The storage overhead of *IT* is $8(mn)$ bytes. The storage overhead can be calculated as $8(mn) + n.I_{avg}$ at the *CS* end.

5.4 Computational Analysis

The proposed PPSEI is also tested and analysed based on the implementation and testing. This section gives the implementation and performance analysis in details. The scheme is implemented in Pyhton language and tested on COCO image dataset. Before going in to the details of the computation cost analysis, some preliminary details about system specifications and image dataset information are given.

5.4.1 System Specification

The implementation of PPSEI was done on a Ubuntu virtual machine using Python3 language. For the running cost analysis and representation of data in form of graphs, the MS Excel 365 is used. Further details about system specifications are given in Table 5.2:

Table 5.2: System Specification

Component	Specification
Operating System	Ubuntu 18.04.4 LTS
Memory	3.8 GiB
Processor	Intel Core i3-4010U CPU AT 1.70GHz $\times$ 3
OS type	64-bit
Virtualization	VMware Workstation 15.5.6 Pro

The programming required some pre-requisite libraries. The library details are given in Table 5.3.

Table 5.3: Program Library Specification

Library name	Version
Python	3.6.9
PyCharm Community	2020.1
Open CV	3.2.0
numpy	1.17.0
Pycryptodomex	3.9.7
Pyexcel	0.6.1

5.4.2 Dataset Specification

As discussed in Section 3.2.3, MS COCO dataset [74] is most widely accepted and used feature rich image dataset with more than 330K images, 250K people with key points, five captions in each image, 91 stuff categories, 80 object categories, 1.5m object instances, and context rich segmentation. The size of this dataset is almost 42.7 GB. To check the feasibility of proposed scheme, this image dataset is used.

5.4.3 Implementation Details

For the analyses of running time of the proposed PPSEI technique, the implementation was done in Python language on Ubuntu operating system. All algorithms of proposed scheme, as given in Section 3.6, are implemented at the client machine. For the testing purpose "2017 Val images [5K/1GB]" image dataset is used. This dataset contains 5,000 images. The pseudocodes of all algorithms are given in Appendix A.

5.5 Computation Overhead

The running time of each phase involved in PPSEI technique is presented in this section. The overhead analysis is represented as graphs generated in MS excel 365. The running time is divided in chunks of 500 images to observe the behavior of algorithms and time complexity required for each phase. We have also tested and presented the analysis of one image and one object query keyword for the analysis of a unit image. The computational overhead of each phase is given as follows:

KeyGen() Phase

Key generation phase is almost the same and independent of the number of images. As discussed in Section 5.2 and shown in Table 5.1, the KeyGen phase have complexity of $O(1)$. This phase is tested for one image only and 5,000 images with chunks of 500 images. The running time required for Master key is the average time of 60 millisecond and for the session key it is 11 millisecond. Figure 5.1 shows the running time of both master and session key generation functions. The session key generation function takes higher time at start but time decreases gradually when the library is properly initialized and loaded in memory. The master key generation function takes almost the same amount of time for key generation each time. It is evident that the key generation phase is almost constant and independent of number of input images. The master key takes more time than session key as shown in Figure 5.1.

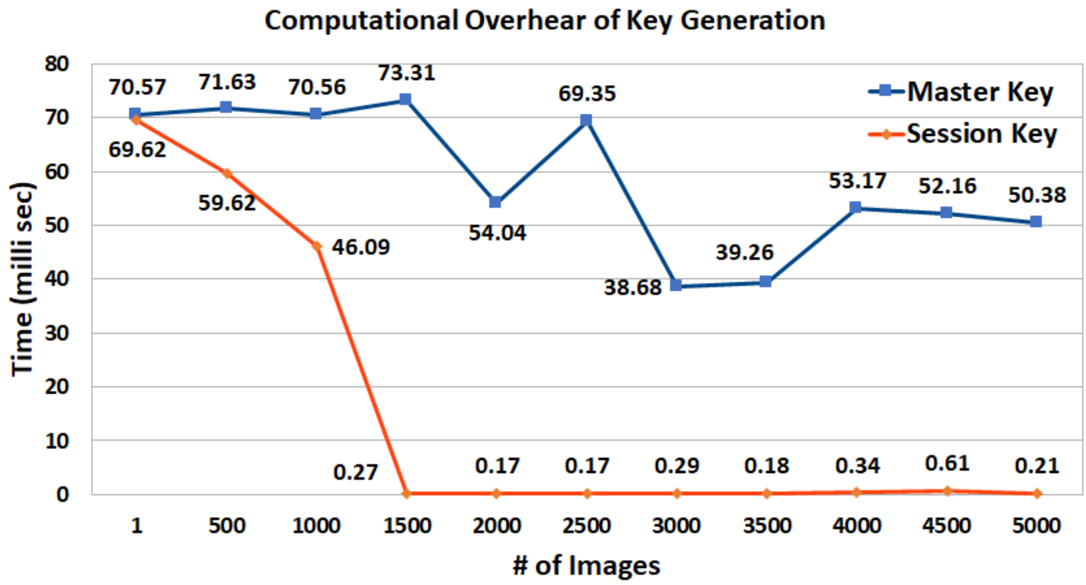


Figure 5.1: Computational Time for Key Generation

Obj_Det() Phase

The object detection phase involves 2 sub phases including object detection and relevance frequency calculation. As a result a table is formed called plain text index table. We can merge Obj_Det phase and Build_Obj_Index phases to reduce the number of operations at user end. For the analysis of computational overhead both phases are discussed separately. The running time of Obj_Det function is an average of 0.6 second for one image. For multiple images, the running time of Obj_Det phase is shown in Figure 5.2. We can see that with the number of images, time increases linearly and clearly verifies the algorithmic complexity of $O(n)$. As discussed earlier, this phase is option in the selection of object detection algorithm. The given running time is for the experimental values of YOLO v3 object detection algorithm.

