IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

OPTIMIZATION IN EDGE COMPUTING FOR CENTRALIZED DATA CENTER USING BIG DATA TECHNIQUE

Dr.K.Venkatasalam[1], K.Mohanapriya[2], P.Mathumitha[3], S.Lavanyasri[4]

1.ASSOCIATE PROFESSOR, 2,3,4. UG STUDENTS
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MAHENDRA ENGINEERING COLLEGE, TAMILNADU, INDIA

ABSTRACT

Advances made in the wearable and biosensor has attracted the applicability and usability of the healthcare applications. The biosensors collect the human physiological data from a remote area using wireless communication medium. Different form of security implications have to be exhibited in Edge Computing Networks. This paper researches on enhancing the network layer and application layer of the Edge Computing Networks in healthcare environment. The objective of the each phase is to enhance the data accuracy of the collected data with minimized delay. In the first phase, we have constructed attack model for Sybil, Sinkhole and Wormhole attacks and detected those attacks under different constraints. Secondly, an efficient packet transmission model which helps to transmit the sensed data based on their emergency using Weighted Product Model (WPM). After devising network layer, a lightweight security scheme, an improved Elliptic Curve Cryptography (ECC) that permits the data access for the authorized users. Experimental analysis is carried out for each phase under pre-defined simulation parameters which explores better results. The framed attack model detects the three eminent attacks in terms of False Positive Rate (FPR) and False Negative Rate (FNR) which helps the Data Controller for message transmission process. Likewise, we have also achieved 97% Packet Delivery Ratio with 0.8ms Packet Dropping Ratio for better packet transmission model than the existing TOPSIS model.

Keywords: Biosensor, Remote system & Network

INTRODUCTION

The idea of Internet of Things (IoT) has become increasingly popular, which enables various objects including physical devices, vehicles, buildings and other items embedded with computing and communication capabilities to exchange data. However, because of limitations in the computation capability, battery, storage and bandwidth, smart devices sometimes may decrease the quality of services and weaken the user experience. Cloud computing supplies resources to end users in terms of software, infrastructure and platform, and delivers services to applications at a comparatively small cost, which has been considered as a promising solution to mitigate the limitation of devices with constrained resources. Unfortunately, cloud computing cannot be an answer to all emerging problems, since some IoT applications need to be instantly responded, some contain sensitive information, and some generate a large amount of data and cause a heavy workload to the network.

Attribute-based encryption (ABE) protects data security and privacy by sharing data among a group of privileged data users, which is believe to be a very desirable candidate for accomplishing scalable (i.e., fine-grained) access control over data items in encrypted forms. One feature of current ABE schemes is that they are built upon bilinear pairings (or bilinear maps), and thus it is significantly challenging to deploy such schemes in applications where the private data will be accessed via a mobile device with a constrained computation capacity. With this issue in mind, Green, Hohenberger and Waters suggested to divide the private attribute-key in an ABE scheme into a transformation key and a decryption key, of which the former is sent to a proxy such that the proxy can make a transformation on the ciphertext (to produce a partially decrypted ciphertext) and the latter is given to the data user such that the data user can completely decrypt the transformed ciphertext. Following this direction of delegating the workloads in the decryption to a third party like a proxy, in terms of enhancing data security and privacy to meet different requirements in the real world, several ABE schemes enabling outsourced decryption have been proposed.

ABE with outsourced decryption (ABE-OD) has an inherent property to be implemented in an edge computing network to enforce access control and protect data security and privacy, where the edge device can play the role of the proxy. However, all existing ABE-OD schemes require secure channels to distribute private keys to data users, which is not feasible for all applications in the edge computing network due to the expensive cost in building secure channels. Motivated by this observation, we consider designing a secure channel free ABEOD scheme (to distinguish from ABE-OD, we call it proxy aided ciphertext-policy ABE (PA-CPABE)) to provide scalable access control over data items in encrypted forms in the edge computing network.

SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

SOFTWARE ENVIRONMENT

FEATURES OF. NET

Microsoft .NET is a set of Microsoft software technologies for rapidly building and integrating XML Web services, Microsoft Windows-based applications, and Web solutions. The .NET Framework is a language-neutral platform for writing programs that can easily and securely interoperate. There's no language barrier with .NET: there are numerous languages available to the developer including Managed C++, C#, Visual Basic and Java Script. The .NET framework provides the foundation for components to interact seamlessly, whether locally or remotely on different platforms. It standardizes common data types and communications protocols so that components created in different languages can easily interoperate.

