Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Architectural Design for Data Security in Cloud-based Big Data Systems

***Mujeeb-ur-Rehman Jamali* *[1], *Najma Imtiaz Ali*[2], *Abdul Ghafoor Memon*[1], *Mujeeb-u-Rehman Maree*[2], *Aadil Jamali*[2]**

[1]Emaan Institute of Management & Sciences, Karachi Sindh, Pakistan
[2]Institute of Mathematics and Computer Science, University of Sindh, Jamshoro, Pakistan.
*Corresponding Author.

Received 10/03/2023, Revised 05/11/2023, Accepted 07/11/2023, Published Online First 20/02/2024

Abstract

Architecture design of system is creative process after all. Non-functional requirements and software architecture have a strong relationship. Security of system is one of great concern and it is a non-functional need and requirement. The architectural style and structure chosen for a system should be based on the non-functional system requirements. Big Data technologies are used to store handles huge amounts of operational data in the public domain. Structure and behavior diagrams of the system is proposed that use a secure architectural design for a system that works over the cloud. Different components of Big Data have been investigated in this research work due to the seriousness of security issues focus on privacy and confidentiality. The security of data over the cloud being outsourced is a growing concern. There are outsider and insider malicious database administrators who can gain access and modify private and sensitive data therefore database service provider can be deceitful. Security mechanisms are necessarily required to protect sensitive data residing on public clouds. At the application level, the proposed solution ensures privacy and confidentiality of data at rest over the cloud. Extensive tests have been carried out, and the results of this research work are as follows that the recorded results revealed statistically significant variations in plaintext and multimedia contents (i.e., audio and video) with time measured in milliseconds for small, medium, and large datasets..

Keywords: Big Data, Cloud Computing, Cryptography, Security, Software Architecture.

Introduction

The performance, safety, security, availability, robustness, and maintainability of a system are all influenced by the software architecture. Individual components carry out the requirements of the functioning system. The system design and security of the most crucial system determine non-functional needs. A software system's architecture may be based on a certain architectural pattern or style i.e., Repository architecture, Pipe and Filter architecture, Client–Server architecture, or layered architecture are examples of architectural patterns. Architectural patterns are encapsulated architectures that have been used in a number of software systems. Each pattern has its own set of strengths and weaknesses that influence the system's design. The layered architecture pattern in organizational design is a method of establishing isolation and independence. In a system where vast volumes of data are structured around a common database, the repository architectural pattern is used. Client–server

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

architecture systems are made up of a set of services and corresponding servers, as well as clients who consume the services. The Pipe and Filter architecture is a run-time system organization paradigm in which functional transformations take inputs and produce outputs, and data is passed from one transformation to the next and changed as it moves through the sequence. In most context models, the environment is presented. System boundaries are established early in the system definition process. The functionality of system and environment are defined in this stage with negotiation with stakeholders. However, it conceals the types of interactions that occur between the environment's systems and the system being specified. External systems may contribute or consume data from the system. They may communicate with the system directly, via a network, or not at all.

Graphical representations of software systems are available. It is part of design process as well as requirement engineering. Abstract model is a process of depict various perspective of system and representation of system as graphically in the system modelling. The system needs are defined during the process in which model are used. This phase of design requirement engineering for implementation of system. Models of the current and future systems can be created. During requirement engineering, model of the current system is employed. The system merit and flaws are highlighted, therefore, requirement of new system is emerged. The proposed system demands are explained to other system stakeholders using models of the new system. These models are used by engineers to discuss design concepts and to define the system for subsequent use. A model-driven engineering technique can be used to generate a whole or partial system implementation from a system model. Model-driven engineering is a type of engineering in which a system is created automatically using structural and behavioral models. There are a variety of models that may be used to portray the system from various angles. An external perspective in which model the system's context or environment. Model the interactions between a system and its surroundings or between its components from an interaction perspective. Model the organization of a system or the structure of the data processed by the system from a structural perspective. A behavioral viewpoint is one in which model the system's dynamic behavior and how it reacts to events. There is a growing demand for new software development patterns. The goal of component-based engineering is to create system software by choosing well-defined software components that are rarely used and assembling them according to a predetermined system architecture. The way software is developed is very different from how it was done in the past. System architecture design encompasses the analysis, choice, and development of software architecture for component-based systems. Collecting user requirements, identifying system specifications, choosing appropriate systems, and figuring out implementation specifics are the main goals of system architecture design. The choice of one architecture over another is the result of system architecture deign[1].

In this research work, different aspects of Big Data were studied and analyzed, due to the severity of security issues especially the confidentiality, integrity, and authentication. Therefore, innovation in the design of a new system for resolving security issues using the novel system has been proposed. The proposed system provides authentication of users and security of data at the application level. The system also ensures the protection of private and sensitive data by enforcing data privacy, confidentiality, and integrity in data at rest in the document database server. Our developed system namely BigDocumentDBCrypto provides application-level data protection of plaintext and multimedia contents with varying key size according to demand and requirements of user. The synthesis of results reveals which help to choose security according to requirement.

Cloud Computing

Cloud computing refers to the pay-as-you-go distribution of IT resources and applications through the Internet on demand. The "cloud" gives fast access to flexible and low-cost IT resources, whether it's for operating apps that share data with millions of users or supporting vital corporate activities. Cloud computing eliminates the need for users to make big upfront hardware expenditures and spend a significant amount of time managing that gear. Users may rapidly access as many resources as they require while only paying for what they consume. Cloud

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

computing designs enable for the efficient operation of big, gigantic, and cost-effective computing. In cloud computing workloads are easier to manage, implement, and deploy, and they are also less expensive. Cloud storage may be used to store large amounts of data in a variety of formats, such as structured, semi-structured, and unstructured data. Cloud hosting is an Internet concept that allows users to access resources such as storage, infrastructure, and applications on a temporary, on-demand basis. It comes in a variety of forms, including public, private, and heterogeneous combinations of the two, and users may access services from anywhere and at any time. With the development of the cloud, an application utilizing an architecture that operates on a public and private cloud, supporting Big Users and Big Data. Rather of purchasing software, software as a service (SaaS) is used to operate it. Users can design, develop, and run applications in a platform service (IaaS) environment. Infrastructure as a service (IaaS) provides computational power and physical capacity, such as storage and space, on demand. Cloud Computing has a number of perks and advantages. New IT resources are just a click away in a cloud computing environment, reducing the time it takes to make them available[2].

The organization's agility increases tremendously due to that significantly reduce the cost as well as time. It also has the advantage of being able to deploy applications in different locations across the world with only a few mouse clicks. The service for massive-scale real-time processing of streaming data. Cloud hosting solutions give companies, non-profits, and government organizations' a flexible, highly scalable, and low-cost method to serve their websites and online applications; it's low-cost because it uses a pay-as-you-go pricing model and elastic scalability to match resources to user needs.

When it comes to infrastructural capacity, it removes the need for guessing. When deciding on capacity before installing an application, user might end up with either expensive idle resources or capacity restrictions. Cloud Computing solves these problems. Users have access to as much or as little as they require, and may scale up and down with only a few minutes' notice. The user can achieve a lower variable cost by using cloud computing. Because the cloud aggregates the use of hundreds of thousands of consumers. Instead of investing substantially in data centers and servers, services can gain better economies of scale, resulting in cheaper pay as you go costs. Users can only pay for computational resources when they use them, and only for the amount they use[3].

A public cloud is a third-party managed platform that makes resources and services available to distant users all over the world using the conventional cloud computing concept. To secure sensitive data stored in public clouds, security methods are essential. Cloud computing includes sensitive information, privacy and security concerns are only becoming worse. Concerns about security and privacy arise when information is outsourced to the cloud or a third party. These issues are now crucial when implementing cloud implementation and services. The wide range of uses for cloud computing raises questions about the security and privacy of data that is outsourced. Applications and data that are outsourced to the cloud have limitless security limits and infrastructure because of its dynamic abstraction and scalability.