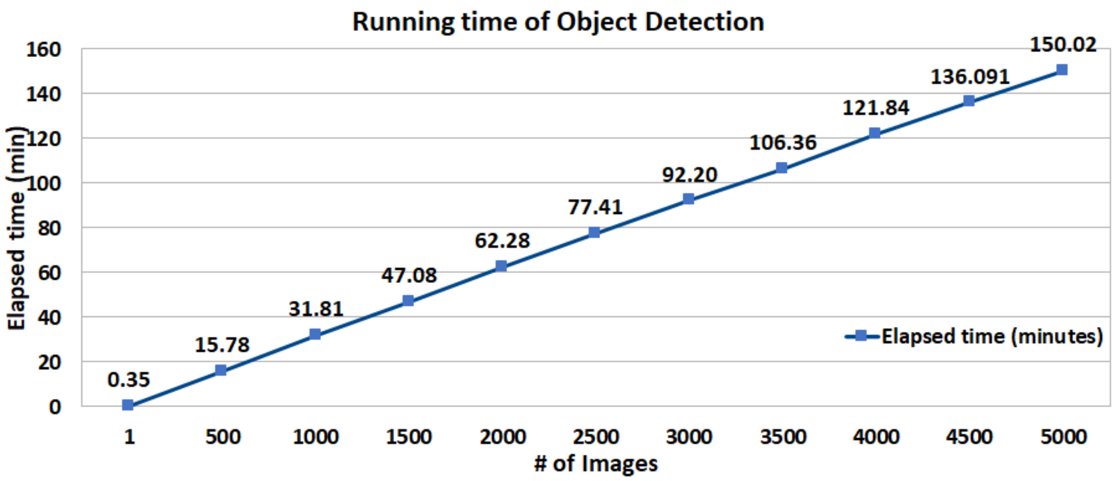


Figure 5.2: Computational Time for Object Detection

Build_Obj_Index() Phase

The user can run this function separately or can be merged with Obj_Det() function. For the computational overhead of this phase, we have calculated the time required for computations for different number of rows of plain text index table to encrypt and generate an encrypted index table IT. Running time for one one image data is an average of 0.36 seconds. The running time of Build_Obj_Index phase is given in Figure 5.3. We can see that with the number of images, rows in plain text index table increases and to generate the encrypted index table, time increases linearly. This verifies the algorithmic complexity is $O(n)$ for Build_Obj_Index phase.

Build_Query() Phase

This phase is almost independent of the number of input images. This phase takes the query object keyword as input and generates the probabilistic encrypted query trapdoor. This phase requires an average of 70 microseconds. For the case of YOLO v3 object detection algorithm trained on MS COCO dataset have 80 object classes. If the user

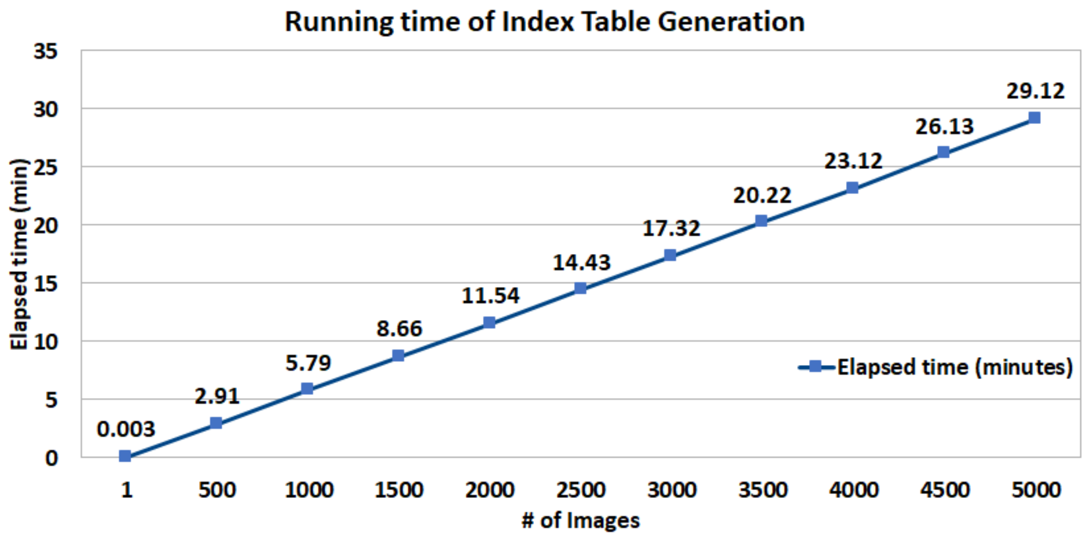


Figure 5.3: Computational Time for Object Index Table

want to search for an image containing all object keywords in an image, each query keyword is processed as an independent query to make it probabilistic. Therefore the time complexity remains the same for this phase *i.e.* $O(1)$ and requires the same amount of time for each object query keyword.

Search_Outcome() Phase

This function is performed at the cloud service provider. This phase takes the input arguments as query trapdoor Q_W and encrypted index table IT. The running time depends on two factors *i.e.* the number images to return and the number object keywords present in query trapdoor. For one image to return with one keyword search takes an average of 0.45 millisecond. The running time for this phase is given in Figure 5.4. We can see that the time required, to process the user query, increases linearly with number of images and that is accordance with the algorithmic analysis of this phase *i.e.* $O(n)$.

