

Haystack-based Facebook's data storage architecture: store, directory, and cache

Tole Sutikno^{1,2}, Ahmad Heryanto^{3,4}, Laksana Talenta Ahmad^{2,5}

¹Faculty of Industrial Technology, Universitas Ahmad Dahlan, Yogyakarta, Indonesia

²Embedded Systems and Power Electronics Research Group, Yogyakarta, Indonesia

³Faculty of Engineering, Universitas Sriwijaya, Palembang, Indonesia

⁴Faculty of Computer Science, Universitas Sriwijaya, Palembang, Indonesia

⁵Department of Information Systems, International Islamic University Malaysia, Kuala Lumpur, Malaysia

Article Info

Article history:

Received Oct 21, 2024

Revised May 8, 2025

Accepted Jun 4, 2025

Keywords:

Architecture

Cache

Data storage

Directory

Facebook

Haystack

ABSTRACT

Haystack is Facebook's unique way of managing large amounts of user-generated content like photos. The architecture prioritizes performance, reliability, and scalability to overcome network-attached storage system bottlenecks. Haystack speeds data access and ensures data integrity during hardware failures by using physical and logical volumes. This study examines the architecture of Facebook's Haystack data storage system and its effects on scalability and efficiency in handling large photo data. According to the study, the store, directory, and cache functions work together to reduce input/output (I/O) operations and improve metadata processing, which traditional network-attached storage systems cannot do. Haystack manages massive photo data storage and retrieval, solving network-attached storage (NAS) limitations. It balances throughput and latency by minimizing disk operations and optimizing metadata processing. Each store, directory, and cache contribute to this ecosystem. The Haystack architecture reduces disk operations and metadata processing bottlenecks with distributed caching. A cache allows instant access to frequently requested images and balances read and write operations across the system. We should study advanced storage system architectures based on Facebook's Haystack architecture. This could involve investigating faster metadata processing algorithms, using artificial intelligence (AI) to improve fault detection and repair systems, and assessing the economic impact of distributed caches.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Tole Sutikno

Faculty of Industrial Technology, Universitas Ahmad Dahlan

Ahmad Yani Street (South of Ring Road), Tamanan, Yogyakarta 55191, Indonesia

Email: tole@te.uad.ac.id

1. INTRODUCTION

In the fast-changing world of data storage technology, having systems that can handle large amounts of information is very important. Facebook's unique way of tackling this problem is shown through its Haystack design, meant to manage the huge scale of user-created content like photos. This design skillfully avoids common problems found in network-attached storage systems by putting together a strong system that focuses on speed, dependability, and growth. By using both physical and logical volumes, Haystack boosts data access speed and adds important redundancy measures to keep data safe during hardware issues. The architecture has three main parts: the store, directory, and cache. Each part is crucial for optimizing photo retrieval, allowing for quick read and write actions. The store handles the effective management of raw photo

information, while the directory makes fast lookups and organizes the stored pictures. At the same time, the cache acts as a speedy link, greatly cutting down response time and enhancing user experience. The combined work of these parts changes the way data storage works, ensuring Facebook can manage billions of photos reliably and easily. Looking into the details of Haystack's architecture shows not just its technical skill but also its important impact on the future of distributed systems. As the digital world keeps growing, knowing how Haystack is designed provides useful insights for creating scalable solutions in many areas [1]–[8]. This study will break down the essential features of the architecture, looking at how each part helps build a strong and effective system for handling large data amounts, ultimately serving as a model for future advancements in data storage technology.

Handling a large amount of content made by users requires a strong and smart way to store data for Facebook. With more than 260 billion photos uploaded by users, having storage that can grow and work well is very important. Old network-attached storage (NAS) systems were not enough because they had issues with disk operations and processing metadata, leading Facebook to look for new methods. The Haystack system came as a solution, organizing storage into many physical volumes, each able to hold millions of photos [9]. This setup is built to copy data across different storage areas to avoid loss and ensure reliability. Also, ongoing background tasks keep checking and fixing possible problems, protecting user data from hardware issues [10]. Thus, Facebook's storage needs require a very effective, scalable, and fault-resistant system to handle the huge amounts of information created by its worldwide users.

A good data storage setup is crucial for boosting performance and ensuring reliability when managing large amounts of data. For example, Facebook's Haystack system was made to handle more than 260 billion photos. A well-organized setup reduces the need for disk operations and metadata processing, which improves speed and cuts down on delay. In distributed systems, this kind of setup promotes scalability, helping organizations grow their storage without many performance issues. Haystack shows how using a mix of storage, directory, and cache elements can enhance fault tolerance and save costs [9]. Such advancements show that smart storage design not only reduces risks related to hardware issues but also meets the growing demand for quick data access in a world where information is extensive and important [11].

The Haystack system is an important step forward in how large data is managed and saved, especially made to meet Facebook's huge photo storage needs. At its core, the system has three main parts: the Haystack store, directory, and cache. This three-part design helps make data retrieval and storage smoother, greatly improving operations by lowering disk input/output (I/O) and boosting fault tolerance with redundancy and ongoing background tasks. By sorting data into logical and physical volumes that provide photo backup, Haystack solves important problems regarding hardware breakdown and data loss, which leads to better reliability and performance [9]. Also, the system uses a distributed cache to cut down on wait times while keeping high output, thus creating a strong setup that handles current needs and is ready for future expansion [12].

