A Proposed Framework for Search as a Service on Private Cloud

Bukhary Ikhwan Ismail, Rajendar Kandan, Hishamadie Ahmad, Ehsan Mostajeran Goortani, Mohd Nizam Mohd Mydin, Mohammad Fairus Khalid, Ong Hong Hoe
Advanced Computing Lab, MIMOS Berhad, Kuala Lumpur, Malaysia
{ikhwan.ismail, rajendar.kandan, hishamadie.ahmad, ehsan.mostajeran, nizam.mydin, fairus.khalid, hh.ong}@mimos.my

***Abstract*—Collecting data from multiple sources and putting into a unified silo is now a common practice. It is important to retrieve these voluminous data in a quick and efficient manner. Using search engine indexing capabilities gives the user a low-latency query towards the data. But often to set up and maintain such an infrastructure is complex and hard to maintain. In this paper, we discuss the challenges and requirement to host a search engine as a service. We take the requirement based on labs and department requirements. We propose a framework to host search engine on top of private cloud deployment. It uses a mixture of containerization and virtualization for faster deployment and easy maintenance.**

***Index Term*—Information Retrieval, Search Engine, Indexing, Big Data, full-text search, Elasticsearch, Lucene, machine learning**

I. Introduction

Search engine (SE) application is suitable for handling a large volume of text data. It optimizes speed and performance in querying or retrieval of text information. Web pages, email, scholarly papers, book, and news article are some of the documents examples. SE is used in different areas e.g. desktop search, enterprise search e.g. Google Search Appliance to latest breed big data search, centralized log management, financial analysis and more.

In current trends, SE expands from mere text indexer into a broader scope. With the norm of web services, micro-services, and distributed computing other forms of text data become an interest. For example logs, audit trails, clickstream (records user's surfing history), social media, financial, marketing, shopping cart, performance metrics and more. Techniques such as machine learning, prediction and analytics are applied to find interesting data point patterns and insights. Data from multiple sources is pushed into data silos (Data Lake) paired with indexing capabilities of SE to provide cohesive analytics and fast retrieval of information.

Search infrastructure provides the foundation to collect, enriched, store and retrieve information faster. With the low-latency search capabilities, many ways of data manipulation can be done. Before a user can use the Search Infrastructure (SI), it requires the help of system admin and best practices to have a solid deployment. Without it, it would be problematic.

In this paper, we propose a Search Engine for A Service (SEaaS). Our SEaaS objective is to automatic deployment SE in our internal private cloud. Currently, in MIMOS, many department and research lab is utilizing SE for multiple purposes. To name a few, social intelligence, geoscience & petroleum engineering analytics, and others. Our goal is to provide search service as a service for research lab by giving them fast deployment, easy to maintain, automated process e.g. index rotation, index backup without any hassle.

The contribution of this paper is a proposed architecture and framework to make on-demand Search as a Service (SEaaS). The work is organized as follows, Section II discusses the background of search engine, section III, current search infrastructure survey, section IV the reference architecture section V recommendation and discussion and lastly section VI conclusion.

II. Background

A. Search Engine

There is two distinct phase of Search Engine (SE). 1) Document ingestion – a process that collects, parse, index and store into storage. 2) Document retrieval – the user performs a query and relevant information returned to the user.

Document ingestion – the process involves multiple steps where indexing is the core process. In indexing, the data or document is scanned to build a list of search term often called an index. The indexer will make an entry for each term found and possibly note its relative position within the document [1]. The index is not an assemblage of original documents e.g. pdf, HTML or logs; rather a list of terms within the documents which can be retrieved. By querying the index data structure instead of the original text, it significantly improves the query request. The most popular indexing strategy is an inverted index. It is inverted in the sense it is the opposite of document file that lists [2]

Document Retrieval - SE searches inside the index and returns hits on the query. It first analyzes the user query using