Component Based Software Engineering

Designing a system using components makes logical. Interfaces define components, which are higher-level abstractions than objects. Components are generally bigger than individual objects, and they keep all implementation details concealed from other components. Independent components are entirely described through their interfaces, which are the foundations of component-based software engineering. The component interface and its implementation should be clearly distinguished. This implies that a component's implementation may be swapped out without affecting other elements of the system. There are standards for component integration. A component model encapsulates guidelines. Components specify how component interfaces should be specified and how components should communicate at the very least. Different programming languages' implemented components can be combined into a single system[4].

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Related Literature

The authors provide a formal model of Federated Identity Management (FIM) systems in terms of architectural design rewriting. FIMs provide cross-domain user authentication as part of the Circle of Trust concept, allowing access management between businesses (CoT). Because each pattern has different security and trust needs, FIM patterns have arisen as recurrent CoT situations. The goal of project to provide a formal framework for FIMs to express their patterns as architectural styles. In further detail, an architectural style is given to identify all possible legal CoT pattern configurations[5].

In terms of architectural design rewriting, the suggested model is described using style-consistent (graphical) designs. Model-driven engineering (MDE) is a software development technique in which the primary outputs of the development process are models rather than codes[6]. In the system UML model are used to describe the system design in the Model Driven Architecture. Model-driven engineering is based on the idea that totally automated model-to-code translation should be feasible[7, 8]. To do so, one needs to be able to create graphical models with well-defined semantics. There is also a need for a way to provide information about how the operations described in the model are implemented to graphical models. This may be done with a subset of UML 2 known as XML or Executable UML[9].

Big data, a phrase that has lately gained popularity and describes a sizable collection of very vast and complex data sets, is confronted with significant security and privacy issues. because of the big data's usual traits—namely, the velocity, volume, and variety—that are connected to expansive cloud infrastructures[10].

The document database's security problems are still present. Various connected works, investigations, and publications are extensively given and discussed in this part. The authors of this study argued that document databases keep data in unencrypted form in internal physical file storage. There is no encryption/decryption system offered by default. As a result, an active/passive intruder/attacker or unauthorized user can access and potentially get sensitive/valuable data. As a result, application-level encryption may be offered for explicit encryption and decryption of sensitive and private data stored in a document database[11,12].

Authors demonstrates that the current deployment of cloud-based apps raises severe issues about data security. According to the authors, developing a fully secured system is nearly difficult[13].

The author indicated that utilizing different key sizes of algorithm (i.e., DES and AES) can greatly reduce security concerns[14].

Authors showed that document database vulnerabilities are due to SQL injection and DoS attacks since encryption/decryption is not enabled by default. As a result, such security vulnerabilities may be minimized and addressed by employing symmetric algorithms such as DES, AES, and Blowfish. The empirically acquired results were subjected to a security examination. The average time was calculated for a well-known document database (MongoDB), which is a NoSQL database used to store unstructured data. Various datasets (A through E) of varying sizes were employed. Each operation for inserting and retrieving data from the document database was performed at least ten 10 times. The algorithms with the smallest key size, namely AES (128 bit), DES (64 bit), and Blowfish (64 bit), were employed to assess the findings. The encryption process of several symmetric algorithms with varying key sizes utilized encryption operation. It was discovered that AES algorithms with 128-bit key sizes conducted encryption operations on various dataset sizes in 139, 212, 249, 298, and 354 milliseconds. DES took 153, 292, 389, 449, and 505 milliseconds, whereas the Blowfish method with a 64-bit key size took 43, 69, 98, 108, and 139 milliseconds. It was discovered that for dataset results, the CPU execution time of the Blowfish method was faster than that of DES and AES. According to the data, Blowfish required less time to encrypt than DES and AES. Decryption operation with symmetric algorithms with varying key sizes. It was discovered that AES algorithms with 128-bit key sizes completed decryption operations on various sizes of datasets in 97, 153, 190, 246, and 286 milliseconds. DES took 119, 204, 284, 303, and 362 milliseconds, whereas the Blowfish algorithms with a 64-bit key size took 31, 46, 53, 73, and 87 milliseconds. It was discovered that for dataset results, the CPU execution time of the Blowfish method was faster than that of DES and AES. The

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

ciphertext generated by secret key symmetric algorithms (AES, DES, and Blowfish) for datasets of varying sizes. Because ciphertext takes up more space than plaintext, the author advised that compression techniques be employed to reduce storage capacity[15].

In[16] authors performed empirical study using secret key algorithms such as AES, DES, and Blowfish. The execution performance of the described algorithms was compared based on CPU time. There are two datasets (02), file sizes for small and big datasets range from 100 bytes to 1000 bytes and 10,000 bytes to 100,000 bytes, respectively. The author(s) discovered that the average execution time in milliseconds on small data text files was 198.5 (AES), 184.4 (DES), and 185.1 (DES) (Blowfish). Meanwhile, the average memory utilization on small data text files was 1,263.030 (AES), 1,271.082 (DES), and 1,258.205 (DES) (Blowfish) in millisecond. The average execution time in millisecond for big data text files was 167.00 (AES), 465.36 (DES), and 459.22 (Blowfish), whilst the average memory use was 1,309.141 (AES), 1,291.483 (DES), and 1,276,567 (Blowfish) in millisecond. Based on actual results of time limitations, the author (s) discovered that AES took 8% less time / was slower than DES, and similarly AES took 7% less time / was slower than Blowfish. As a result, DES was quicker than both methods for small dataset encryption. Meanwhile, for big datasets, three algorithms required almost the same time to encrypt and decrypt. In terms of memory, it was discovered that DES was 1.01 percent slower than Blowfish but used 0.63 percent less RAM to execute the AES algorithm. AES used 2.48 percent more memory than Blowfish for big datasets, and DES took 1.34 percent more RAM than AES. Blowfish outperformed other symmetric algorithms as a result. With time limits, DES may be a better alternative than Blowfish and AES for the use of algorithms. Blowfish may be superior than DES and AES for memory-constrained applications such as mobile devices such as smartphones and tablets. The authors suggested a future study area in which multimedia materials, such as pictures, audio, and video, as well as CPU execution time and memory constraints, may be examined.

The security flaws identified in[17] that well-known secret key algorithms experimental result suggests that the DES algorithm's efficiency/performance was superior than AES, as claimed by the author. Meanwhile, Blowfish performed well in terms of execution speed on small text data files (less than 1000 bytes in size). The results revealed that symmetric algorithm efficiency, i.e., AES, DES, and Blowfish algorithms, were almost equal on big text data files (size greater than 10,000 bytes). The author concluded that the testing results clearly suggest that Blowfish outperforms AES in encryption performance. Meanwhile, DES was superior for memory limitation applications that relied on memory requirements during execution. Blowfish can be used to achieve good/strong data security.

The comparison and study of symmetric algorithms was carried out in[18]. Evaluation of the performance of the DES, 3DES, AES, and Blowfish algorithms. The symmetric algorithms were utilized to create a block cypher with adjustable block sizes and keys. The authors concluded that when the block size is big, the result is faster owing to reduced execution time required to encrypt data. Meanwhile, if the block size is short, more time is required for execution. The authors determined that Blowfish needed the least amount of time and was the quickest when compared to other secret key algorithms, whereas 3DES required the more time for encryption and decryption and was the slowest algorithm.

In[19] authors presented a data concealment method based on audio steganography and cryptography to protect data transport between the source and destination. The audio medium is utilized for steganography, and an LSB (Least Significant Bit) technique is used to encode the message within the audio file. The system was assessed for efficacy, and the results demonstrate that the encryption and decryption methods used to construct the system make the security of the proposed system more efficient in safeguarding data from unwanted access.