The main goals of this research paper are to analyze the design features of Facebook's Haystack data storage system and to explain how they affect scalability and efficiency when dealing with large amounts of photo data. By looking closely at the store, directory, and cache functions, this paper aims to show how these components work together to reduce I/O operations and improve metadata processing, thus tackling issues seen with older network-attached storage systems. Moreover, the research will discuss the effects of using standard hardware and redundancy methods on system reliability and fault tolerance [9]. Through case studies and measurable performance metrics, this paper intends to add to the current knowledge of distributed systems design and provide useful guidance for developers and architects who want to improve large-scale data management strategies [13]. In the end, the research hopes to deepen the understanding of scalable architectures in modern software settings. It is very important to explain Haystack's data storage structure at Facebook. The introduction gives an overview of why scalable storage systems matter in today's world, especially for content-heavy sites like Facebook. Next, it looks closely at the main parts store, directory, and cache—explaining how each part works and interacts to ensure quick photo retrieval and reliability. The conclusion brings together these parts, focusing on what Haystack means for future data storage methods and improvements in distributed system design. By clearly moving from basic ideas to more complex uses, the study not only shows Facebook's innovative method but also provides a model for creating strong storage systems, as pointed out in [9] and [11].

2. UNDERSTANDING HAYSTACK ARCHITECTURE

Facebook's Haystack system has a design that manages a lot of photo data storage and retrieval. It solves issues seen in standard NAS systems [8], [14], [15] as shown in Figure 1. By reducing disk operations and improving metadata tasks, Haystack finds a good mix of high speed and low wait time, which is important for services with over 260 billion images [9]. Each part, store, directory, and cache has a

specific function in this system. The Haystack store arranges physical volumes into logical structures, which helps with data backup and improves reliability by using regular hardware, as shown in Figure 2. This method helps reduce problems from hardware failures and is supported by background tasks that find and fix problems. Additionally, recent studies show that knowing how memory access costs relate to computing efficiency is important; using tools like Haystack helps developers learn how to improve their code with cache systems, leading to a better storage setup overall [16]. Table 1 shows the NAS vs Haystack comparison table.

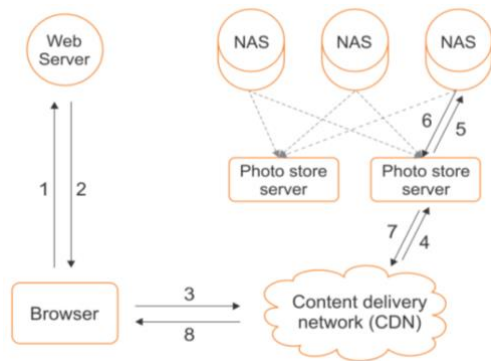


Figure 1. NAS over network file system (NFS)-based storage architecture [9]

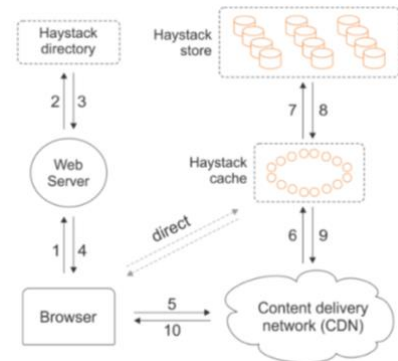


Figure 2. Haystack storage architecture [9]

Table 1. Comparison table of NAS vs. Haystack

Aspect	NAS	Haystack
Purpose of design	General purpose for various types of data and applications.	Specifically designed to store and manage user photos efficiently.
Architecture	File-based; uses protocols like NFS or server message block (SMB) for data access.	Object-based; uses a unique approach to store metadata and data together.
Performance	Higher latency for frequently accessed data.	Low latency for fast access to data (photos) with query optimization.
Storage efficiency	Metadata and data are stored separately, increasing metadata overhead.	Metadata and data are stored in a single unit, reducing metadata overhead.
Scalability	More challenging to scale at a large scale.	Designed for large-scale support, it efficiently handles billions of files.
Redundancy	Depends on redundant array of independent disks (RAID) systems for fault tolerance.	Uses distributed replication mechanisms for data reliability and availability.
Optimization for specific data	Not optimized for specific data types.	Optimized for photos and other large media, including management of large object sizes.
Metadata management	Metadata is managed separately, requiring additional queries for actual data.	Metadata is stored with data in a single index, accelerating data retrieval.
Operational cost	Relatively high due to reliance on specialized hardware.	Lower due to the use of commodity hardware and custom design.
Reliability	Relies on traditional NAS hardware and software.	More reliable with integrated data replication and fault tolerance.

2.1. Definition and components of Haystack

In Facebook's data storage setup, Haystack is a complex system that helps manage large photo collections efficiently. It has three main parts: the store, directory, and cache, each important for improving how data is stored and retrieved. The store holds most photos, organizing them in various physical volumes to provide backup and dependability against hardware issues. The directory takes care of the metadata, allowing fast access to necessary information while reducing disk use and metadata processing. This greatly improves performance. Lastly, the cache acts as a fast link, lowering delay by quickly providing frequently accessed data. This three-part system not only improves efficiency but also aids Facebook in managing a huge amount of content with strength and flexibility, ensuring a good user experience even as the demand for storage increases [10].