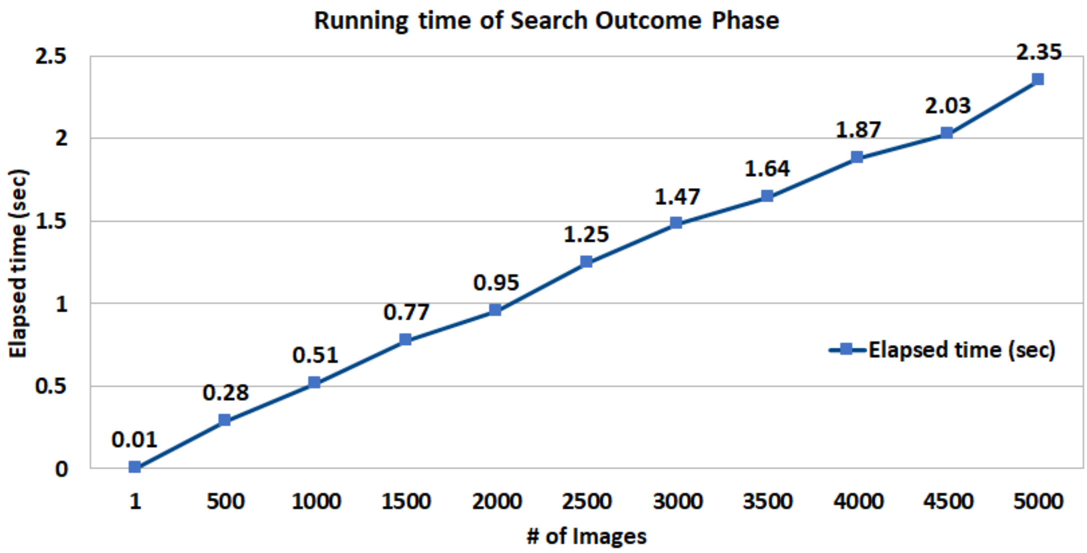


Figure 5.4: Computational Time for Search Outcome

Dec() Phase

Same like KeyGen phase, this phase is independent on the number of images present in database but depends on the number image identifiers returned to the user. As discussed earlier, the time complexity of this function is $O(1)$ and requires almost same amount of time to decrypt the image identifiers or images. The average running time is 11 microseconds to decrypt one image identifier.

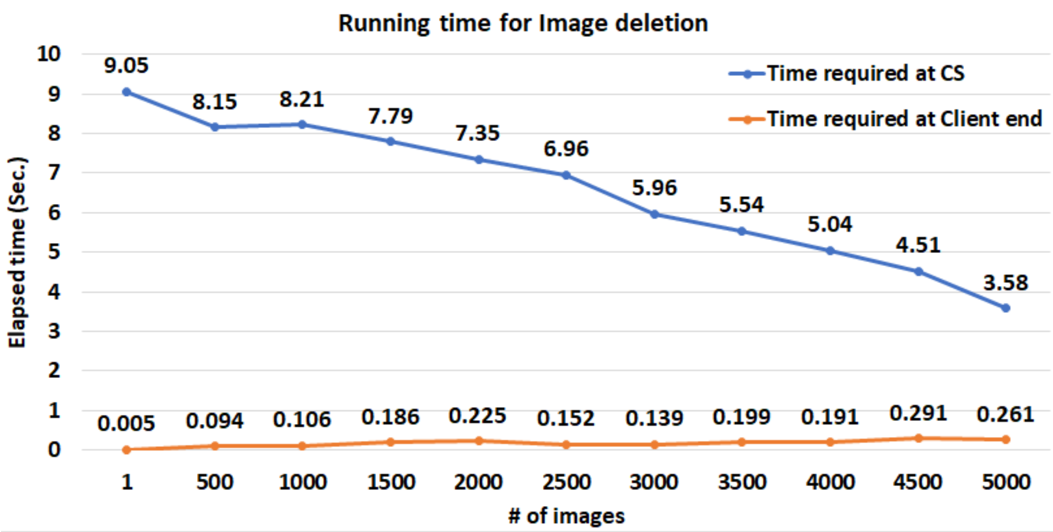


Figure 5.5: Computational Time for deletion of images

Dynamic Database

If the user want to add or delete some images from the database, dynamic database function is processed. To add new images to the database, the user will process first three phases including KeyGen, Obj_Det, and Build_Obj_Index. The Build_Obj_Index is processed without the object keyword encryption. This is because the encrypted keywords are already present in index table IT at CS. The running time for KeyGen and Obj_Det remains the same for image addition function in dynamic database while Build_Obj_Index phase takes less amount of time then the time required to generate an index table IT. At the CS, this IT_{add} index table is appended at the end of IT. The append function takes few microseconds to process.

While for the deletion of images, the user need to encrypt the image identifiers only. This requires the KeyGen phase and AES encryption function only. The running time increases with the number of images. For the deletion function at CS, only rows with encrypted image identifiers are searched and deleted. The image deletion query generation at user end is almost a constant with average time of 0.16 seconds while the image deletion query processing at the CS end have a linear function of time complexity as shown in Figure 5.5. Initially, more time is required to check the condition if requested image IDs are present in IT or not. With higher the number of requested image IDs in deletion request, less amount of time is required to process. This is because the image ID is found quickly rather traversing all the image IDs present in IT.

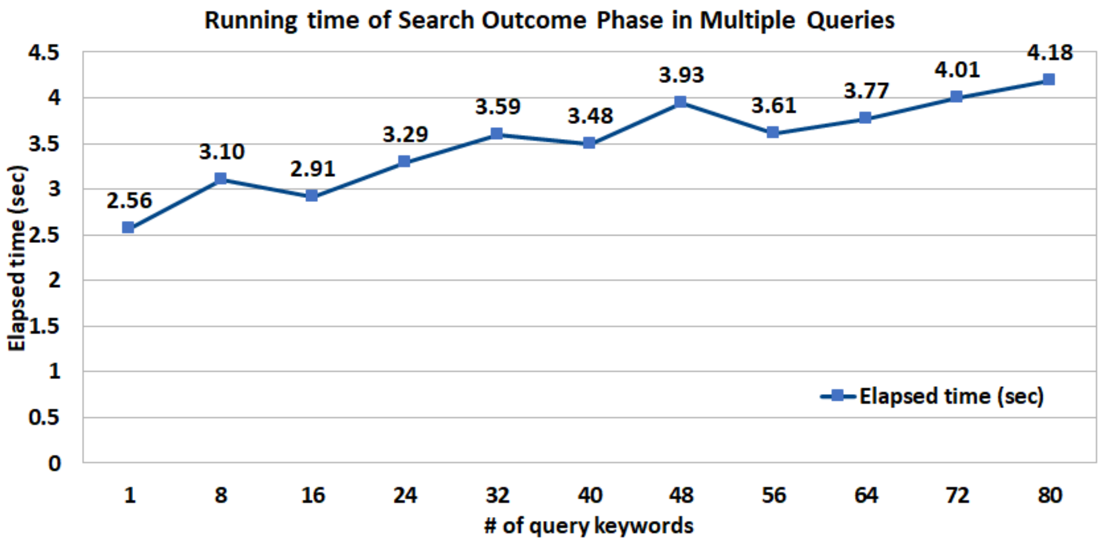


Figure 5.6: Computational Time for Search Outcome with batch query

Batch Queries

For the batch queries, user enters multiple query keywords or enter image containing multiple objects. As mentioned in Build_Query phase above, the maximum number of objects can be 80 for the case of object detection algorithm trained at MS COCO image dataset. At the user end, each object keyword is processed separately. Hence, the running time is constant for each object keyword and independent on the number of query object keywords or object keywords present in a query image. At the *CS* end, the Search_Outcome phase performs searching for *num* number of image identifiers against batch query keywords. The running time of batch query with different keywords is shown in Figure 5.6. The graph was plotter for the running time of batch queries at *CS* with fixed number of images *i.e.* 5,000. From the Figure 5.6, we can get see that the running time for this phase is slightly linear. Another point to note here is, higher the number of images, will require more processing time as to return the *num* number of images to the user.

