Design & Development of Provable Data Possession and Proofs of Retrievability Techniques for Clients' Data Integrity

Deepesh Jarori[1]; Jhade Srinivas[2]& Prof.Dr.G.Manoj Someswar[3]

1. **M.Tech.(CSE) from Narasimha Reddy Engineering College, Affiliated to JNTUH, Hyderabad, Telangana, India**
2. **M.Tech., Ph.D., Associate Professor, Department of CSE, Narasimha Reddy Engineering College, Affiliated to JNTUH, Hyderabad, Telangana, India**
3. **B.Tech., M.S.(USA), M.C.A., Ph.D., Principal & Professor, Department Of CSE, Anwar-ul-uloom College of Engineering & Technology, Affiliated to JNTUH, Vikarabad, Telangana, India**

ABSTRACT

Provable data possession (PDP) is a probabilistic proof technique for cloud service providers (CSPs) to prove the clients' data integrity without downloading the whole data. In 2012, Zhu et al. proposed the construction of an efficient PDP scheme for multi cloud storage. They studied the existence of multiple CSPs to cooperatively store and maintain the clients' data. Then, based on homomorphic verifiable response and hash index hierarchy, they presented a cooperative PDP (CPDP) scheme from the bilinear pairings. They claimed that their scheme satisfied the security property of knowledge soundness. It is regretful that this comment shows that any malicious CSP or the malicious organizer (O) can generate the valid response which can pass the verification even if they have deleted all the stored data, i.e., Zhu et al.'s CPDP scheme cannot satisfy the property of knowledge soundness. Then, we discuss the origin and severity of the security flaws.

KEYWORDS: Provable Data Possession(PDP); Cloud Service Provider (CSP); Cooperative PDP(CPDP); Hash Index Hierarchy; Provable Data Possession; Proofs of Retrievability; Trusted Third Party (TTP); Third Party Auditor(TPA)

INTRODUCTION

In recent years, cloud computing has rapidly expanded as an alternative to conventional computing model since it can provide a flexible, dynamic, resilient, and cost-effective infrastructure. When multiple internal and/or external cloud services are incorporated, we can get a distributed cloud environment, i.e., multicloud. The clients can access his/her remote resource through interfaces, for example, Web browser. Generally, cloud computing has three deployment models: public cloud, private cloud, and hybrid cloud. Multicloud is the extension of hybrid cloud. When multicloud is used to store the clients' data, the distributed cloud storage platforms are indispensable for the clients' data management.[1] Of course, multicloud storage platform is also more vulnerable to security attacks. For example, the malicious CSPs may modify or delete the clients' data since these data are outside the clients. To ensure the remote data' security, the CSPs must provide security techniques for the storage service.

In 2007, Ateniese et al. proposed the PDP model and concrete PDP schemes. It is a probabilistic proof technique for CSPs to prove the clients' data integrity without downloading the whole data. After that, Ateniese et al. proposed the dynamic PDP security model and the concrete dynamic PDP schemes. To support data insert operation, Erway et al. proposed a fulldynamic PDP scheme based on authenticated flip table. Since PDP is an important lightweight remote data integrity checking model, many researchers have studied this model.[2]

In 2012, Zhu et al. proposed the PDP model in distributed cloud environment from the following aspects: high security, transparent verification, and high performance. They proposed a verification framework for multicloud storage and constructed a CPDP scheme which is claimed to be provably secure in their security model. Their scheme took use of the techniques: hash index hierarchy (HIH), homomorphic verifiable response, and multiprover zero-knowledge proof system.[3] They claimed that their scheme satisfied the security properties: completeness, knowledge soundness, and zero-knowledge. These properties ensure that their CPDP can implement the security against data leakage attack and tag forgery attack.

In this comment, we show that Zhu et al.'s CPDP scheme does not satisfy the property of knowledge soundness. The malicious CSPs or organizer can cheat the clients. Then, we discuss the origin and severity of the security flaws. Our work can help cryptographers and engineers design and implement more secure and efficient CPDP scheme for the multicloud storage.

EXISTING SYSTEM

The existing various tools and technologies for multi cloud, such as Platform VM Orchestrator, VMwarevSphere, and Ovirt. These tools help cloud providers construct a distributed cloud storage platform for managing clients' data. However, if such an important platform is vulnerable to security attacks, it would bring irretrievable losses to the clients.[4] For example, the confidential data in an enterprise may be illegally accessed through a remote interface provided by a multi-cloud, or relevant data and archives may be lost or tampered with when they are stored into an· uncertain storage pool outside the enterprise. Therefore, it is indispensable for cloud service providers to provide security techniques for managing their storage services.