In[20] authors created lightweight cryptography (LWC) for improving existing cryptographic methods and developing new ones. Because of the large number of research, there has been an increase in the number of review on LWC in IoT. Further authors concentrated on symmetric block cryptography, with just a few reviews on asymmetric-key and hash in LWC. The study's findings showed that reviews in LWC in IoT are still in their early stages, and researchers are urged to

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

investigate by performing review studies in underserved regions. According to the authors, additional empirical study in the domain of LWC for IoT is needed.

In[21] researchers presented a novel encryption technology. It generates hybrid keys by combining two encryption methods, DES and AES. The authors suggested that this combination boosts the proposed W-method by creating highly randomized keys. As compared to typical DES and AES algorithms, the suggested technique works quicker and more securely.

The various authors presented their work of vulnerabilities of the security issues of Big Data on document database over cloud environment. Authors provided encryption data at rest with symmetric key cryptography with secret key for encryption and decryption. The proposed work provides a different approach from already proposed by other researches using application-level security instead of data at rest. encryption. The application-level provides encryption of the sensitive attributes of a document in the database collection. It is independent of the database and provides complete control of data. Through literature, it was revealed that already existing systems are according to that era where today enhanced security problems that need to be solved through enhanced techniques.

Hardware and Software Requirement

The partial system was created and the experiment was carried out to assess the aims and objectives. Architecture description languages (Papyrus), Eclipse Modelling Neon, and Acme Studio were among the software needs for developing the proposed system's architecture. The well-known document database i.e., MongoDB 3.4 is used. Java (1.9) utilized for implementation with third-party cryptography provider for libraries of JCA and JCC.

Materials and Methods

UML Component, class, and sequence diagrams are utilized in the proposed system for presenting architectural models. UML contains diagram that may be used to create a wide range of system models. According to UML, might illustrate the basics of a system. Data processing activity is depicted using the UML activity diagram. The system interaction with its surrounding is demonstrated by use case diagram. The sequence diagram demonstrates the interaction of components of system as well as actors. Sequence diagrams are used to represent interactions between system components. In the UML, sequence diagrams are used to describe interactions between actors and objects in a system, as well as interactions among objects themselves. Class diagrams depict the system's object classes and their relationships. State diagrams depict how a system responds to both internal and external events. Interaction models in the proposed system imply some form of interaction. User interaction, which involves inputs and outputs from the user. interaction between the system under development and external systems, and interaction between the system's components. Using case modelling is a technique for simulating interactions between a system and its external actors (users or other systems). Structural models of a proposed system show how a system is organized in terms of the components that make it up and how they interact. Static models describe the structure of the system architecture, whereas dynamic models demonstrate how the system works.

In the development of system using UML object-oriented pattern the classes and their relationship in the system are used. An association is a connection between classes that shows a link between them. Aggregation is a specific sort of class connection provided by the UML, which indicates that one object (the whole) is made up of other objects (the parts). The dynamic behavior of the system is demonstrated at run time system with external stimuli using the behavior model.

Software assistance for component integration is provided by middleware. To make dispersed, self-contained components communicate with one another. Component communication is supported and managed by middleware. Resource allocation, transaction management, security, and concurrency may also be supported by component support middleware. Component-based software engineering is the focus of this development method. Allowing requirements to vary during the development process is necessary, depending on the capability of accessible components. The implementation of the component can be altered without having an impact

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

on the rest of the system. Interfaces determine how components communicate. If these interfaces are preserved, one component can be replaced with another that offers more or better functionality. Component infrastructures provide a set of common services that application systems can use.

System Design

The proposed system's high-level architecture is shown in Fig 1. It highlights some of the key tasks that the CA Server, CSP Server, and CSU Application are responsible for carrying out. The system is designed namely BigDocumentDBCrypto in which configuration of components and connections is represented by systems. The proposed framework's architecture is designed to ensure privacy and data integrity when interacting with and storing Big Data on the public cloud, where the user is unaware of where the data is stored and may be accessed from anywhere. The system is made up with components that is Document Database, Middleware, Certificate Authority (CA) Server and User, which will be discussed in more detail in the next sections. The CA Component ensures that the user's authenticity is verified in the system. Meanwhile, Middleware completed the necessary transformations, and the database server did not require any changes. The cryptographic's computational work is done by middleware. Users communicate with the CA Server and the Middleware. Middleware connects to a cloud-based non-relational document database management system[22,23].

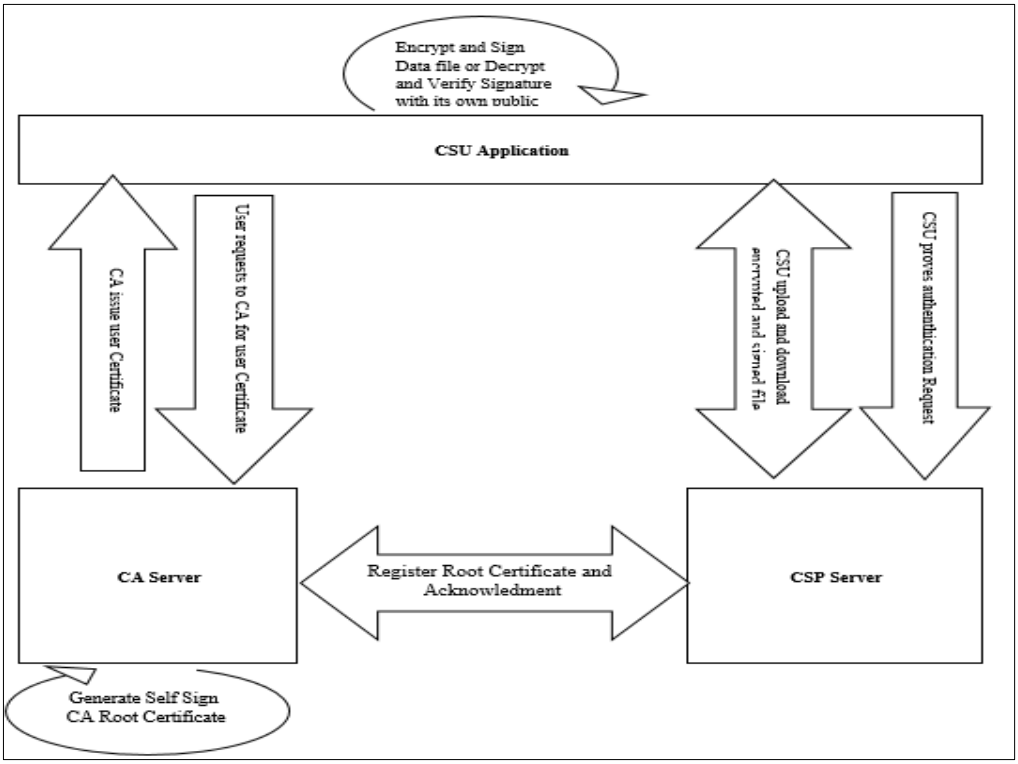

Figure 1. High-level architecture of Proposed Model.

Proposed Method

Data Flow

Step:1

Initially, a user requests that a digital certificate be generated by the CA Server. The user must register with the CA Server during the registration process by providing identifying information. The CA offers the user a password-protected digital certificate.

Step:2

After proving authentication, the user is permitted to conduct CRUD (Create, Read/Projection, Update, and Delete) activities.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Step:3

As a user security requirement, middleware encrypts and signs sensitive document fields such as (PII) using symmetric and asymmetric cryptography with changing key size. Middleware initially encrypts the sensitive field of the document before calculating the hash value (i.e., the hash value of any given field remains constant regardless of the length of the original plaintext). Following that, middleware converts the encrypted data and hash value and forwards / retrieves to and from the cloud-based document database. The process of recalculating the Hash Value to validate the authenticity of document fields. The hash value / digital signature is validated by middleware. Users can request updates, deletions, and projections from Middleware.
SSL (data encryption in motion/transaction) will be used for every communication.