5.6 Summary

This chapter discussed about the performance analysis in terms of computational and running time complexity of the proposed PPSEI technique. The algorithmic analysis tells the time complexity required to process each phase of the proposed scheme. The storage overhead gives the analysis of required space to store security keys, object detection algorithm files, random numbers file, index tables, and encrypted image files. The computational analysis shows the system specifications, dataset specifications and the implementation details. We have discussed the running time of each phase. Running time and computational overhead of dynamic databases and batch queries are also discussed in details. In Chapter 6, we have given the conclusion and future directions of the research. Some challenges faced during this research are also discussed in next chapter.

Chapter 6

Conclusion and Future Directions

Following swift enhancement in technologies concerning cloud computing, machine learning, and big data, clients are relying on outsourcing their data to the cloud storage. This data compromises mainly of multimedia and images. The main benefit that cloud storage offers to clients, be it any individual or an organization, is significant reduction of computational overhead for resource constraint devices. However, security threats remains major issue till date. Clients are concerned regarding privacy preservation and security of their personal data kept under the cloud administration. To overcome these issues, image processing over encrypted data can overcome this issue and provides different privacy-preserving techniques. The ability of image processing over encrypted data greatly reduces the privacy issues of individual users and enterprises as it gives the protection of valuable information from being leaked to non-trusted parties.

Currently, available techniques do not provide full privacy of image content and owner information or have high computational cost. Especially, while retrieving images from the CSP, user sends the query request to the CSP. These queries are not well protected and/or randomized. Therefore, they are prone to traceability issues and do not provide security from search pattern leakage attacks. In this thesis, we design a novel image processing technique that provides image content-based search and user privacy along with the un-tractability of user's search queries. Theoretical and experimental analysis shows that the proposed technique is robust in providing more security and performance than the available state-of-the-art techniques. In this chapter, we have presented an overview of our research, summary of contributions, and discuss some challenges & future directions.

6.1 Overview of Research

With the growth of technology and for new business, people are relying more on cloud service providers. To store a large number of images and processing over it becomes a challenging task while having resource constraint devices. The cloud services providers, provides storage as a service. The individual users and enterprises are motivated to outsource their personal and business related data on to the servers. While on the other hand, outsourcing restricts the users as the outsourced data is out of control from the users. The confidentiality of data is no longer exist in most of the cases. The user needs to perform normal operations over the data keeping the confidentiality of data intact. This leads us to the searchable encryption schemes. The current available techniques in the domain of image processing and searching are limited and are prone to search pattern leakage attacks.

In this thesis, we have presented a novel, content based privacy-preserving search over encrypted images PPSEI technique which resists the search pattern leakage attacks by generating probabilistic query keyword trapdoors. Also, the proposed scheme is independent of the object detection algorithm at user end. User can choose this algorithm depending upon his needs, working scenarios, and/or computational resources. The PPSEI can be implemented and used in real time scenarios.

6.2 Summary of Contributions

The research presented in this thesis discussed the security issues associated with currently available SE techniques in the domain of encrypted images. From the literature review presented in Chapter 2, it is clear that the current techniques are prone to search pattern leakage attacks that leads to the user traceability issue. As a results the user privacy is at risk. We have proposed a novel content based image retrieval scheme that preserves the user's privacy by generating the indeterministic/probabilistic keyword trapdoors as discussed in Chapter 3. The security analysis presents that the proposed PPSEI technique is secure in terms of leakage profiles defined in Chapter 4 and provides security in terms of keyword-trapdoor indistinguishability and trapdoor-index indistinguishability. The performance analysis, given in Chapter 5, clearly shows that the PPSEI technique is efficient and can be utilized in real word applications.

6.3 Challenges and Future Work

This section discusses the challenges faced during this research. These challenges will be addressed in future work.

6.3.1 Malicious Cloud Server

When the user outsources their data to *CS*, they lost partial/full control over the outsourced data. This prevents the adoption and utilization of cloud services for individuals and business organizations. The searchable encryption scheme resists this problem and provides more control over data to the end user. This thesis presented an SE scheme that provides the user privacy in terms of user un-traceability during query keyword searching process. The design of system architecture was on the assumptions of *CS* being honest-but-curious or trusted-but-curious. To address theses issues, probabilistic encryption of search queries are incorporated. To enhance the user privacy and security, the threat model can be modeled as cloud service providers being malicious *i.e.* the *CS* does not provide correct images in return to the query keyword, or does not delete images from the database when deletion request is sent from the user to the *CS*.

6.3.2 Multi-user setting

With the advancements in cloud computing, individual users and organizations are interested to outsource their personal and business related data to the *CSP*. The organizations and business are involved multiple vendors, entities, and individuals working from same geo-location or connected from around the globe. This needs the shared services and access of all required and desired entities to the outsourced data. This requires the multi-user environment and the SE scheme should handle all the entities based on their role or access controls. The proposed PPSEI technique is based on single writer / single reader (S/S) *i.e.* one owner and one user architecture. The owner can also be a user. The PPSEI scheme can be enhanced to S/M, M/S, and/or M/M architectures.