".NET" is also the collective name given to various software components built upon the .NET platform. These will be both products (Visual Studio.NET and Windows.NET Server, for instance) and services (like Passport, .NET My Services, and so on)..

Managed Data

With Managed Code comes Managed Data. CLR provides memory allocation and Deal location facilities, and garbage collection. Some .NET languages use Managed Data by default, such as C#, Visual Basic.NET and JScript.NET, whereas others, namely C++, do not. Targeting CLR can, depending on the language you're using, impose certain constraints on the features available. As with managed and unmanaged code, one can have both managed and unmanaged data in .NET applications - data that doesn't get garbage collected but instead is looked after by unmanaged code.

Common Type System

The CLR uses something called the Common Type System (CTS) to strictly enforce type-safety. This ensures that all classes are compatible with each other, by describing types in a common way. CTS define how types work within the runtime, which enables types in one language to interoperate with types in another language, including cross-language exception handling. As well as ensuring that types are only used in appropriate ways, the runtime also ensures that code doesn't attempt to access memory that hasn't been allocated to it.

Common Language Specification

The CLR provides built-in support for language interoperability. To ensure that you can develop managed code that can be fully used by developers using any programming language, a set of language features and rules for using them called the Common Language Specification (CLS) has been defined. Components that follow these rules and expose only CLS features are considered CLS-compliant.

THE CLASS LIBRARY

.NET provides a single-rooted hierarchy of classes, containing over 7000 types. The root of the namespace is called System; this contains basic types like Byte, Double, Boolean, and String, as well as Object. All objects derive from System. Object. As well as objects, there are value types. Value types can be allocated on the stack, which can provide useful flexibility. There are also efficient means of converting value types to object types if and when

C#.NET is also compliant with CLS (Common Language Specification) and supports structured exception handling. CLS is set of rules and constructs that are supported by the CLR (Common Language Runtime). CLR is the runtime environment provided by the .NET Framework; it manages the execution of the code and also makes the development process easier by providing services.

C#.NET is a CLS-compliant language. Any objects, classes, or components that created in C#.NET can be used in any other CLS-compliant language. In addition, we can use objects, classes, and components created in other CLS-compliant languages in C#.NET .The use of CLS ensures complete interoperability among applications, regardless of the languages used to create the application.

CONSTRUCTORS AND DESTRUCTORS:

Constructors are used to initialize objects, whereas destructors are used to destroy them. In other words, destructors are used to release the resources allocated to the object. In C#.NET the sub finalize procedure is available. The sub finalize procedure is used to complete the tasks that must be performed when an object is destroyed. The sub finalize procedure is called automatically when an object is destroyed. In addition, the sub finalize procedure can be called only from the class it belongs to or from derived classes.

GARBAGE COLLECTION

Garbage Collection is another new feature in C#.NET. The .NET Framework monitors allocated resources, such as objects and variables. In addition, the .NET Framework automatically releases memory for reuse by destroying objects that are no longer in use.

In C#.NET, the garbage collector checks for the objects that are not currently in use by applications. When the garbage collector comes across an object that is marked for garbage collection, it releases the memory occupied by the object.

OVERLOADING

Overloading is another feature in C#. Overloading enables us to define multiple procedures with the same name, where each procedure has a different set of arguments. Besides using overloading for procedures, we can use it for constructors and properties in a class.

MULTITHREADING:

C#.NET also supports multithreading. An application that supports multithreading can handle multiple tasks simultaneously, we can use multithreading to decrease the time taken by an application to respond to user interaction.

STRUCTURED EXCEPTION HANDLING

C#.NET supports structured handling, which enables us to detect and remove errors at runtime. In C#.NET, we need to use Try…Catch…Finally statements to create exception handlers. Using Try…Catch…Finally statements, we can create robust and effective exception handlers to improve the performance of our application.

THE .NET FRAMEWORK

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet.

OBJECTIVES OF. NET FRAMEWORK

1. To provide a consistent object-oriented programming environment whether object codes is stored and executed locally on Internet-distributed, or executed remotely.
2. To provide a code-execution environment to minimizes software deployment and guarantees safe execution of code.
3. Eliminates the performance problems.

There are different types of application, such as Windows-based applications and Web-based applications.

FEATURES OF SQL-SERVER

The OLAP Services feature available in SQL Server version 7.0 is now called SQL Server 2000 Analysis Services. The term OLAP Services has been replaced with the term Analysis Services. Analysis Services also includes a new data mining component. The Repository component available in SQL Server version 7.0 is now called Microsoft SQL Server 2000 Meta Data Services. References to the component now use the term Meta Data Services. The term repository is used only in reference to the repository engine within Meta Data Services

SQL-SERVER database consist of six type of objects, They are,

1. TABLE
2. QUERY
3. FORM
4. REPORT
5. MACRO

TABLE:

A database is a collection of data about a specific topic.