CA Server Component

It is critical to guarantee that the database's data is maintained safe. Authentication is a mechanism for proving a user's identity and gaining access to resources. Authentication relying on a password (id and password) is unsafe and susceptible. In a database, encryption is essential for keeping the data safe. Authentication is a technique for confirming a user's identity and gaining access to resources. The username/password authentication technique is the most frequent insecure approach. Kerberos Authentication is an extremely secure authentication technique that only allows the password to be used if it is encrypted. It is employed in a large client-server context.
Big Document DB Crypto uses a PKI authentication mechanism using digital certificates to prove and validate the user's identity. It consists of a certificate, an encryption key, and a certificate authority that ensures and signs the digital certificate. To validate user authentication and perform cloud transactions, PKI based system use the asymmetric cryptography where CA Server is used for generating the user digital certificate. The object of class shows the operation as well as field that involved in the functionality in the class diagram. The system constraint is defined using UML Object Constraint Language. The CA Server's characteristics and actions are depicted in Fig 2.

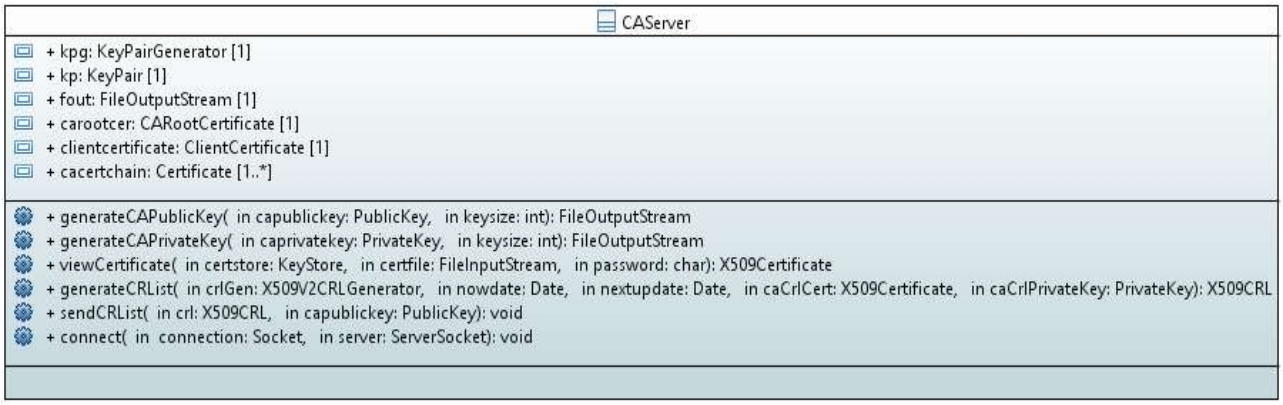

Figure 2. CA Server UML Class Diagram.

Middleware Component

Big Document DB Crypto proposes a trustworthy Middleware that cannot be compromised/hacked at any moment. After user identification has been validated, legitimate users utilize digital certificates, which are provided to middleware to authenticate their identities. The needed transformation was completed by middleware, and the database server did not require any changes.
UML Class Diagrams are used to illustrate a class's attributes, operations, and limitations in Fig 3. The class's action is specified in operation and is represented as a class method. Attributes and fields are two of a class's properties. The Document DB class displays the operations and attributes used in Document Database's CRUD activities.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

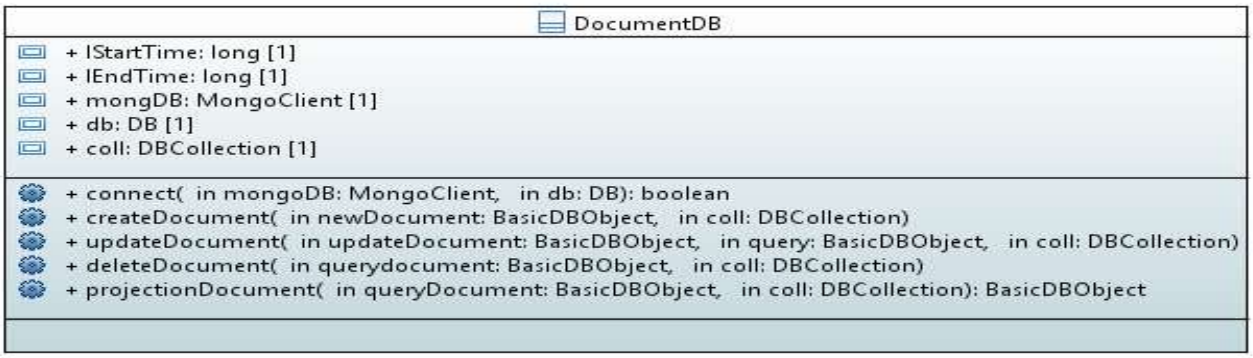

Figure 3. DocumentDB UML Class Diagram.

User Component

The useability of system using he Mobile Apps, Web Interface and Desktop application via user Interface (UI) in the BigDocumentDBCrypto. The system's users must register in order to be included in the security system. To encrypt communication between the Middleware and establish the user's validity, the CA Server registered the user and issued a Personal Digital Certificate (PKCS#12). It gives the information supplied by the user a higher level of trustworthiness. Before granting access, the middleware verifies the identity.

Middleware enables access control by requiring users to present a digital certificate as proof of identity. As a result, the targeted user will have access to the data and will be able to conduct CRUD activities. For the verifying the authentication of user operation the user certificate generation is illustrated in Fig 4.

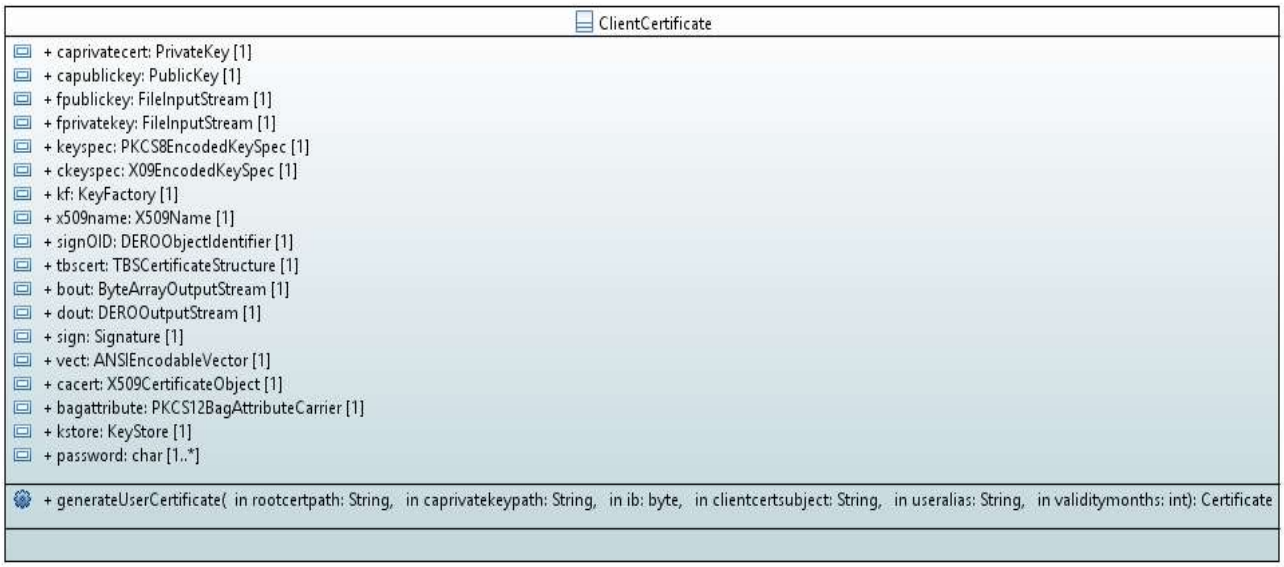

Figure 4. Client Certificate Generation.