6.3.3 Dynamic object detection algorithms

With the advancement of technology and computer vision algorithms, the more accurate object detection algorithms are being presented and tested each day. The user requirements vary from time to time, scenario to scenario, and/or area of applications. This requires the SE scheme should be as flexible as to incorporate any object detection algorithm in base line of the proposed scheme. As discussed earlier, the proposed scheme is independent on the selection of object detection algorithm. The selection of one algorithm is necessary to perform other operations and the selection of algorithm can be considered as prerequisite for the proposed scheme. For testing purposes, we have used YOLO v3 algorithm. If the baseline object detection algorithm is changed *i.e.* from YOLO to SSD, Masked RCNN, or any good algorithm, the object detection process can be more precise, accurate and more number of objects, categories, classes, and instances can be found. Index table can incorporate more objects. For custom trained object detection algorithm settings, the algorithm can detect limited/more number of objects and/or can be optimized for user specified use case scenarios. The proposed scheme is flexible to incorporate any different object detection algorithm than YOLOv3 object detection algorithm.

References

[1] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation computer systems, 25(6):599–616, 2009.

[2] Kui Ren, Cong Wang, and Qian Wang. Security challenges for the public cloud. IEEE Internet Computing, 16(1):69–73, 2012.

[3] Zhengwei Ren, Lina Wang, Qian Wang, and Mingdi Xu. Dynamic proofs of retrievability for coded cloud storage systems. IEEE Transactions on Services Computing, 11(4):685–698, 2015.

[4] Zhangjie Fu, Kui Ren, Jiangang Shu, Xingming Sun, and Fengxiao Huang. Enabling personalized search over encrypted outsourced data with efficiency improvement. IEEE transactions on parallel and distributed systems, 27(9):2546–2559, 2015.

[5] Zhihua Xia, Xinhui Wang, Xingming Sun, and Qian Wang. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE transactions on parallel and distributed systems, 27(2):340–352, 2015.

[6] Richard Chow, Philippe Golle, Markus Jakobsson, Elaine Shi, Jessica Staddon, Ryusuke Masuoka, and Jesus Molina. Controlling data in the cloud: outsourcing computation without outsourcing control. In Proceedings of the 2009 ACM workshop on Cloud computing security, pages 85–90, 2009.

[7] Hassan Takabi, James BD Joshi, and Gail-Joon Ahn. Security and privacy challenges in cloud computing environments. IEEE Security & Privacy, 8(6):24–31, 2010.

[8] Lifei Wei, Haojin Zhu, Zhenfu Cao, Xiaolei Dong, Weiwei Jia, Yunlu Chen, and Athanasios V Vasilakos. Security and privacy for storage and computation in cloud computing. Information sciences, 258:371–386, 2014.

[9] Gansen Zhao, Chunming Rong, Jin Li, Feng Zhang, and Yong Tang. Trusted data sharing over untrusted cloud storage providers. In 2nd IEEE International Conference on Cloud Computing Technology and Science, pages 97–103. IEEE, 2010.

[10] Dimitrios Zissis and Dimitrios Lekkas. Addressing cloud computing security issues. Future Generation computer systems, 28(3):583–592, 2012.

[11] Deyan Chen and Hong Zhao. Data security and privacy protection issues in cloud computing. In 2012 International Conference on Computer Science and Electronics Engineering, volume 1, pages 647–651. IEEE, 2012.

[12] Eugenia Politou, Efthimios Alepis, and Constantinos Patsakis. Forgetting personal data and revoking consent under the gdpr: Challenges and proposed solutions. Journal of Cybersecurity, 4(1):tyy001, 2018.

[13] Dongyoung Koo, Junbeom Hur, and Hyunsoo Yoon. Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage. Computers & Electrical Engineering, 39(1):34–46, 2013.

[14] John E Dunn. Whatsapp's end-to-end encryption explained: What is it and does it matter? Techworld, April 2016. URL https://www.techworld.com/security/whatsapps-end-end-encryption-explained/.

[15] Tara Golshan. Why it's now impossible for whatsapp to help agencies like the fbi access messages. Vox, April 2016. URL https://www.vox.com/2016/4/6/11376642/whatsapp-encryption-terrorism-fbi.

[16] Kurt Wagner. Is your messaging app encrypted? Vox, December 2015. URL https://www.vox.com/2015/12/21/11621610/is-your-messaging-app-encrypted.

[17] Oleksandr Bodriagov and Sonja Buchegger. Encryption for peer-to-peer social networks. In Security and Privacy in Social Networks, pages 47–65. Springer, 2013.

[18] Katie Costello and Meghan Rimol. Gartner says worldwide iaas public cloud services market grew 37.3% in 2019, Aug 2020. URL https://www.gartner.com/en/newsroom/press-releases/2020-08-10-gartner-says-worldwide-iaas-public-cloud-services-market-grew-3

[19] Ashwin Chaudhary. Cloud security challenges in 2020, Feb 2020. URL https://cloudsecurityalliance.org/blog/2020/02/18/cloud-security-challenges-in-2020/.

[20] Ming Li, Shucheng Yu, Kui Ren, Wenjing Lou, and Y Thomas Hou. Toward privacy-assured and searchable cloud data storage services. IEEE Network, 27 (4):56–62, 2013.

[21] Dawn Xiaoding Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000, pages 44–55. IEEE, 2000.

[22] Christoph Bösch, Pieter Hartel, Willem Jonker, and Andreas Peter. A survey of provably secure searchable encryption. ACM Computing Surveys (CSUR), 47(2): 1–51, 2014.

[23] Fei Han, Jing Qin, and Jiankun Hu. Secure searches in the cloud: A survey. Future Generation Computer Systems, 62:66–75, 2016.

[24] Khadijah Chamili, Md Jan Nordin, Waidah Ismail, and Abduljalil Radman. Searchable encryption: A review. International Journal of Security and Its Applications, 11:79–88, 2017.

[25] Zuojie Deng, Kenli Li, Keqin Li, and Jingli Zhou. A multi-user searchable encryption scheme with keyword authorization in a cloud storage. Future Generation Computer Systems, 72:208–218, 2017.

[26] Ruixuan Li, Zhiyong Xu, Wanshang Kang, Kin Choong Yow, and Cheng-Zhong Xu. Efficient multi-keyword ranked query over encrypted data in cloud computing. Future Generation Computer Systems, 30:179–190, 2014.

[27] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano.

Public key encryption with keyword search. In International conference on the theory and applications of cryptographic techniques, pages 506–522. Springer, 2004.

[28] Ning Cao, Cong Wang, Ming Li, Kui Ren, and Wenjing Lou. Privacy-preserving multi-keyword ranked search over encrypted cloud data. IEEE Transactions on parallel and distributed systems, 25(1):222–233, 2013.