2.2. Historical context and development of Haystack

The rise of Haystack as a key data storage design is closely tied to Facebook's rapid increase in user-generated content, especially photos. At first, Facebook used NAS systems [8], [14], [15], which became

insufficient when the platform grew to house over 260 billion photos. This inadequacy highlighted the necessity for a new storage system to meet the growing needs for speed, reliability, and data integrity. The development of Haystack drew from detailed case studies in scalable system design, uncovering important ways to minimize disk operations and make metadata processing more efficient, both vital for handling large datasets. Haystack was built with a well-structured design that includes a distributed cache, storage methods, and directory services, which tackled common issues, improving throughput and fault tolerance while using inexpensive hardware [9], [17]. This background not only informed Haystack's design principles but also laid the groundwork for future scalable system models that could handle large amounts of data.

2.3. Comparison with traditional data storage systems

Traditional data storage systems often use NAS setups, which have problems with scalability and performance. Haystack's architecture smartly tackles these challenges with its unique store, directory, and cache features. Conventional systems often experience slowdowns due to issues with metadata handling and heavy disk I/O, making it hard to efficiently manage large amounts of data, like Facebook's huge photo collection. On the other hand, Haystack reduces the need for disk access and uses a distributed caching system to provide both high throughput and low latency, which are essential for real-time interactions. Additionally, by organizing storage into logical and physical volumes with strong redundancy, Haystack boosts fault tolerance, greatly enhancing reliability compared to traditional systems [9]. Overall, this architecture marks a clear change from old storage methods, making it a great choice for today's data-heavy applications [18].

2.4. Key features of Haystack architecture

A unique part of the Haystack design is its new way of improving photo storage and access, which is crucial for handling large digital collections. This system is made to use several physical volumes, each holding millions of images, to make sure it can grow and stay dependable. The way volumes are arranged in Haystack allows photos to be copied across different physical storage, greatly reducing the chance of losing data because of hardware issues. Additionally, by using a distributed cache and cutting down on disk operations, the design achieves a good mix of high speed and low delay, meeting the high demand from users accessing more than 260 billion photos. These improvements not only make access easier but also fit a budget-friendly approach for managing cloud data, highlighting the need to enhance storage solutions in today's data-driven environment. Overall, the Haystack design shows important features that tackle the challenges of large-scale data storage, being vital for platforms like Facebook, which need strong and effective systems for handling user-generated content.

2.5. Role of Haystack in Facebook's overall infrastructure

In Facebook's big tech environment, Haystack is key in solving problems with photo storage and retrieval, improving the site's overall performance and user experience. By making data management more efficient, Haystack helps reduce the slowdowns seen in regular storage systems, shown by its clever design that cuts down on disk operations and processing demands [9]. This big improvement is not just a tech upgrade; it aligns with Facebook's larger goal of efficiently managing its extensive operations. With Haystack, users can easily access over 260 billion photos quickly, which is crucial for keeping users engaged and satisfied. As Facebook works to upgrade its systems to support growing amounts of data, Haystack's role in maintaining fast and efficient operations shows its vital importance in the company's setup [19]. In the end, Haystack shows how focused solutions can improve the overall effectiveness of the infrastructure while providing a strong and scalable way to manage data storage.

3. DATA STORAGE MECHANISMS IN HAYSTACK

The storage methods used by Haystack are a key part of how it is built, aimed at handling Facebook's large photo collection while solving performance issues. A key feature of this setup is the split into physical volumes that store millions of photos, which are grouped into logical volumes for better backup. This approach helps reduce the chance of losing data from hardware problems, making sure it works well in a distributed setup, which is important for data access across different clouds and regions [20]. The system also includes a distributed cache to improve speed and lower delays, making data retrieval easier. Recent updates show that knowing memory structures is important, as it reveals the hidden costs related to data movement, a detail often missed by developers [16]. By distributing read and write tasks among its parts, Haystack demonstrates a flexible data storage solution that fulfills Facebook's large operational requirements [9]. Figure 3 depicts the photo-serving stack, which consists of four layers: browser caches, edge caches, origin cache, and Haystack backend.

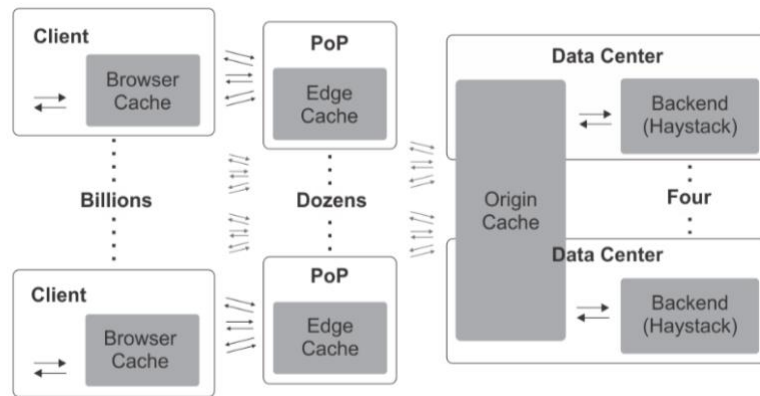


Figure 3. Data storage mechanisms in Haystack

Figure 3 illustrates the data storage mechanism in the Haystack system, which leverages various types of caches playing a critical role in enhancing overall system performance and efficiency. The browser cache stores webpage elements, such as images and other files, on the user's side, aiming to accelerate page load times and reduce server load by utilizing previously downloaded data. The pop-end (edge) cache operates on servers located close to users, such as those in a content delivery network (CDN), storing copies of data to reduce latency and improve access speed by delivering content from servers geographically closer to the user. Meanwhile, the origin cache functions on the origin server by storing frequently accessed data copies, designed to expedite data availability during requests and alleviate the burden on the primary storage server. Additionally, the backend cache is employed within backend systems, such as Haystack, to store large data objects, such as images and other media, with the purpose of accelerating data retrieval, reducing the load on slower storage systems, and enhancing data reliability and availability through replication and distributed storage mechanisms. All these caching mechanisms work synergistically to ensure fast, efficient, and reliable data access across all layers of the storage system. The interaction among different types of caches, browser, edge, origin, and backend supports large-scale data distribution optimization by minimizing latency and improving overall data access speed and efficiency.