Results and Discussion

This study investigated the symmetric algorithms with maximum key size: an empirical comparison and assessment of small, medium and large datasets of plaintext and multimedia contents i.e., image, audio, and video encryption and decryption operations have been performed at the application level. In Table 1, the system turnaround time recorded in millisecond and dispersion from mean is shown as 32.52.799, 28.71.252, 32.8 1.135 and 33.12.885 respectively. From the results, it can be deduced that AES takes less time to generate secret keys than other symmetric algorithms with inconsistent behavior.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Table 1. Symmetric algorithms secret key generation and plaintext comparative analysis.

	Blowfish (448)			AES (256)			DES (56)			3DES (168)		
	GEN	ENC	DEC	GEN	ENC	DEC	GEN	ENC	DEC	GEN	ENC	DEC
						Small Dataset						
Mean	32.5	12	2	28.7	10.5	2.9	32.8	12	1.8	33.1	12.3	2.2
STDEV	2.799	1	0.823	1.252	4.062	0.316	1.135	1.155	0.422	2.885	1.16	0.919
						Medium Dataset						
Mean	-	31	6	-	71	5	-	33	5	-	35	9
STDEV	-	3	2	-	3	1	-	3	1	-	3	2
						Large Dataset						
Mean	-	30	7	-	72	5	-	34	6	-	39	7
STDEV	-	2	1	-	4	2	-	3	1	-	2	2

Source: primary data

Fig 5 depicts the cooperative analysis of key generation with maximum size of various symmetric algorithms. AES was 13.240 percent quicker than Blowfish, 14.29 percent faster than DES and 15.33 percent faster than Triple DES, according to the data. As a result, AES is the most efficient method for generating secret keys.

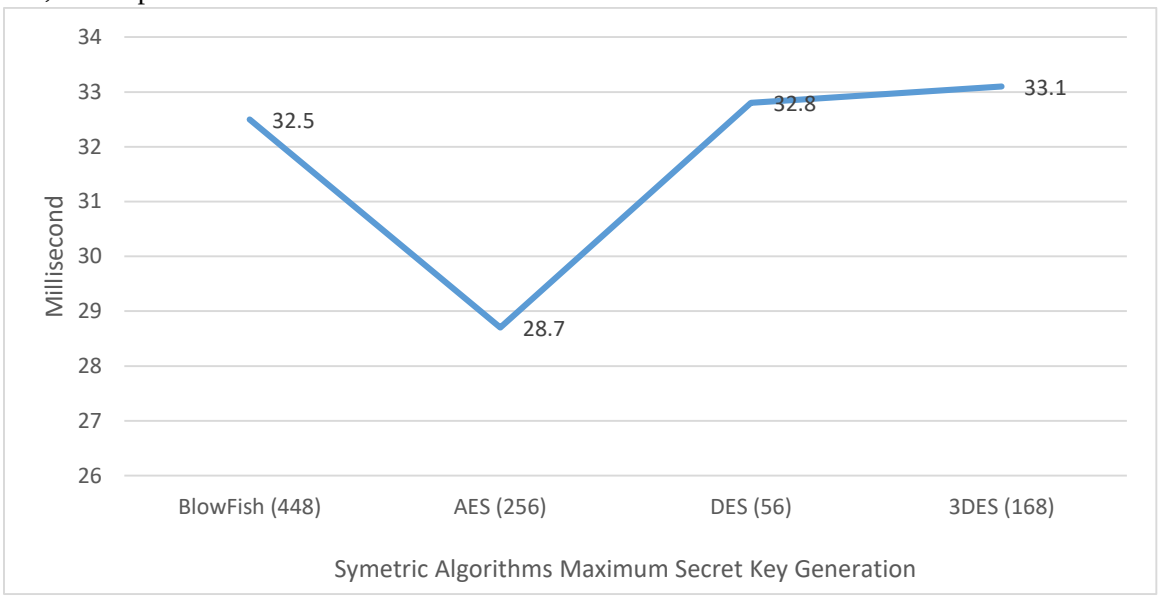

Figure 5. Symmetric algorithms secret key generation comparative analysis.

Fig 6 depicts the cooperative analysis of symmetric methods small dataset encryption and decryption with maximum key size. The turnaround time of various algorithms with maximum key size during of time in millisecond and standard deviation for the secret key was found i.e., 121, 10.54.062, 121.155 and 12.31.16 for encryption operation. The acquired results for the decryption procedure were 20.823, 2.90.316, 1.80.422, and 2.20.919. AES outperformed Blowfish and DES in terms of encryption by 14.285 percent. When compared to Triple DES, AES is 17.14 percent quicker. Decryption times for DES were determined to be 61.11 percent quicker than AES, 22.22 percent faster than Triple DES, and 11.11 percent faster than Blowfish. The result indicates that AES takes less time to encrypt than other symmetric algorithms with inconsistent behavior. DES, on the other hand, was quicker than other symmetric algorithms with a minor deviation from the mean. The turnaround time of various algorithms with maximum key size during of time in millisecond and dispersion of value from mean for the secret key were obtained for medium dataset encryption process. The trial results for decryption operation were 62%, 51%, 51%, and 92%.

Blowfish was shown to be 6.452 percent quicker than DES, 12.9 percent faster than Triple DES, and 129 percent faster than AES in terms of encryption.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Meanwhile, it was discovered that AES with a bigger key size and DES with a small key size were both 20% quicker than Triple DES and DES with a small key size in terms of time and scatter from mean. The results show that AES and DES take less time to encrypt data than other symmetric algorithms with inconsistent behavior.

The turnaround time of various algorithms with maximum key size during of time in millisecond and spreading of value from mean for the secret key was 302, 723, 343, and 392 for large dataset encryption operations. 71%, 52%, 61%, and 72% for decryption operations, respectively.

Blowfish was shown to be 13.33% quicker than DES, 30% faster than Triple DES, and 140 percent faster than AES in terms of encryption. Meanwhile, AES decryption was shown to be 20% quicker than DES, 40% faster than Triple DES, and 60% faster than Blowfish. The result indicates that AES takes less time to encrypt and decode than other symmetric algorithms with inconsistent behavior.

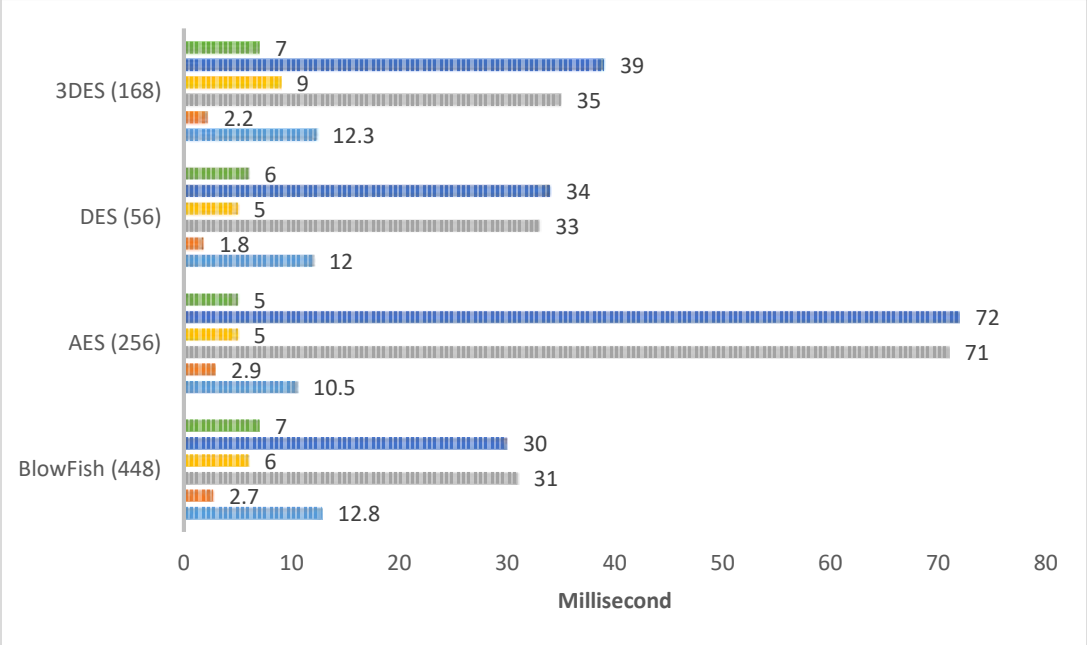

Figure 6. Symmetric algorithms plaintext comparative analysis.