[29] Bo Zhang and Fangguo Zhang. An efficient public key encryption with conjunctive-subset keywords search. Journal of Network and Computer Applications, 34(1):262–267, 2011.

[30] Hui Yin, Zheng Qin, Lu Ou, and Keqin Li. A query privacy-enhanced and secure search scheme over encrypted data in cloud computing. Journal of Computer and System Sciences, 90:14–27, 2017.

[31] M Chuah and W Hu. Privacy-aware bedtree based solution for fuzzy multi-keyword search over encrypted data. In 2011 31st International Conference on Distributed Computing Systems Workshops, pages 273–281. IEEE, 2011.

[32] Zhangjie Fu, Xinle Wu, Chaowen Guan, Xingming Sun, and Kui Ren. Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement. IEEE Transactions on Information Forensics and Security, 11(12): 2706–2716, 2016.

[33] Eu-Jin Goh et al. Secure indexes. IACR Cryptology ePrint Archive, 2003:216, 2003.

[34] Adi Shamir. Identity-based cryptosystems and signature schemes. In Workshop on the theory and application of cryptographic techniques, pages 47–53. Springer, 1984.

[35] Xin Dong, Jiadi Yu, Yanmin Zhu, Yingying Chen, Yuan Luo, and Minglu Li. Seco: Secure and scalable data collaboration services in cloud computing. computers & security, 50:91–105, 2015.

[36] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In annual international conference on the theory and applications of cryptographic techniques, pages 146–162. Springer, 2008.

[37] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM conference on Computer and communications security, pages 89–98, 2006.

[38] Xu An Wang, Fatos Xhafa, Weiyi Cai, Jianfeng Ma, and Fushan Wei. Efficient privacy preserving predicate encryption with fine-grained searchable capability for cloud storage. Computers & Electrical Engineering, 56:871–883, 2016.

[39] Jong Hwan Park. Efficient hidden vector encryption for conjunctive queries on encrypted data. IEEE Transactions on Knowledge and Data Engineering, 23(10): 1483–1497, 2010.

[40] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hi-

erarchical) inner product encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 62–91. Springer, 2010.

[41] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 591–608. Springer, 2012.

[42] Ruixuan Li, Zhiyong Xu, Wanshang Kang, Kin Choong Yow, and Cheng-Zhong Xu. Efficient multi-keyword ranked query over encrypted data in cloud computing. Future Generation Computer Systems, 30:179–190, 2014.

[43] Ben-Zion Chor, Oded Goldreich, and Eyal Kushilevitz. Private information retrieval, December 29 1998. US Patent 5,855,018.

[44] Xun Yi, Mohammed Golam Kaosar, Russell Paulet, and Elisa Bertino. Single-database private information retrieval from fully homomorphic encryption. IEEE Transactions on Knowledge and Data Engineering, 25(5):1125–1134, 2012.

[45] Thomas Pöppelmann and Tim Güneysu. Towards efficient arithmetic for lattice-based cryptography on reconfigurable hardware. In International Conference on Cryptology and Information Security in Latin America, pages 139–158. Springer, 2012.

[46] Wei Wang, Yin Hu, Lianmu Chen, Xinming Huang, and Berk Sunar. Accelerating fully homomorphic encryption using gpu. In 2012 IEEE conference on high performance extreme computing, pages 1–5. IEEE, 2012.

[47] Zhigang Chen, Jian Wang, ZengNian Zhang, and Xinxia Song. A fully homomorphic encryption scheme with better key size. China Communications, 11(9): 82–92, 2014.

[48] Liquan Chen, Hongmei Ben, and Jie Huang. An encryption depth optimization scheme for fully homomorphic encryption. In 2014 International Conference on Identification, Information and Knowledge in the Internet of Things, pages 137–141. IEEE, 2014.

[49] Yatao Yang, Shuang Zhang, Junming Yang, Jia Li, and Zichen Li. Targeted fully homomorphic encryption based on a double decryption algorithm for polynomials. Tsinghua science and technology, 19(5):478–485, 2014.

[50] Yan-Cheng Chang and Michael Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. In International Conference on Applied Cryptography and Network Security, pages 442–455. Springer, 2005.

[51] Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. Searchable symmetric encryption: improved definitions and efficient constructions. Journal of Computer Security, 19(5):895–934, 2011.

[52] Shahzaib Tahir, Sushmita Ruj, Yogachandran Rahulamathavan, Muttukrishnan Rajarajan, and Cornelius Glackin. A new secure and lightweight searchable encryption scheme over encrypted cloud data. IEEE Transactions on Emerging Topics in Computing, 2017.

[53] Tengfei Yang, Jianfeng Ma, Qian Wang, Yinbin Miao, Xuan Wang, and Qian

Meng. Image feature extraction in encrypted domain with privacy-preserving hahn moments. IEEE Access, 6:47521–47534, 2018.

[54] Yuan Wang, Meixia Miao, Jian Shen, and Jianfeng Wang. Towards efficient privacy-preserving encrypted image search in cloud computing. Soft Computing, 23(6):2101–2112, 2019.

[55] Zhihua Xia, Xinhui Wang, Liangao Zhang, Zhan Qin, Xingming Sun, and Kui Ren. A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. IEEE transactions on information forensics and security, 11(11):2594–2608, 2016.

[56] Zhihua Xia, Neal N Xiong, Athanasios V Vasilakos, and Xingming Sun. Epcbir: An efficient and privacy-preserving content-based image retrieval scheme in cloud computing. Information Sciences, 387:195–204, 2017.

[57] Shengshan Hu, Qian Wang, Jingjun Wang, Zhan Qin, and Kui Ren. Securing sift: Privacy-preserving outsourcing computation of feature extractions over encrypted image data. IEEE Transactions on Image Processing, 25(7):3411–3425, 2016.

[58] Sayyada Fahmeeda Sultana and DC Shubhangi. Privacy preserving lbp based feature extraction on encrypted images. In 2017 International Conference on Computer Communication and Informatics (ICCCI), pages 1–4. IEEE, 2017.

[59] Hongqing Zhu, Huazhong Shu, Jian Zhou, Limin Luo, and Jean-Louis Coatrieux. Image analysis by discrete orthogonal dual hahn moments. Pattern Recognition Letters, 28(13):1688–1704, 2007.