VIEWS OF TABLE:

We can work with a table in two types,

1. Design View

2. Datasheet View

Design View

To build or modify the structure of a table we work in the table design view. We can specify what kind of data will be hold.

Datasheet View

To add, edit or analyses the data itself we work in tables datasheet view mode.

QUERY:

A query is a question that has to be asked the data. Access gathers data that answers the question from one or more table. The data that make up the answer is either dynaset (if you edit it) or a snapshot (it cannot be edited).Each time we run query, we get latest information in the dynaset. Access either displays the dynaset or snapshot for us to view or perform an action on it, such as deleting or updating.

SYSTEM ANALYSIS

EXISTING SYSTEM

Cloud computing supplies resources to end users in terms of software, infrastructure and platform, and delivers services to applications at a comparatively small cost, which has been considered as a promising solution to mitigate the limitation of devices with constrained resources. Unfortunately, cloud computing cannot be an answer to all emerging problems, since some IoT applications need to be instantly responded, some contain sensitive information, and some generate a large amount of data and cause a heavy workload to the network. The demand for distributing the IoT workloads between the local data centre and the cloud has resulted in an architectural model called Edge Computing (which is also known as Fog Computing Edge computing extends cloud computing and facilitates cloud computing in significantly reducing the delays incurred by service deployments. End devices, edge and cloud form a three-layer hierarchical architecture (as shown in Fig. 1) for the service delivery, which supports a wide range of applications (e.g., the smart city network). Take the autonomous vehicle network as an instance, where the vehicle might produce gigabyte data in one second, and the real-time processing is in necessity as any delay in practice could lead the vehicle to make false resolutions

PROPOSED SYSTEM

To address this problem, Green, Hohenberger and Waters recommended to outsource the decryption workload in ABE to a proxy (or a server) where the private attribute-key is divided into a transformation key for the proxy and a decryption key for the data user such that only one exponentiation operation is needed to be conducted by the data user to decrypt the result received from the proxy to obtain the original message. The proxy is not a trusted entity, so it may not do the calculation in a correct way. To address this issue in ABE with outsourced decryption (ABE-OD)

SOFTWARE REQUIREMENTS:

- Operating system : Windows XP.
- Coding Language : ASP.NET, C#.NET
- Data Base : MS SQL SERVER 2005

SYSTEM DESIGN

DATA FLOW DIAGRAM / USE CASE DIAGRAM / FLOW DIAGRAM

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

8.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CONCLUSION

Nowadays, cloud storage, which can offer on-demand outsourcing data services for both organizations and individuals, has been attracting more and more attention. However, one of the most serious obstacles to its development is that users may not fully trust the CSPs in that it is difficult to determine whether the CSPs meet their legal expectations for data security. Therefore, it is critical and significant to develop efficient auditing techniques to strengthen data owners' trust and confidence in cloud storage. In this paper, we are motivated to present a novel public auditing scheme for secure cloud storage using dynamic hash table (DHT), which is a new two dimensional data structure used to record the data property information for dynamic auditing. Differing from the existing works, our scheme migrates the auditing metadata excerpt the block tags from the CSP to the TPA, and thereby significantly reduces the computational cost and communication overhead. Meanwhile, exploiting the structural advantages of the DHT, our scheme can also achieve better performance than the state-of-the-art schemes in the updating phase. In addition, for privacy preservation, our scheme introduces a random masking provided by the TPA into the process of generating proof to blind the data information.

Moreover, we would like to point out that no single method can achieve perfect audits for all types of cloud data, just as no standard has a universal validity. Thus, it may be a new trend to design a more effective scheme, including different audit strategies for various types of cloud data, which is also the direction for our future work.

FUTURE WORK

- In future enhanced to selects the most effective routing path and the method increases the transmitted power in order to get high communication rate.
- Meanwhile, a coarse-grained transmission path algorithm is provided for low deadline situation.

REFERENCE

[1] H. Dewan and R. C. Hansdah. ″A Survey of Cloud Storage Facilities ″, Proc. 7th IEEE World Congress on Services, pp. 224-231, July 2011.

[2] C. Wang, Q. Wang, K. Ren, N. Cao and W. Lou. ″Toward Secure and Dependable Storage Services in Cloud Computing″, IEEE Trans. Service Computing, vol. 5, no. 2, pp. 220-232, 2012.