3.1. Overview of data storage techniques used

In modern data storage systems, different methods are important for making things efficient, reliable, and scalable. One key method is using distributed storage setups, which use standard hardware to build a strong infrastructure that can handle large amounts of data. An example is Facebook's Haystack system, which effectively deals with the issues of storing a lot of photos. It does this by using both physical and logical volume management, along with redundancy methods to avoid data loss. By splitting photos into several volumes and using a distributed cache for quick access, Haystack greatly enhances read and write processes, leading to better performance and system strength [9]. Additionally, ongoing background tasks in this system keep an eye on potential problems and fix them, thus maintaining data integrity and smooth operations [11]. This varied approach to data storage highlights the need for creative design in dealing with today's digital challenges, especially in areas with rapid data growth.

3.2. File storage and management in Haystack

Efficient file storage and management in Haystack's system helps lessen problems of dealing with large data amounts such as Facebook's big photo library. By organizing storage into many physical volumes, each holding millions of photos, Haystack provides an orderly and strong way to handle data. Key parts of the system include the Haystack store, directory, and cache, which work together to improve data access. Redundancy methods in physical volumes protect against data loss from hardware issues, showing a forward-thinking approach to file management. Each logical volume is designed to duplicate photos across physical storage units, making it more reliable and able to resist faults [9]. Also, Haystack allows for quick access to images, improving user experience while reducing disk operations and metadata load [16]. These innovations highlight how important Haystack is in supporting Facebook's ability to manage and store data smoothly.

3.3. Data replication strategies employed

Reliability in data storage systems greatly depends on good replication methods, particularly in large setups like Facebook's Haystack. At the center of Haystack's design is a strong data replication system that

Haystack-based Facebook's data storage architecture: store, directory, and cache (Tole Sutikno)

provides redundancy over several physical volumes, which helps prevent data loss from hardware failures. Each volume holds millions of photos, arranged into logical groups for better access and maintenance, while also improving reliability through careful duplication. This method reduces the risk of single points of failure and enhances read and write processes, allowing requests to be spread out across different replicas. Haystack uses standard hardware to carry out these methods, enabling cost-effective growth while ensuring high throughput and low delays in metadata handling and access. With ongoing background tasks focused on checking and fixing data integrity problems, Haystack serves as a solid replication model that values both performance and reliability [11].

3.4. Scalability considerations in data storage

In data storage, thinking about scalability is very important for handling the rapid increase of user-generated content, especially on platforms like Facebook, which deals with billions of photos every day. The design used in Haystack shows important ways to achieve scalability, like removing bottlenecks found in traditional NAS systems and using a distributed caching system that improves both speed and response time. By spreading data over several physical volumes, each with millions of photos, Haystack allows the system to handle large data sets effectively while keeping backups to guard against hardware issues. This design supports reliability and fault tolerance since background tasks keep an eye on and fix possible problems, ensuring the system works well without slowing down [9]. As the need for storage grows, the insights from Haystack's approach highlight the need for a strong, scalable framework that can change to meet the needs of data-heavy applications [11].

3.5. Performance metrics for data storage efficiency

Knowing how to measure data storage efficiency is important for assessing systems like Facebook's Haystack. There are main metrics to consider, like throughput, latency, and redundancy levels, which together analyze how well data is stored and retrieved. Throughput shows the speed of reading or writing data, impacting user experience and system performance. Latency indicates how long it takes to get data, a delay that can be improved with good caching techniques found in Haystack, leading to better performance. Additionally, metrics on redundancy are crucial for keeping data safe and available; the replication methods used in Haystack help prevent data loss from hardware issues, increasing reliability [9]. By balancing these measurements, system designers can ensure that the data storage setup meets current needs and can grow efficiently for more users [12].

4. DIRECTORY SERVICES IN HAYSTACK

Integrating directory services in the Haystack structure is important for improving how data is retrieved and ensuring good access to the large amounts of photos stored on Facebook's system. The directory acts as a link between data storage and client requests, helping to quickly find physical volumes that contain particular photo data, which reduces waiting times. This setup meets the system's main aim of making operations smoother by lowering the computing load linked to metadata processing, which is a big upgrade compared to older NAS options. Also, the directory uses a logical layer that boosts fault tolerance, making sure that even if there is a hardware problem, data can still be accessed and is secure. By using these directory services, Haystack not only improves efficiency but also makes better use of resources, which is important given the rising demands for cross-cloud data access seen in recent studies [20]. Furthermore, the directory's role fits with current cloud optimization trends, as highlighted in recent studies that stress the need for flexible systems that can lower resource consumption while keeping up performance [21]. Hence, directory services in Haystack are crucial for providing a flexible and strong data storage solution, confirming its significance in today's cloud computing strategies.