[60] Pew-Thian Yap, Raveendran Paramesran, and Seng-Huat Ong. Image analysis using hahn moments. IEEE transactions on pattern analysis and machine intelligence, 29(11):2057–2062, 2007.

[61] Fatima Akhmedova and Simon Liao. Face recognition using discrete orthogonal Hahn moments. PhD thesis, Department of Applied Computer Science, University of Winnipeg, 2015.

[62] Zhuohua Liu, Caijuan Huang, Hui Suo, and Bin Yang. A novel content based image retrieval scheme in cloud computing. In International Conference on Artificial Intelligence and Security, pages 606–616. Springer, 2019.

[63] Wenjun Lu, Ashwin Swaminathan, Avinash L Varna, and Min Wu. Enabling search over encrypted multimedia databases. In Media Forensics and Security, volume 7254, page 725418. International Society for Optics and Photonics, 2009.

[64] J Anju and R Shreelekshmi. Secure content-based image retrieval using combined features in cloud. In International Conference on Distributed Computing and Internet Technology, pages 179–197. Springer, 2020.

[65] Jiawei Yuan, Shucheng Yu, and Linke Guo. Seisa: Secure and efficient encrypted image search with access control. In 2015 IEEE conference on computer communications (INFOCOM), pages 2083–2091. IEEE, 2015.

[66] Lan Zhang, Taeho Jung, Kebin Liu, Xiang-Yang Li, Xuan Ding, Jiaxi Gu, and Yunhao Liu. Pic: Enable large-scale privacy preserving content-based image search on cloud. IEEE Transactions on Parallel and Distributed Systems, 28(11): 3258–3271, 2017.

[67] Xiangyu Wang, Jianfeng Ma, Ximeng Liu, and Yinbin Miao. Search in my way: Practical outsourced image retrieval framework supporting unshared key. In IEEE INFOCOM 2019-IEEE Conference on Computer Communications, pages 2485–2493. IEEE, 2019.

[68] Zhengbai Huang, Meng Zhang, and Yi Zhang. Toward efficient encrypted image retrieval in cloud environment. IEEE Access, 7:174541–174550, 2019.

[69] Hua Wang, Zhihua Xia, Jianwei Fei, and Fengjun Xiao. An aes-based secure image retrieval scheme using random mapping and bow in cloud computing. IEEE Access, 8:61138–61147, 2020.

[70] Chengyuan Zhang, Lei Zhu, Shichao Zhang, and Weiren Yu. Tdhppir: An efficient deep hashing based privacy-preserving image retrieval method. Neurocomputing, 2020.

[71] Jianhua Chen, Jiaohua Qin, Xuyu Xiang, and Yun Tan. A new encrypted image retrieval method based on feature fusion in cloud computing. International Journal of Computational Science and Engineering, 22(1):114–123, 2020.

[72] Zhihua Xia, Lihua Lu, Tong Qiu, HJ Shim, Xianyi Chen, and Byeungwoo Jeon. A privacy-preserving image retrieval based on ac-coefficients and color histograms in cloud environment. Computers, Materials & Continua, 58(1):27–44, 2019.

[73] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[74] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva

Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer, 2014.

[75] Surya Remanan. Beginner's guide to object detection algorithms. Analytics Vidhya, February 2020. URL https://medium.com/@ksuryaremanan/beginners-guide-to-object-detection-algorithms-6620fb31c375.

[76] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. arXiv preprint arXiv:1905.05055, 2019.

[77] Lilian Weng. Object detection part 4: Fast detection models. github, December 2018. URL https://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html.

[78] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587, 2014.

[79] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9):1904–1916, 2015.

[80] Ross Girshick. Fast r-cnn. In Proceedings of the IEEE international conference on computer vision, pages 1440–1448, 2015.

[81] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards

real-time object detection with region proposal networks. In Advances in neural information processing systems, pages 91–99, 2015.

[82] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 779–788, 2016.

[83] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7263–7271, 2017.

[84] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In European conference on computer vision, pages 21–37. Springer, 2016.

[85] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2117–2125, 2017.

[86] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision, pages 2980–2988, 2017.

[87] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In Advances in neural information processing systems, pages 379–387, 2016.

[88] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In Proceedings of the IEEE international conference on computer vision, pages 2961–2969, 2017.

[89] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[90] Ankit Sachan. Zero to hero: Guide to object detection using deep learning: Faster r-cnn,yolo,ssd. cv-tricks, December 2017. URL https://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/.

[91] Ronald L Rivest and Alan T Sherman. Randomized encryption techniques. In Advances in Cryptology, pages 145–163. Springer, 1983.

[92] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O'neill. Order-preserving symmetric encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 224–241. Springer, 2009.

[93] Jianfeng Wang, Jingdong Wang, Nenghai Yu, and Shipeng Li. Order preserving hashing for approximate nearest neighbor search. In Proceedings of the 21st ACM international conference on Multimedia, pages 133–142, 2013.

[94] Guoming Chen, Qiang Chen, Xiongyong Zhu, and Yiqun Chen. Encrypted image feature extraction by privacy-preserving mfs. In 2018 7th International Conference on Digital Home (ICDH), pages 42–45. IEEE, 2018.

[95] Haihua Liang, Xinpeng Zhang, Hang Cheng, and Qiuhan Wei. Secure and ef-

ficient image retrieval over encrypted cloud data. Security and Communication Networks, 2018, 2018.

[96] Shaheen Ayyub and Praveen Kaushik. Secure searchable image encryption in cloud using hyper chaos. Int. Arab J. Inf. Technol., 16(2):251–259, 2019.

[97] Wentao Ma, Jiaohua Qin, Xuyu Xiang, Yun Tan, and Zhibin He. Searchable encrypted image retrieval based on multi-feature adaptive late-fusion. Mathematics, 8(6):1019, 2020.

Appendix A

Appendixes

The pseudo codes of proposed scheme are given here.

1: KeyGen() This function generates master and session keys. Master key is based on user password while session key is generated randomly each time. Pseudo code of KeyGen function is given in Algorithm 1.

2: Obj_Det() This function identifies the objects available in images and return the object class name along with the frequency of that object appearing in image. Any object detection algorithm can be used with the proposed PPSEI scheme.