PROPOSED SYSTEM

To check the availability and integrity of outsourced data in cloud storages, we researchers have proposed two basic approaches called Provable Data Possession and Proofs of Retrievability . We first proposed the PDP model for ensuring possession of files on untrusted storages and provided an RSA-based scheme for a static case that achieves the communication cost. We also proposed a publicly verifiable version, which allows anyone, not just the owner, to challenge the server for data possession..We then proposed a lightweight PDP scheme based on cryptographic hash function and symmetric key encryption but the servers can deceive the owners by using previous metadata or responses due to the lack of randomness in the challenges. The numbers of updates and challenges are limited and fixed in advance and users cannot perform block insertions anywhere.

Literature Survey

1. Ensuring Data Storage Security in Cloud Computing

Cloud Computing has been envisioned as the next generation architecture of IT Enterprise. In contrast to traditional solutions, where the IT services are under proper physical, logical and personnel controls, Cloud Computing moves the application software and databases to the large data centers, where the management of the data and services may not be fully trustworthy. This unique attribute, however, poses many new security challenges which have not been well understood. In this article, we focus on cloud data storage security, which has always been an important aspect of quality of service.[5] To ensure the correctness of users' data in the cloud, we propose an effective and flexible distributed scheme with two salient features, opposing to its predecessors. By utilizing the homomorphic token with distributed verification of erasure-coded data, our scheme achieves the integration of storage correctness insurance and data error localization, i.e., the identification of misbehaving server(s). Unlike most prior works, the new scheme further supports secure and efficient dynamic operations on data blocks, including: data update, delete and append. Extensive security and performance analysis shows that the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks.

Disadvantages

1. There is no feature of automatic blocking the cloud server attackers.

2. Less Security – No cryptographic technique is used on the cloud data

2. Privacy-Preserving Audit and Extraction of Digital Contents

A growing number of online services, such as Google, Yahoo!, and Amazon, are starting to charge users for their storage. Customers often use these services to store valuable data such as email, family photos and videos, and disk backups. Today, a customer must entirely trust such external services to maintain the integrity of hosted data and return it intact.[6] Unfortunately, no service is infallible. To make storage services accountable for data loss, we present protocols that allow a third-party auditor to periodically verify the data stored by a service and assist in returning the data intact to the customer. Most importantly, our protocols are privacy-preserving, in that they never reveal the data contents to the auditor. Our solution removes the burden of verification from the customer, alleviates both the customer's and storage service's fear of data leakage, and provides a method for independent arbitration of data retention contracts.

Disadvantages

1. The data integrity is proving only based on the filename and not on the public key or any other key.

2. The attackers details are not dynamic instead its maintaining the log files to store the attacker details and viewing using data mining concepts which is time consuming job and less security.

3. Provable Data Possession at Untrusted Stores

We introduce a model for provable data possession (PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage systems. We present two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees.[7] In particular, the overhead at the server is low (or even constant), as op- posed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and re- veal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

Disadvantages

1. There is no feature of automatic blocking the cloud server attackers.
2. Owner data will be stored in untrusted cloud servers.

MODULE DESCRIPTION:

Multi cloud storage

Distributed computing is used to refer to any large collaboration in which many individual personal computer owners allow some of their computer's processing time to be put at the service of a large problem. In our system the each cloud admin consist of data blocks. The cloud user uploads the data into multi cloud. cloud computing environment is constructed based on open architectures and interfaces, it has the capability to incorporate multiple internal and/or external cloud services together to provide high interoperability. We call such a distributed cloud environment as a *multi-Cloud* .A multi-cloud allows clients to easily access his/her resources remotely through interfaces.

Cooperative PDP

Cooperative PDP (CPDP) schemes adopting zero-knowledge property and three-layered index hierarchy, respectively. In particular efficient method for selecting the optimal number of sectors in each block to minimize the computation costs of clients and storage service providers. Cooperative PDP (CPDP) scheme without compromising data privacy based on modern cryptographic techniques

Data Integrity

Data Integrity is very important in database operations in particular and Data warehousing and Business intelligence in general. Because Data Integrity ensured that data is of high quality, correct, consistent and accessible.

Third Party Auditor

Trusted Third Party (TTP) who is trusted to store verification parameters and offer public query services for these parameters. In our system the Trusted Third Party, view the user data blocks and uploaded to the distributed cloud. In distributed cloud environment each cloud has user data blocks. If any Modification tried by cloud owner an alert is send to the Trusted Third Party.

Cloud User

The Cloud User who has a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data. The User's Data is converted into data blocks. The data blocks are uploaded to the cloud. The TPA views the data blocks and Uploaded in multi cloud. The user can update the uploaded data. If the user wants to download their files, the data's in multi cloud is integrated and downloaded.