4.1. Functionality of directory services in Haystack

The effectiveness of data management in Haystack largely depends on its directory services, which are crucial for navigating the many stored items. The directory serves as a metadata layer that effectively links user searches to actual storage locations, thus lessening the usual burdens related to metadata searches. Unlike standard directory systems, Haystack's directory is made to cut down on disk operations, resulting in much better performance for reading and writing. Each entry in the directory is carefully crafted to handle high demand for speed and low wait times, meeting Facebook's needs for quickly accessing and delivering a vast number of photos. Additionally, the smooth integration of the directory with the Haystack store and cache systems [9] guarantees that retrieving data is both fast and reliable, supporting the overall goal of reducing costs while remaining scalable [20]. Through its unique directory services, Haystack shows how smart design can help solve data access issues and improve the efficiency of storage systems.

4.2. Data indexing and retrieval processes

Data indexing and retrieval are very important for large storage systems like Facebook's Haystack, which helps manage billions of photos. Haystack's structure includes special indexing methods that make retrieval faster and cut down on slow disk operations. By combining how data is logically and physically organized, Haystack keeps data stored well and easy to get when users need it. The system aims to lower the workload on metadata processing to improve performance, enabling quick access without stressing the storage hardware. Additionally, it uses a distributed cache system to ensure fast data retrieval with high throughput and low delays, which also helps the system resist faults [9], [22]. Overall, these advanced indexing and retrieval methods are crucial for keeping Facebook's large photo storage system responsive and reliable.

4.3. Role of metadata in directory services

In modern directory services, metadata plays an important role in making data retrieval systems work better. Providing information about the data-like where it comes from, its layout, and how it connects to other data-metadata, helps users access resources quickly and supports data integrity and management efforts. For example, in Facebook's Haystack system, managing metadata is key to sorting through large photo collections stored in different physical and logical locations, which helps prevent data loss from hardware failures. This method, where metadata functions as a main directory, assists in fast lookups and keeps a high rate of data processing while cutting down on delays in accessing data. Additionally, using metadata effectively helps to optimize storage and enhance fault tolerance since it enables background processes to check for and fix issues early, thus improving overall system reliability [9], [11]. If there were no solid metadata systems, the efficiency and growth of directory services would be seriously affected, negatively impacting user experience and company productivity.

4.4. Challenges in directory management

Managing directories in big data storage systems, like Facebook's Haystack, comes with many challenges that need handling for system efficiency and reliability. A major problem is the need for quick access to metadata, which is essential for organizing and retrieving large data sets. With more than 260 billion stored photos in Haystack, the directory must handle and index this data well to allow fast access, reducing delays and boosting performance. Also, keeping data consistent and ensuring integrity across many physical and logical volumes gets trickier as directory sizes grow; system failures can threaten this integrity if strong backup methods are not in place [9]. Moreover, the changing nature of user actions, such as uploads and deletions, requires ongoing updates to the directory, making management more complicated and needing advanced algorithms to maintain load and performance [23].

4.5. Innovations in directory services within Haystack

The improvements in directory services in the Haystack architecture greatly improve how photo storage is managed at Facebook. The directory, which is crucial for finding and retrieving images, is carefully created to reduce latency and increase throughput. By using a distributed setup, the directory service manages millions of photos while providing redundancy across physical volumes to prevent data loss, a key point in the Haystack model. Additionally, the directory includes background tasks that support early detection and fixing of failures, which enhances the strength of the whole system [9]. This architecture's ability to balance read and write tasks not only makes it easier to access stored images but also helps with cost-effective growth, meeting the needs of an expanding user base [11]. Therefore, the new method used in Haystack's directory services shows advanced techniques in the design of distributed systems.

5. CACHING STRATEGIES IN HAYSTACK

The use of caching methods in the Haystack architecture is essential for improving performance and user experience in Facebook's large photo storage system. By using a distributed cache, the architecture decreases the number of disk operations, which helps reduce delays from metadata processing, as mentioned in [9]. This decrease is important, considering the high number of photo retrieval requests that Haystack deals with each day. The addition of a cache not only allows quick access to popular images but also helps manage read and write operations throughout the system. Moreover, the design of the architecture focuses on cost-effectiveness while ensuring reliability, highlighting its efficiency. Overall, the caching methods in Haystack represent a smart way to manage data, which is vital for Facebook's goal of providing smooth photo access on a large scale [24], [25].

5.1. Importance of caching in data retrieval

In data retrieval systems that are complex, caching is an important tool that helps with performance and efficiency. Caching cuts down on access delays by temporarily saving data that is asked for often,

reducing the load on main storage systems. This can be seen in systems like Facebook's Haystack, which needs fast and dependable access to a vast number of images. By using a distributed cache, Haystack gets high performance while keeping low delays, effectively managing the important read and write tasks while making sure users can access content [9]. Additionally, caching plays a big role in lessening the total load on storage systems behind the scenes, which lets tasks like metadata handling happen more quickly [11]. Therefore, good caching methods not only speed up data retrieval but also improve how scalable, reliable, and fault-tolerant modern distributed systems are.