[3] K. Ren, C. Wang and Q. Wang. “Security Challenges for the Public Cloud,” IEEE Internet Computing, vol. 16, no. 1, pp. 69– 73, 2012.

[4] J. Ryoo, S. Rizvi, W. Aiken and J. Kissell. ″Cloud Security Auditing: Challenges and Emerging Approaches″, IEEE Security & Privacy, vol. 12, no. 6, pp. 68-74, 2014.

[5] C. Wang, K. Ren, W. Lou and J. Li. ″Toward Publicly Auditable Secure Cloud Data Storage Services″, IEEE network, vol. 24, no. 4, pp. 19-24, 2010.

[6] Q. Wang, C. Wang, K. Ren, W. Lou and J. Li. ‘‘Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing,’’ IEEE Trans. on Parallel and Distributed Systems, vol. 22, no. 5, pp. 847-859, 2011.

[7] F. Sebé, J. Domingo-Ferrer, A. Martínez-Ballesté, Y. Deswarte and J.-J. Quisquater, “Efficient Remote Data Possession Checking in Critical Information Infrastructures,” IEEE Trans. Knowledge Data Eng., vol. 20, no. 8, pp. 1034-1038, 2008.

[8] A. Juels and B.S. Kaliski Jr., “PoRs: Proofs of Retrievability for Large Files,” Proc. ACM Conf. Computer and Communications Security (CCS ’07), pp. 584-597, 2007.

[9] G. Ateniese, R.B. Johns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson and D. Song, ‘‘Provable Data Possession at Untrusted Stores,’’ Proc. 14th ACM Conf. on Comput. and Commun. Security (CCS), pp. 598-609, 2007.

[10] K. Yang and X. Jia. ″Data Storage Auditing Service in Cloud Computing: Challenges, Methods and Opportunities″. World Wide Web, vol. 15, no. 4, pp. 409-428, 2012

[11] C. Wang, Q. Wang, K. Ren and W. Lou, ‘‘Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing,’’ Proc. IEEE INFOCOM, pp. 1-9, 2010.

[12] C. Wang, S. M. Chow, Q. Wang, K. Ren and W. Lou, ″Privacy-Preserving Public Auditing for Secure Cloud Storage,″ IEEE Trans. on Computers, vol. 62, no. 2, pp. 362-375, 2013.

[13] Y. Zhu, H. Hu, G. Ahn, and M. Yu, “Cooperative Provable Data Possession for Integrity Verification in Multi-Cloud Storage,” IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231-2244, 2012.

[14] K. Yang and X. Jia, “An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing,” IEEE Trans. on Parallel and Distributed Systems, vol. 24, no. 9, pp.1717-1726, 2013.

[15] C. C. Erway, A. Küpçü, C. Papamanthou and R. Tamassia.“Dynamic Provable Data Possession,” Proc. 16th ACM Conf. Computer and Comm. Security, pp. 213-222, 2009.

[16] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu and S. S.Yau, “Dynamic Audit Services for Outsourced Storage in Clouds,” IEEE Trans. on Services Computing, vol. 6, no. 2, pp. 227–238, 2013.

[17] D. Boneh, B. Lynn and H. Shacham, ‘‘Short Signatures from the Weil Pairing,’’ Proc. ASIACRYPT, vol. 2248, LNCS, pp. 514-532, 2001.

[18] B. Wang, B. Li and H. Li. ″Panda: Public Auditing for Shared Data with Efficient User Revocation in the Cloud″, IEEE Trans. on Service Computing, vol. 8, no. 1, pp. 92-106, 2015.

[19] C. Liu, R. Ranjian, X. Zhang, C. Yang, D. Georgakopoulos and J. Chen. ″Public Auditing for Big Data Storage in Cloud Computing-- A Survey″, Proc. 16th IEEE International Conf. Computational Science and Engineering (CSE), pp. 1128-1135, 2013.

[20] C. Liu, J. Chen, L. T. Yang, X. Zhang, C. Yang, R. Ranjan and K. Ramamohanarao,″ Authorized Public Auditing of Dynamic Big Data Storage on Cloud with Efficient Verifiable Fine-grained Updates,″ IEEE Trans. on Parallel and Distributed Systems, vol. 25, no. 9, pp. 2234-2244, 2014.

[21] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, “Aggregate and Verifiably Encrypted Signatures from Bilinear Maps,” Proc. 22nd Int’l Conf. Theory and Applications of Cryptographic Techniques (Eurocrypt ’03), pp. 416-432, 2003.

[22] H. Shacham and B. Waters, “Compact Proofs of Retrievability,” Proc. 14th Int’l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT ’08), pp.