3: Build_Obj_Index() This function generates an encrypted index table. The inputs to this function is the plain text index table, master key, and an initialization vector. Pseudo code of Build_Obj_Index() phase is given in Algorithm 2.

4: Build_Query() This function generates the probabilistic encrypted queries based on user query keyword by taking object keyword, session key, and an initialization vector as input arguments. Pseudo code of Build_Query() phase is given in Algorithm 3.

5: Search_Outcome() This function finds the image identifiers based on user object query. The input arguments of this function are query trapdoor Q_W and index table IT. Based on user query trapdoor, image identifiers are sent to the user. Pseudo code of Search_Outcome() phase is given in Algorithm 4.

6: Dec() This function decrypts the image identifiers received from the cloud server. Pseudo code of Dec() phase is given in Algorithm 5.

7: Image Encryption: When the user successfully generates an IT, he will also encrypts the images. This IT and encrypted images are then sent to the CS. Pseudo code of Image encryption function is given in Algorithm 6. After performing a successful query, user will get encrypted images from the CS. To decrypt the encrypted images, an image decryption function is used. Pseudo code of image decryption function is given in Algorithm 7.

8: Dynamic Databases: For dynamic database, if the user want to outsource more images and to delete images from the database, addition and deletion of image function will be used at the user side. Pseudo codes of image addition and image deletion functions at user end are given in Algorithms 8 and 9 respectively. At the CS end, 10 and 11 functions, image addition and image deletion functions are used.

Algorithm 1: KeyGen() Phase

Input: A security parameter λ;
Output: Master Key K, Session Key k_s;
KeyGen: Generate keys $K, k_s \leftarrow \{0,1\}^{\lambda}$

Algorithm 2: Build_Obj_Index() Phase

```
Input: Master key, iv;
Output: Encrypted index table IT;
import AES encryption function;
import HMAC function;
import log from math function;
input plain text index table as PIT;
for number of rows in PIT do
    if row = = 1 then
        # Calculate hashes of object classes ;
        for number of columns in row do
            parameter_a = HMAC(Master key, object class);
            save in first row of IT;
        end
    else
        # Calculate encrypted image identifiers & RF masking ;
        for number of columns in row do
            if column number = = 1 then
                Encrypted Img ID = AES(image id, Master key, iv);
                save Encrypted Img ID in IT;
            else
                input random numbers;
                set mask_RF = log(RF+2) × random_1 + random_2;
                save mask_RF in IT;
            end
        end
    end
end
return Encrypted index table IT;
```

Algorithm 3: Build_Query() Phase

```
Input: Master key, Session key, iv, keyword, num;
Output: Q_W;
import AES encryption function;
import HMAC function;
import XOR from XOR function;
parameter_a = HMAC(Master key, object keyword);
parameter_b = AES(Object keyword, Session key, iv);
parameter_c = XOR (parameter_a, parameter_b);
paramete_d = Hash (parameter_b);
set Q_W = parameter_c, paramete_d, num
return Q_W;
```

Algorithm 4: Search_Outcome() Phase

```
Input: IT, Q_W;
Output: A[ ];
import Hash function;
import XOR function;
import IT from database;
for number of columns in IT do
    parameter_a′ = object keyword in IT;
    xor data = XOR(parameter_a′, parameter_c);
    paramete_d′ = Hash (xor data);
    if paramete_d′ = = paramete_d then
        for number of rows in IT do
            # Find the num number of img ids based on RF ranking
            set A[ ] = Encrypted Img IDs ;
        end
    end
end
return Image identifiers array A;
```

Algorithm 5: Dec() Phase

```
Input: ciphered Img IDs, Master key, iv;
Output: Plaint text Image IDs;
def AES_Dec_fun(ciphered Img IDs, Master key, iv)
    import AES from Pycryptodome functions;
    for (number of Img IDs) do
        plain text ID = AES decryption (ciphered Img IDs, Master key, iv);
    end
    return Plain text image IDs;
```

Algorithm 6: Image encryption function

```
Input: Plain text Images;
Output: Encrypted Images;
import os, struct;
import AES from Pycryptodome functions;
def file encryption(Master key, file name)
    import random bytes from Random function ;
    set iv = random bytes of size 16;
    encrypted img ID = AES encryption (file name, Master key, iv);
    open file with write function ;
    check size of file;
    encrypt image content ;
    set encrypted img ID to this content ;
    Encrypted Image = (encrypted image content, encrypted img ID) ;
    return Encrypted Image;
```

Algorithm 7: Image decryption function

```
Input: Encrypted Images;
Output: Plain text Images;
def file decryption(Encrypted file name, Master key)
    read iv from image content;
    open file with write access rights ;
    check size of file;
    decrypted img ID = AES decryption (Encrypted Img ID, Master key, iv);
    decrypt image content ;
    set decrypted img ID to this content ;
    return Decrypted Image;
```

Algorithm 8: Image addition Phase at client side

```
Input: New images to add;
Output: Image addition IT_add;
import KeyGen(), Obj_Dec() function ;
import Build_Obj_Index function ;
for number of images to add do
    read image from folder;
    object detected = Obj_Dec(image);
    IT_add = Build_Obj_Index (object detected) ;
end
return Image addition IT_add;
```

Algorithm 9: Image deletion Phase at client side

```
Input: Images ID to delete;
Output: Encrypted image IDs to delete;
user input img IDs to delete;
import AES from Pycryptodome functions;
for number of Img IDs do
    Encrypted img ids = AES encryption (Img IDs, Master key, iv);
    set img deletion [] = Encrypted img ids;
end
return img deletion [];
```

Algorithm 10: Image addition Phase at cloud server

```
Input: Image addition IT_add;
Output: updated IT;
read IT from database ;
append IT_add at the end of IT;
return updated IT;
```

Algorithm 11: Image deletion Phase at cloud server

```
Input: Encrypted image IDs to delete;
Output: updated IT;
read IT from database ;
for number of encrypted Img IDs do
    for N = 1,2,..., number of rows in IT do
        if Encrypted Img ID = = IT[N] then
            delete the current row;
        else
            Print: Encrypted Img ID not present;
        end
    end
end
return updated IT;
```