5.2. Types of caching mechanisms used

Caching methods are important for improving how quickly data can be retrieved in systems like Facebook's Haystack, which needs fast access to stored images. One type of caching, called distributed caching, helps decrease delays by keeping often-used data on many nodes, making user requests quicker. Also, in setups similar to those mentioned in [20], combining both dynamic random-access memory (DRAM) and object storage in caching systems helps balance speed and cost. These kinds of hybrid caching approaches show how data can be managed well to adjust to changing demand, emphasizing the need for flexible cache size and type. As cloud applications change often, new techniques like memory optimization, including ideas from evolutionary algorithms mentioned in [21], show possibilities for better resource management without hurting system performance. These methods are key to keeping large-scale data systems working well and accessible while dealing with the complexities of modern data structures.

5.3. Cache consistency and invalidation techniques

In systems that store data across many locations, like Haystack, keeping cache consistent and handling invalidation methods well is very important for data correctness and performance. When many clients use the same data at the same time, problems can come up, which may result in outdated or incorrect data being shown. To solve this, methods such as write-through and write-back caching can be used, with each having distinct pros and cons related to speed and the use of resources. Also, there are strategies for invalidation that are either proactive or reactive, which are vital for managing the cache. Proactive invalidation updates or deletes cached data ahead of time based on set rules, whereas reactive strategies rely on alerts that occur when data changes, thus helping with bandwidth use and lowering the costs of keeping cache consistent. Building a strong caching system, like the one in Haystack, not only boosts performance and reduces delay but also helps tackle the issues that come with ensuring reliable access to large data collections, backing the system's main aim of being efficient and scalable [11].

5.4. Impact of caching on system performance

Caching methods are very important for making systems run better by lowering delay and using resources more efficiently. By keeping data that is accessed often in a quick-access cache, applications can cut down on the extra work that comes with reading from disks and boost overall performance. For example, in Facebook's Haystack setup, adding a distributed cache helped the system send out over 260 billion photos with little delay, fixing big problems found in older NAS. This smart caching use not only speeds up data access but also reduces pressure on backend storage tools, leading to better response times and user satisfaction [9]. Furthermore, a good caching plan helps create a better balance between reading and writing data, which increases reliability and operational effectiveness [13]. Therefore, using caching is key for strong data handling in large, spread-out systems.

5.5. Future trends in caching strategies for Haystack

The need for effective data access is growing, and future trends in caching for Haystack need to consider the changing storage solution landscape. One useful approach is using artificial intelligence and machine learning to improve cache management. This would let us analyze data access patterns in real time and fill caches proactively. This kind of predictive caching could lower latency and boost throughput, which fits well with Haystack's current design that supports billions of photos while cutting down on disk operations and metadata handling. In addition, using advanced distributed caching methods, like multi-tier architectures, can help spread out the load among nodes, making the system more fault-tolerant and resilient to hardware issues [9]. Lastly, looking into edge computing could enhance Haystack's abilities by placing cached resources nearer to users for quicker data access, leading to a more efficient system overall [26].

6. CONCLUSION

In finishing the look at Facebook's Haystack-based data storage system, it is clear that this new setup marks a big step forward in managing data at scale. The architecture successfully deals with the issues

caused by the huge amount of user-created content, particularly in relation to storing and finding photos. By spreading storage over different physical volumes and using a strong caching system, Haystack reduces metadata load and boosts throughput, which helps in accessing billions of photos. Also, the use of proactive background tasks to check and fix system failures shows the architecture's focus on being reliable and able to handle faults. These features not only improve how well it works but also help keep costs down, which is part of a bigger trend in cloud computing toward using cheap, regular hardware. In the end, Haystack stands out as a strong example of using smart architectural methods to satisfy the changing needs of applications that use a lot of data. Facebook's Haystack data storage system is a groundbreaking solution for managing large amounts of digital media. Its design combines storage, directory, and cache parts, resulting in higher data flow and less waiting time for accessing photos. The system reduces disk use and improves metadata handling, addressing delays in older network-attached storage systems. The system also employs robust backup methods to ensure data safety, regularly copying millions of photos to prevent hardware issues. The Haystack architecture significantly enhances distributed systems design, setting a standard for future advancements in data storage methods. It outperforms old methods like NAS, which can face slowdowns and issues under heavy use. The combination of the Haystack store, directory, and cache makes photo retrieval easier, increasing speed and lowering delays. This system meets today's requirements for quick data access in cloud computing, highlighting the need for flexible caching methods. However, there are shortcomings in current research, such as the lack of real-world testing on the system's scalability under heavy usage and the complexities of various data types and access behaviors in a fast-changing social media environment. Future studies should explore better designs for big storage systems, focusing on new methods that reduce metadata processing, artificial intelligence for fault detection and repairs, and the cost effects of distributed caches. The shift from old data storage models to new systems highlights the need for flexible solutions that can solve today's storage problems and prepare for future requirements as data grows. Focusing on backup measures and preventive upkeep in these systems helps protect against hardware issues, ensuring data reliability and availability in more complex digital environments.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to Universitas Ahmad Dahlan (UAD), the Embedded System and Power Electronics Research Group (ESPERG), Universitas Palembang, and International Islamic University Malaysia (IIUM) for their invaluable support and contributions to this research. The facilities, guidance, and collaborative environment provided by these institutions were essential to the successful completion and publication of this work.