MODULE DESCRIPTION:

Multi cloud storage

Distributed computing is used to refer to any large collaboration in which many individual personal computer owners allow some of their computer's processing time to be put at the service of a large problem. In our system the each cloud admin consist of data blocks. The cloud user uploads the data into multi cloud. cloud computing environment is constructed based on open architectures and interfaces, it has the capability to incorporate multiple internal and/or external cloud services together to

provide high interoperability. We call such a distributed cloud environment as a *multi-Cloud* .A multi-cloud allows clients to easily access his/her resources remotely through interfaces.

Cooperative PDP

Cooperative PDP (CPDP) schemes adopting zero-knowledge property and three-layered index hierarchy, respectively. In particular efficient method for selecting the optimal number of sectors in each block to minimize the computation costs of clients and storage service providers. Cooperative PDP (CPDP) scheme without compromising data privacy based on modern cryptographic techniques

Data Integrity

Data Integrity is very important in database operations in particular and Data warehousing and Business intelligence in general. Because Data Integrity ensured that data is of high quality, correct, consistent and accessible.

Third Party Auditor

Trusted Third Party (TTP) who is trusted to store verification parameters and offer public query services for these parameters. In our system the Trusted Third Party, view the user data blocks and uploaded to the distributed cloud. In distributed cloud environment each cloud has user data blocks. If any Modification tried by cloud owner an alert is send to the Trusted Third Party.

Cloud User

The Cloud User who has a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data. The User's Data is converted into data blocks. The data blocks are uploaded to the cloud. The TPA views the data blocks and Uploaded in multi cloud. The user can update the uploaded data. If the user wants to download their files, the data's in multi cloud is integrated and downloaded.

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. [8] 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

- ECONOMIC FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMIC 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.[9]

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. [10]

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.[11] 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.

SYSTEM DESIGN

ARCHITECTURE DIAGRAM

Figure 1: Architecture Diagram

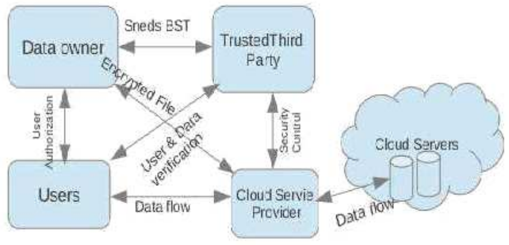


Figure 2: Flow Chart

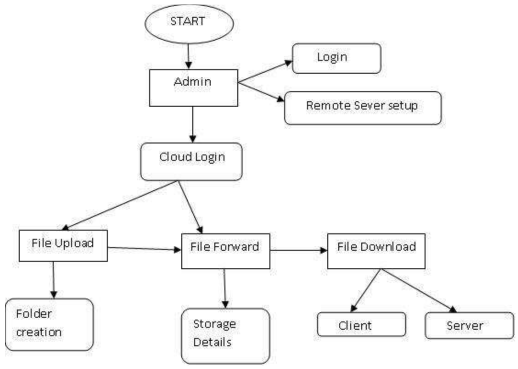


Figure 3: Data Flow Diagr

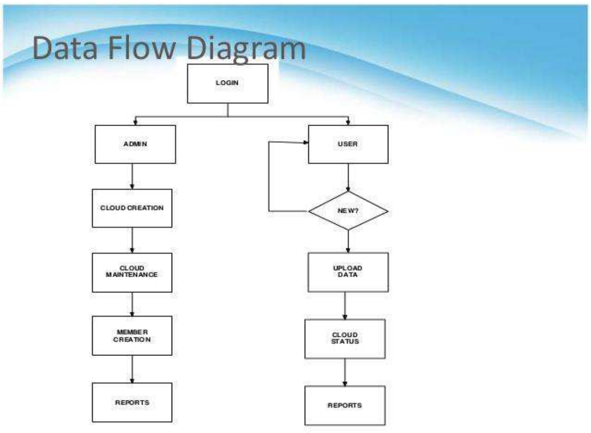


Figure 4: Use Case Diagram

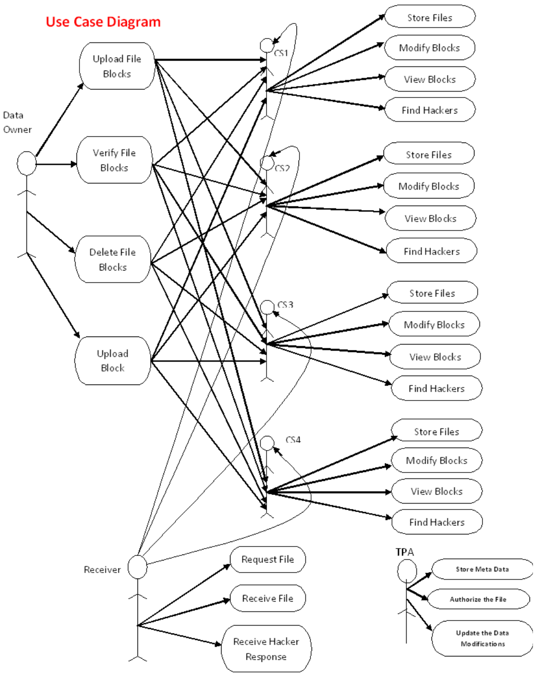
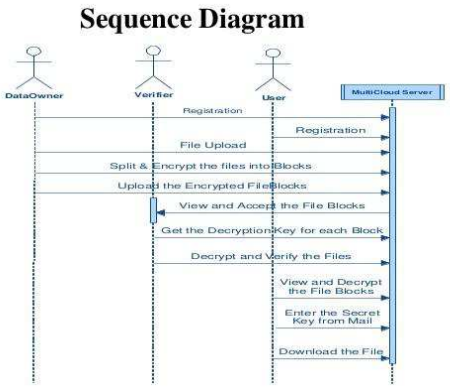