In the application-level encryption and decryption, comparative analysis of encryption and decryption of small, medium and large dataset of image using the secret key as shown in Table 2.

Table 2. Symmetric algorithms image comparative analysis.

	Blowfish (448)		AES (256)		DES (56)		3DES (168)	
	ENC	DEC	ENC	DEC	ENC	DEC	ENC	DEC
				Small Dataset				
Mean	2.2	3.1	2.1	2.3	3.5	3.9	9.4	12.7
STDEV	0.422	0.516	0.316	0.675	0.527	0.316	0.516	1.829
				Medium Dataset				
Mean	10	13	8	8	22	24	64	69
STDEV	2	1	1	0	1	2	4	13
				Large Dataset				
Mean	24	30	8	8	54	55	154	162
STDEV	1	2	1	1	3	2	3	6

Source: primary data

Fig 7 shows image encryption of a small dataset using symmetric algorithm i.e., Blowfish (448 bits), AES (256 bits), DES (56 bits), and Triple DES (168 bits) and results obtained in milliseconds with dispersion from the mean value of 2.20 ± 0.422, 2.1 ± 0.316, 3.5 ± 0.527 and 9.4 ± 0.516. The decryption operation took 3.1 ± 0.516, 2.3 ± 0.675, 3.9 ± 0.316 and 12.7 ± 1.829.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

AES was shown to be 66.67 percent quicker than DES, 347.6 percent faster than Triple DES, and 4.76 percent faster than Blowfish in terms of encryption. Meanwhile, decryption times for AES were found to be 69.57 percent quicker than DES, 452.2 percent faster than Triple DES, and 34.78 percent faster than Blowfish. It may be deduced from the results that AES takes less time to encrypt and decode data than other symmetric algorithms with consistent behavior.

Medium dataset encryption with Blowfish (448 bits), AES (256 bits), DES (56 bits), and Triple DES (168 bits) mean value and dispersion from mean value is 10 ± 2, 8 ± 1, 2 ± 21, and 64 ± 4. The decryption results are 13 ± 1, 8 ± 0, 24 ± 2 and 69 ± 13 respectively.

AES was shown to be 175 percent quicker than DES, 700 percent faster than Triple DES, and 25% faster than Blowfish in terms of encryption. Meanwhile, decryption times for AES were found to be 200 percent quicker than DES, 762.5 percent faster than Triple DES, and 62.5 percent faster than Blowfish. It can be deduced from the results that AES takes less time to encrypt and decode data than other symmetric algorithms with modest standard deviation observed data behavior is consistent since data points are near together. In large dataset encryption results in mean value and spread from mean value is 24 ± 1, 8 ± 1, 54 ± 3, and 154 ± ss3 respectively. Meanwhile, the decryption process took 30 ± 2, 8 ± 1, 55 ± 2, and 162 ± 6 milliseconds respectively. AES was shown to be 575 percent quicker than DES, 1825 percent faster than Triple DES, and 200 percent faster than Blowfish in terms of encryption. Meanwhile, decryption times for AES were found to be 587.5 percent quicker than DES, 1925 percent faster than Triple DES, and 275 percent faster than Blowfish. It can be deduced from the results that AES takes less time to encrypt and decode data than other symmetric algorithms with modest standard deviation observed data behavior is consistent since data points are near together.

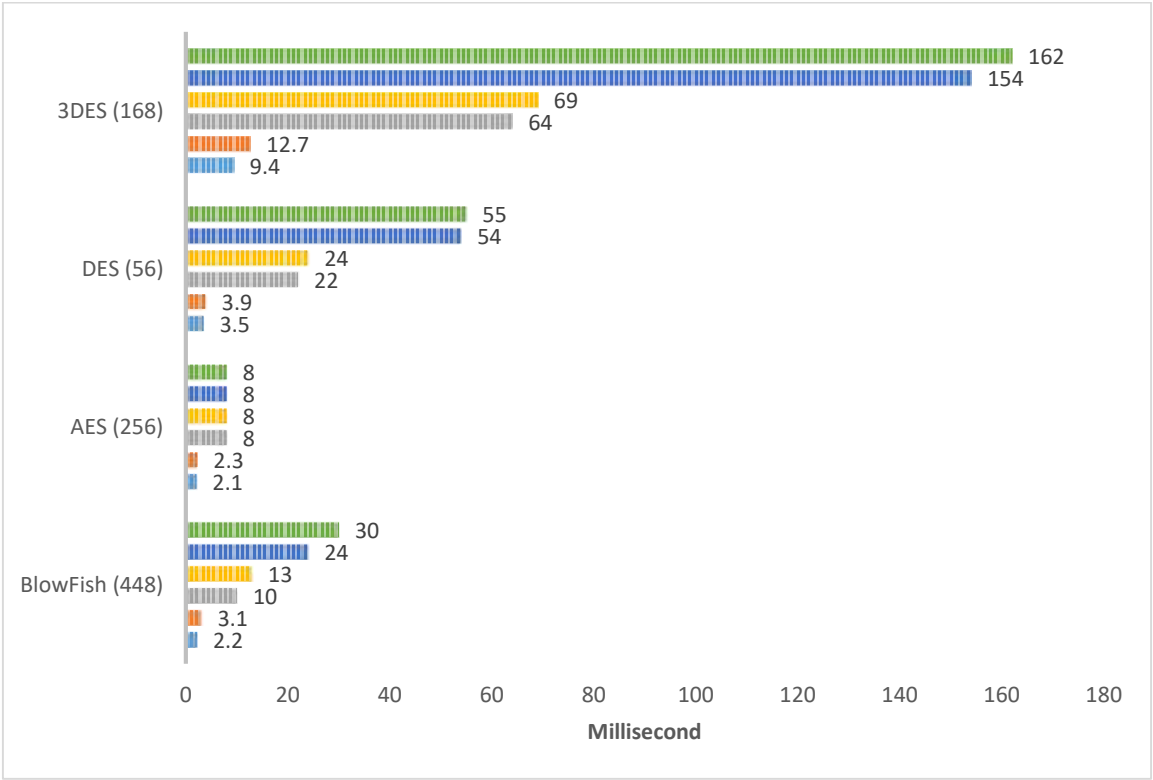

Figure 7. Symmetric algorithms image comparative analysis.

The audio and video files had a total size of 10680262 bytes when they were provided to the algorithms. The algorithms changed the audio and video into an unreadable format, which was then decrypted to return the audio and video to their original MP4 format. The symmetric algorithm's maximum key size. Table 3 indicates that, when compared to other symmetric algorithms, AES took

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

less time to encrypt and decode images, audio, and video. Experiments were carried out and results were collected for the encryption and decryption of audio and video on a small, medium, and big dataset in this work.

Table 3. Symmetric algorithms audio and video comparative analysis.

	Blowfish (448)		AES (256)		DES (56)		3DES (168)	
	ENC	DEC	ENC	DEC	ENC	DEC	ENC	DEC
				Small Dataset				
Mean	133.7	108	74.9	79.4	181	174	418	324
STDEV	8.286	11	7.4	2.716	5	6	26	2
				Medium Dataset				
Mean	206.2	209.5	84	76	418.7	438.9	1218	1200.5
STDEV	3.444	1.505	7	7	1.567	7.866	4.761	3.44
				Large Dataset				
Mean	699.3	702.1	181	173	1470	1504	4399	4351
STDEV	6.675	11.129	6	5	22	28	13	1263

Source: primary data

Fig 8, shows the turnaround time for encryption operation in milliseconds and spread of value from mean using secret key are 133±.8.286, 74.9±7.4, 181+5, and 418+26 respectively. Meanwhile, the decryption process took 108±11, 79.4±2.716, 174±6, and 324±2 respectively.