FUNDING INFORMATION

The research was funded by PT. Intelektual Pustaka Media Utama (IPMU) under contract number 09/RST-E/IPMU/I/2024, which supported the facilitation of this work.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Tole Sutikno	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Ahmad Heryanto		✓		✓	✓	✓		✓		✓				
Laksana Talenta Ahmad		✓		✓		✓				✓			✓	

C : **C**onceptualization

M : **M**ethodology

So : **S**oftware

Va : **V**alidation

Fo : **F**ormal analysis

I : **I**nvestigation

R : **R**esources

D : **D**ata Curation

O : Writing - **O**riginal Draft

E : Writing - Review & **E**ding

Vi : **V**isualization

Su : **S**upervision

P : **P**roject administration

Fu : **F**unding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

This study does not involve direct interaction with individuals. Therefore, informed consent is not necessary, as the research relies on publicly available data.

ETHICAL APPROVAL

This study does not include any human or animal participants; therefore, ethical approval is not required for this article.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.




REFERENCES

- [1] S. Muralidhar *et al.*, “F4: Facebook’s warm blob storage system,” in *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2014*, Facebook Inc.: USENIX Association, 2014, pp. 383–398. [Online]. Available: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85075667407&partnerID=40&md5=e9934c5a1b372cb2303fa5729c992512>
- [2] M. V. Barbera, S. Bronzini, A. Mei, and V. C. Perta, “A needle in the haystack-delay based user identification in cellular networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Sapienza University, Rome, Italy: Springer Verlag, 2014, pp. 265–267. doi: 10.1007/978-3-319-04918-2_27.
- [3] A. R. Sætнан, “The Haystack fallacy, or why big data provides little security,” in *The Politics of Big Data: Big Data, Big Brother?*, Norwegian University of Science and Technology, Trondheim, Norway: Taylor and Francis, 2018, pp. 21–38. doi: 10.4324/9781315231938-2.
- [4] J. McCormick, *Nine algorithms that changed the future: the ingenious ideas that drive today’s computers*. Dickinson College, United States: Princeton University Press, 2020. doi: 10.5860/choice.49-5106.
- [5] J. Yin, H. Zhu, and P. C. Vinh, “Formalization and analysis of Haystack architecture from process algebra perspective,” *Mobile Networks and Applications*, vol. 25, no. 3, pp. 1125–1139, 2020, doi: 10.1007/s11036-019-01433-1.
- [6] D. Beaver, S. Kumar, H. C. Li, J. Sobel, and P. Vajgel, “Finding a needle in Haystack: Facebook’s photo storage,” in *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2010*, Facebook Inc., United States: USENIX Association, 2019, pp. 47–60. [Online]. Available: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85076926134&partnerID=40&md5=c5b6b2216dabf1e3748392c69d237b5e>
- [7] S. Comfort *et al.*, “Sorting through the safety data Haystack: using machine learning to identify individual case safety reports in social-digital media,” *Drug Safety*, vol. 41, no. 6, pp. 579–590, 2018, doi: 10.1007/s40264-018-0641-7.
- [8] D. Cunha, N. Guimarães, and A. Figueira, “An architecture for a continuous and exploratory analysis on social media,” in *Proceedings of the International Conferences on Computer Graphics, Visualization, Computer Vision and Image Processing 2017 and Big Data Analytics, Data Mining and Computational Intelligence 2017 - Part of the Multi Conference on Computer Science and Info*, R. L., X. Y., and A. A.P., Eds., CRACS / INESC TEC and Universidade Do Porto, Rua do Campo Alegre 1021/1055, Porto, 4169-007, Portugal: IADIS, 2017, pp. 339–342. [Online]. Available: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85040191575&partnerID=40&md5=bfe7796f7223a45e3bc65dff1fa5a929>
- [9] Shiyang, “Facebook’s photo storage architecture.” 2024. [Online]. Available: <https://shorturl.at/8XVzY>
- [10] T. N. Mouratidis, “Optimizing the recovery of data consistency gossip algorithms on distributed object-store systems (ceph) βελτιστοποίηση αλγορίθμου φλωρίας για συνεπή δεδομένα σε κατανεμημένα συστήματα αποθήκευσης αντικειμένων (ceph),” National and Kapodistrian University of Athens, 2020.
- [11] L. George, *HBase the definitive guide. random access to your planet-size data*. O’Reilly Media, 2011. [Online]. Available: <https://books.google.co.id/books?id=nUhiQxUXVpMC>
- [12] B. Ellis, *Real-time analytics: techniques to analyze and visualize streaming data*. Wiley, 2014. [Online]. Available: <https://books.google.co.id/books?id=DFnOAwAAQBAJ>
- [13] M. J. Franklin, *Client data caching: a foundation for high performance object database systems*. in The Springer International Series in Engineering and Computer Science. Springer US, 2011. [Online]. Available: <https://books.google.co.id/books?id=drcKswEACAAJ>
- [14] M. H. Fu *et al.*, “Study and implementation of knowledge-based innovative social media services,” *ICIC Express Letters*, vol. 9, no. 3, pp. 759–765, 2015, [Online]. Available: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84924067908&partnerID=40&md5=3298373e92d8299b8e4c4230ddb01d88>
- [15] K. R. Krish, A. Anwar, and A. R. Butt, “HatS: a heterogeneity-aware tiered storage for hadoop,” in *Proceedings - 14th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2014*, Department of Computer Science, Virginia Tech, Blacksburg, VA, United States: IEEE Computer Society, 2014, pp. 502–511. doi: 10.1109/CCGrid.2014.51.
- [16] P. Fompeyrine, “Cache model plugin for memory hierarchy aware programming,” ETH Zurich, 2020. doi: <https://doi.org/10.3929/ethz-b-000431675>.
- [17] S. Cha, R. N. Taylor, and K. Kang, *Handbook of software engineering*. Cham: Springer International Publishing, 2019. doi: 10.1007/978-3-030-00262-6.
- [18] K. D. McCracken, *Flexible and robust data storage and retrieval in the haystack system*. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2001. [Online]. Available: <https://books.google.co.id/books?id=0bhLOAAACAAJ>
- [19] A. Simeunovic and U. Tripunovic, “Managing micro frontends across multiple tech stacks - sharing, finding and publishing,” Lund University, 2023. [Online]. Available: <https://lup.lub.lu.se/student-papers/search/publication/9128849>




- [20] H. Park, Z. Qiu, G. R. Ganger, and G. Amvrosiadis, "Reducing cross-cloud/region costs with the auto-configuring macaron cache," in *30th ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*, ACM Digital Library, 2024. doi: <https://doi.org/10.1145/3694715.3695972>.
- [21] K. Sumalatha and M. S. Anbarasi, "A review on various optimization techniques of resource provisioning in cloud computing," *International Journal of Electrical and Computer Engineering*, vol. 9, no. 1, p. 629, Feb. 2019, doi: 10.11591/ijece.v9i1.pp629-634.
- [22] H. Decker, L. Lhotská, S. Link, M. Spies, and R. R. Wagner, Eds., *Database and expert systems applications*, vol. 8645. in *Lecture Notes in Computer Science*, vol. 8645. Cham: Springer International Publishing, 2014. doi: 10.1007/978-3-319-10085-2.
- [23] Á. Rocha, A. M. Correia, S. Costanzo, and L. P. Reis, *New contributions in information systems and technologies*, vol. 354. in *Advances in Intelligent Systems and Computing*, vol. 354. Cham: Springer International Publishing, 2015. doi: 10.1007/978-3-319-16528-8.
- [24] M. Varade and V. Jethani, "Distributed metadata management scheme in HDFS," *International Journal of Scientific and Research Publications*, vol. 3, no. 5, pp. 1–5, 2013, [Online]. Available: <https://www.ijsrp.org/research-paper-0513/ijsrp-p1770.pdf>
- [25] G. Kolev, G. Markarian, N. Polushkina, L. Petersen, and G. Havârneanu, "Deliverable d4.3 – developed modular app for practitioners," 2023. [Online]. Available: <https://bit.ly/d43modular>
- [26] M. Chen, S. Mao, Y. Zhang, and V. C. M. Leung, *Big data*. in *SpringerBriefs in Computer Science*. Cham: Springer International Publishing, 2014. doi: 10.1007/978-3-319-06245-7.

BIOGRAPHIES OF AUTHORS






Tole Sutikno    has been a professor at Universitas Ahmad Dahlan (UAD) in Yogyakarta, Indonesia, since July 2023, following a tenure as an associate professor that began in July 2008. He obtained his B.Eng., M.Eng., and Ph.D. degrees in Electrical Engineering from Universitas Diponegoro, Universitas Gadjah Mada, and Universiti Teknologi Malaysia in 1999, 2004, and 2016, respectively. As of 2024, he has taken on the role of head of the Master's Program in Electrical Engineering at Universitas Ahmad Dahlan after previously serving as head of the Undergraduate Program in Electrical Engineering from 2022 to 2024 at the same institution. In addition, he leads the Embedded Systems and Power Electronics Research Group. His teaching responsibilities encompass both the undergraduate and master's degrees in electrical engineering, as well as the doctoral program in informatics at Universitas Ahmad Dahlan. His research interests span a wide array of topics, including digital design, industrial applications, industrial electronics, industrial informatics, power electronics, motor drives, renewable energy, FPGA applications, embedded systems, robotics and automation, artificial intelligence, intelligent systems, information technology, information systems, and digital libraries. He can be reached via email at tole@te.uad.ac.id.



Ahmad Heryanto    earned his doctorate (Dr.) in informatics from Universitas Sriwijaya in Palembang, Indonesia, in 2025. He also holds a Master of Engineering (M.Eng.) degree in Electrical Engineering from Institut Teknologi Sepuluh Nopember (ITS) in Surabaya, Indonesia, which he received in 2014. Prior to that, he completed his bachelor's degree in computer science at Universitas Sriwijaya and obtained his professional engineering designation (Ir.) in informatics from Universitas Hasanuddin. Additionally, he is a MikroTik Certified Trainer with over 20 years of experience in Linux, Windows, Cisco, and MikroTik. He also holds the Certified Penetration Testing Professional (CPENT) designation and several certifications related to internetworking, routing, and wireless communication. His research interests encompass parallel processing, distributed computing, software security, and network intrusion detection. He can be contacted at email: hery@unsri.ac.id.



Laksana Talenta Ahmad    is a researcher at the Embedded Systems and Power Electronics Research Group (ESPERG) in Yogyakarta, Indonesia, and a student in the Bachelor of Information Technology (BIT) program, Department of Information Systems (DIS), Kuliyah of Information and Communication Technology (KICT) at the International Islamic University Malaysia (IIUM) in Kuala Lumpur, Malaysia. His research interests encompass various aspects of information technology, including organizational informatics, database programming, robotics, artificial intelligence, and more. He can be contacted at email: laksanatadz@gmail.com.