In terms of encryption, AES outperformed DES by 141.7 percent, Triple DES by 458.1 percent, and Blowfish by 78.50 percent. Meanwhile, decryption times for AES were found to be 119.1% quicker than DES, 308.1% faster than Triple DES, and 36.02 percent faster than Blowfish. According to the findings, AES takes less time to encrypt and decode a small dataset of audio and video than other symmetric algorithms with inconsistent behaviour.

Encryption of medium dataset of multimedia contents i.e., audio and video, the duration of time is measured in milliseconds, and the scatter from the mean value is 206.2+3.444, 84+7, 418.7+1.567, and 1218 + 4.761. For decryption operation obtained results is 209.5+1.505, 76+7, 438.9+7.866, and 1200.5+3.44 respectively.

AES was shown to be 398.5 percent quicker than DES, 1350 percent faster than Triple DES, and 145.47 percent faster than Blowfish in terms of encryption. Meanwhile, decryption times for AES were found to be 477.5 percent slower than DES, 1480 percent quicker than Triple DES, and 175.65 percent faster than Blowfish. It can be concluded from the results that AES takes less time to encrypt and decrypt data than other symmetric algorithms with modest standard deviation recorded data behaviour is consistent, and data point tendencies are quite near.

Encryption operation with Blowfish (448 bits), AES (256 bits), DES (56 bits), and Triple DES (168 bits) multimedia contents i.e., audio and video large dataset, the time is measured in milliseconds and the scatter from the mean value is 699.3+6.675, 181+6, 1470+22, and 4399+13. For decryption operation is 702.1+11.129, 173+5, 1504+28 and 4351+1263 respectively.

AES was shown to be 712.2 percent quicker than DES, 2330 percent faster than Triple DES, and 286.35 percent faster than Blowfish in terms of encryption. Meanwhile, decryption times for AES were found to be 769.4 percent quicker than DES, 2415 percent faster than Triple DES, and 305.83 percent faster than Blowfish. Because data points tend to be relatively near, it may be concluded that AES takes less time to encrypt and decrypt than other symmetric algorithms with small standard deviation observed behaviour of data is consistent.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

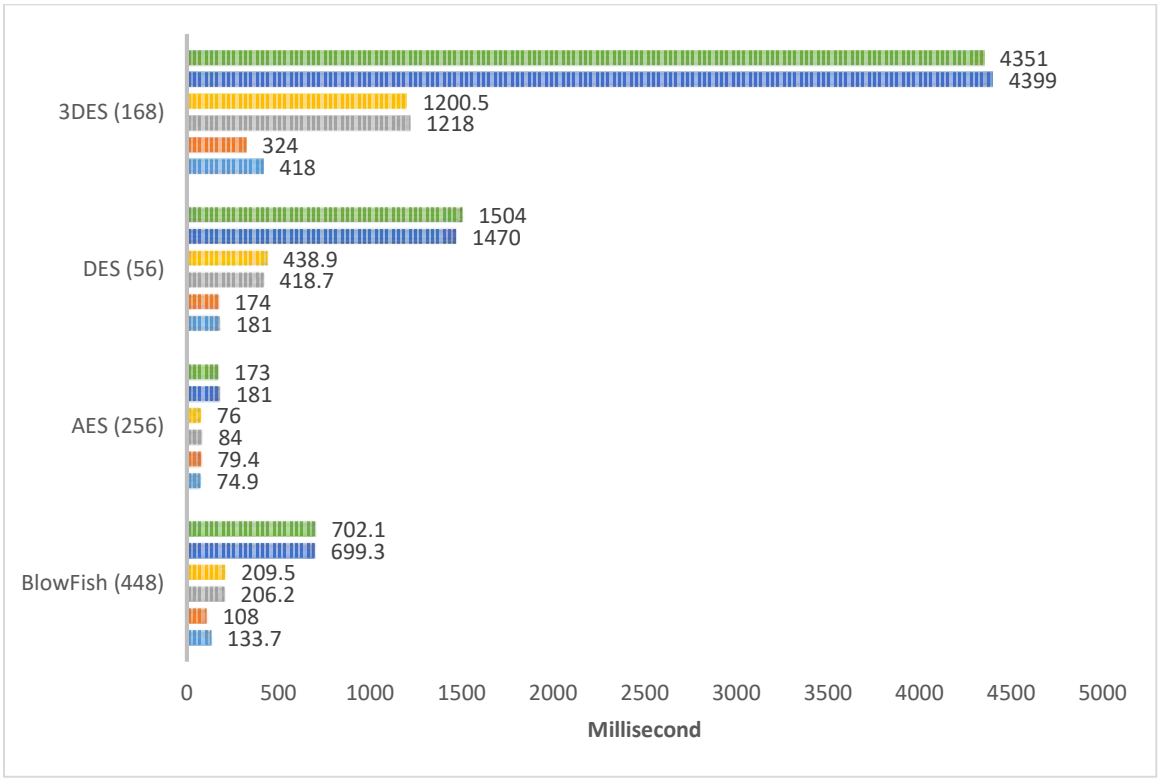

Figure 8. Symmetric algorithms audio and video comparative analysis.

Conclusion

The architectural design is a creative process in which a system is created and built to fulfil all of the system's functional and non-functional requirements. An architectural model is the product of the architectural design process and explains how the system is constructed as a series of communicating components. Architecture design is concerned in developing the overall structure of the system. The component based secure system working over cloud is proposed in this research work. The proposed system rapidly may be extended on demands. Due to document database security vulnerabilities, proper data protection and integrity should be done at the application level using BigDocumentDBCrypto to mitigate any security issues that arise while achieving the intended goals, according to the prevalence of security issues. The middleware's major benefit is that it does not require any changes to the document database management system and can easily be expanded to non-relational databases such as column-oriented, key-value, and graphics-oriented databases.

For small, medium, and large data sets, recorded findings demonstrated statistically significant differences in plain text and multimedia content (e.g., audio and video) with time measured in milliseconds. It was revealed from results that in the process of transforming the image into ciphertext symmetric algorithms took less time for encryption and decryption as compared to plaintext and audio and video.

Authors' Declaration

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are ours. Furthermore, any Figures and images, that are not ours, have been included with the necessary permission for re-publication, which is attached to the manuscript.
- Ethical Clearance: The project was approved by the local ethical committee in University of Sindh, Jamshoro, Pakistan.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Authors' Contribution Statement

This work was carried out in collaboration between all authors. M. J. conducted research experiments, implementation, analysis, interpretation and wrote the manuscript. N.I. contributed to the data curation and acquisition of data. A. G. Supervision. M. M. Co-supervision. A. J. Contributed to the conceptualization and methodology. All authors read and approved the final manuscript.

References

1. Shubh S, Vyom S. Component Based Software Engineering, Int J Res Appl Sci Eng Technol. 2021; 9(VIII): 1588-1595. https://doi.org/10.22214/ijraset.2021.37632.
2. Yunusa A, Mustapha H. Security and Privacy in Cloud Computing: Technical Review. Future Internet. 2022; 14(11): 2-27. https://doi.org/10.3390/fi14010011.
3. Dan G, Jeremy K, Evan S, Zev W, Dulani W. Cloud-Trust—a Security Assessment Model for Infrastructure as a Service (IaaS) Clouds. IEEE Trans. Cloud Comput. 2017; 5(3): 523-536. https://doi.org/10.1109/TCC.2015.2415794 .
4. Zhifeng X, Yang X. Security and Privacy in Cloud Computing. IEEE Commun. Surv. Tutor. 2013; 15(2): 843-859. https://doi.org/10.1109/SURV.2012.060912.00182.
5. Demchenko Y, Ngo C, de L, Lee C. Federated Access Control in Heterogeneous Intercloud Environment: Basic Models and Architecture Patterns. IEEE Int Conf Cloud Eng., Boston, USA. 2014: 439-445. https://doi.org/10.1109/IC2E.2014.84.
6. Kent S. Model Driven Engineering. In: Butler, M., Petre, L., Sere, K. (Eds) Integrated Formal Methods. IFM 2002. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg**.** 2002**;** 2335**:** 286-298. https://doi.org/10.1007/3-540-47884-1_16
7. Mantas J, Lina C, Rita B. Solidity code generation from UML state Machines in model-driven smart Contract Development. IEEE Access. 2022; 10(2022): 33465-33481. https://doi.org/ 10.1109/ACCESS.2022.3162227
8. Berdik D, Otoum S, Schmidt N, Porter D, Jararweh Y. A survey on blockchain for information systems management and security. Inf Process Manage. 2021; 58(1): 1-15. https://doi.org/ 10.1016/j.ipm.2020.102397.
9. Alvarez ML, Sarachaga I, Burgos A, Estevez E, Marcos M. A Methodological Approach to Model-Driven Design and Development of Automation Systems. IEEE Trans. Auto Sci Eng. 2018; 15(1): 67-79. https://doi.org/10.1109/TASE.2016.2574644.
10. Sitalakshmi V, Ramanathan V. Big data security challenges and strategies. AIMS Math. 2019; 4(3): 860–879. https://doi.org/10.3934/math.2019.3.860
11. Parra P, Polo OR., Fernandez J, Da Silva A, Sanchez S. Martínez A. A Platform-Aware Model-Driven Embedded Software Engineering Process Based on Annotated Analysis Models. IEEE Trans Emerg Top Comput. 2021; 9(1): 78-89. https://doi.org/10.1109/TETC.2018.2866024.
12. Samuel P. Automatic Code Generation from UML State Chart Diagrams. IEEE Access. 2019; 7: 8591-8608. https://doi.org/ 10.1109/ACCESS.2018.2890791.
13. Hou B, Shi Y, Qian K, Tao L. Towards Analyzing MongoDB NoSQL Security and Designing Injection Defense Solution. IEEE 3rd Int. Conf. Big Data Secur. Cloud (bigdatasecurity), IEEE Int Conf Hi. Per Sm Comput. (HPSC) and IEEE Int Conf Intgen Data Secur. (IDS), Beijing, China. 2017: 90-95. https://doi.org/10.1109/BigDataSecurity.2017.29.
14. Ron Aviv, Alexandra SP, Emanuel B. NoSQL No Injection? Examining NoSQL Security. In Proc of the 9th Workshop on Web 2.0 Secur. Priv. (W2SP). 2015; 1-4. https://doi.org/ 10.48550/arXiv.1506.04082.
15. Wu G, Mu Y, Susilo W. Threshold privacy-preserving cloud auditing with multiple uploaders. Int J Inf Secur. 2019; 18: 321–331. https://doi.org/10.1007/s10207-018-0420-6.
16. Samaraweera GD, Chang JM. Security and Privacy Implications on Database Systems in Big Data Era: A Survey. IEEE Trans Knowl Data Eng. 2021; 33: 239-258. https://doi.org/ 10.1109/TKDE.2019.2929794.
17. Jitender K, Varsha G. Security analysis of unstructured data in NOSQL Mongodb Database. IEEE Int Conf Comput Technol Smt Nat. (IC3TSN). 2018: 300-305. https://doi.org/ 10.1109/IC3TSN.2017.8284495.
18. Kuntal P. Performance analysis of AES, DES and Blowfish cryptographic algorithms on small and large data files. Springer. Int J Inf Tecnol. 2019; 11, 813–819. https://doi.org/10.1007/s41870-018-0271-4.
19. Abikoye OC, Adewole KS, Oladipupo AJ. Efficient data hiding system using cryptography and steganography. Int J Appl Inf Syst. 2012; 4(11):6–12. https://doi.org/10.5120/ijais12-450763.
20. Ikenna RC, Norliza K. A Scoping Study on Lightweight Cryptography Reviews in IoT. Baghdad Sci J. 2021; 18(2): 989-1000. http://doi.org/10.21123/bsj.2021.18.2(Suppl.).0989 .
21. Shukur WA, Qurban LK, Aljuboori A. Digital Data Encryption Using a Proposed W-Method Based on AES and DES Algorithms. Baghdad Sci J. 2023; 20(4): 1414. https://doi.org/ 10.21123/bsj.2023.7315.
22. Mujeeb-ur-Rehman J, Abdul GM, Nadeem AK and Mujeeb-u-Rehman M. Data integrity issues and challenges in next generation non-relational document-oriented database outsourced in public cloud, Int J Emerg Tren Eng Res. 2021; 9(4): 416-420.

Published Online First: February, 2024
https://doi.org/10.21123/bsj.2024.8722
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

https://doi.org/10.30534/ijeter/2021/13942021.ISSN 2347-3983.
23. Jamali MR, Memon AG, Maree MR. Security issues in data at rest in a non-relational Document Database. Sindh Univ Res J. (Sci Ser.). 2021; 52 (03): 279-284. http://doi.org/10.26692/sujo/2020.09.41.

تصميم معمارية لامنية البيانات في الحوسبة السحابية المعتمدة عل انظمة البيانات الكبيرة

مجيب الرحمن جمالي[1]، نجمة امتياز علي[2]، عبد الغفور ميمون[1]، مجيب الرحمن مرعي[2]، عادل جمالي[2]

[1] معهد الإيمان للإدارة والعلوم، كراتشي السند، باكستان
[2]معهد الرياضيات وعلوم الكمبيوتر، جامعة السند، جامشورو، باكستان.

الخلاصة

التصميم المعماري للنظام هو عملية إبداعية بعد كل شيء. المتطلبات غير الوظيفية وبنية البرامج لها علاقة قوية. يعد أمن النظام أحد الشواغل الكبيرة وهو حاجة ومتطلب غير وظيفي. يجب أن يعتمد الأسلوب والهيكل المعماري المختاران للنظام على متطلبات النظام غير الوظيفية. تُستخدم تقنيات البيانات الضخمة لتخزين كميات هائلة من البيانات التشغيلية في المجال العام. تم اقتراح مخططات هيكل وسلوك النظام التي تستخدم تصميمًا معماريًا آمنًا لنظام يعمل عبر السحابة. تم التحقيق في مكونات مختلفة من البيانات الضخمة في هذا العمل البحثي نظرًا لخطورة القضايا الأمنية التي تركز على الخصوصية والسرية. يعد تأمين البيانات عبر السحابة التي يتم الاستعانة بمصادر خارجية لها مصدر قلق متزايد. هناك مسؤولو قواعد بيانات ضارة من الخارج ومن الداخل يمكنهم الوصول إلى البيانات الخاصة والحساسة وتعديلها ، وبالتالي يمكن أن يكون مزود خدمة قاعدة البيانات مخادعًا. آليات الأمان مطلوبة بالضرورة لحماية البيانات الحساسة الموجودة على السحابات العامة. على مستوى التطبيق ، يضمن الحل المقترح خصوصية وسرية البيانات الموجودة على السحابة. تم إجراء اختبارات مكثفة ، وكانت نتائج هذا البحث كما يلي: كشفت النتائج المسجلة عن اختلافات ذات دلالة إحصائية في النص العادي ومحتويات الوسائط المتعددة (أي الصوت والفيديو) مع قياس الوقت بالمللي ثانية لمجموعات البيانات الصغيرة والمتوسطة والكبيرة.

الكلمات المفتاحية: البيانات الضخمة، الحوسبة السحابية، التشفير، الأمن، هندسة البرمجيات..