Figure 5: Sequence Diagram


Sequence Diagram
DataOwner
Verifier
User
MultiCloud Server
Registration
Registration
File Upload
Split & Encrypt the files into Blocks
Upload the Encrypted FileBlocks
View and Accept the File Blocks
Get the Decryption Key for each Block
Decrypt and Verify the Files
View and Decrypt the File Blocks
Enter the Secret Key from Mail
Download the File

Project Life Cycle

The following diagram depicts the general design and development p

Figure 6: Pr le

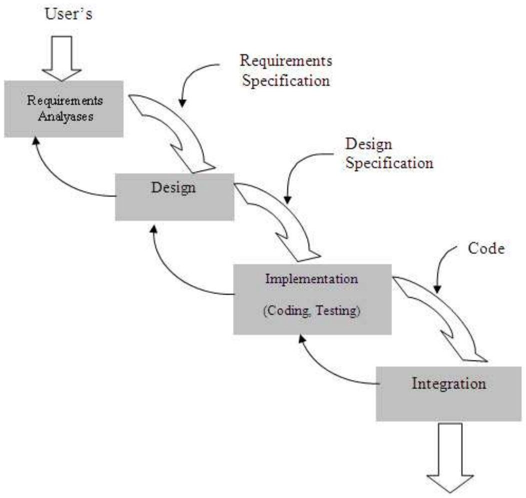
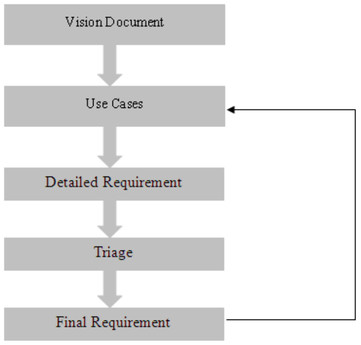


Figure 7: Requirement Analysis



PROTOTYPE

This stage involves deriving classes and class diagrams, table structures, business rules, and forms.

Classes

These are derived from Use Cases. A class can be of an actor or an artifact. For a class all possible attributes and methods are identified. For e.g. A customer is a class then his Name, Address etc. can be the attributes.

Class diagrams

Depict the relationship between classes i.e whether they are one to many ,one to one, many to one or many to many. Once these are established detailed data structures are arrived at and developed. Other than this detailed class diagrams for forms are also drawn at this stage.

Detailed development plan

This is also drawn at this stage and actual development starts.

Business rules are arrived at but not coded

The prototype goes in for quality control. Again a bug-tracking tool is used here to record errors and track their statuses. A check sheet is used to perform basic quality control procedures. This check sheet is consistently upgraded from the accumulated errors from the bug-tracking tool.

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.

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

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.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTENANCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY :

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

RESULTS & CONCLUSION

In this research paper, we point out some flaws in Zhu et al.'s CPDP scheme for integrity verification in multicloud storage. Through cryptanalysis, we find that their CPDP scheme does not satisfy the knowledge soundness. Thus, Zhu et al.'s CPDP scheme is insecure. It is still an open problem to design secure and efficient CPDP scheme for integrity verification in multicloud storage.

REFERENCES

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 598-609, 2007.

[2] G. Ateniese, R. Dipietro, L.V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proc. Fourth Int'l Conf. Security and Privacy in Comm. Networks (SecureComm '08), 2008.

[3] C.C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic Provable Data Possession," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), pp. 213-222, 2009.

[4] F. Sebe, J. Domingo-Ferrer, A. Martinez-balleste, Y. Deswarte, and J. Quisquater, "Efficient Remote Data Possession Checking in Critical Information Infrastructures," IEEE Trans. Knowledge and Data Eng., vol. 20, no. 8, pp. 1034-1038, Aug. 2008.

[5] Y. Zhu, H. Wang, Z. Hu, G.J. Ahn, H. Hu, and S.S. Yau, "Efficient Provable Data Possession for Hybrid Clouds," Proc. 17th ACM Conf. Computer and Comm. Security (CCS '10), pp. 756-758, 2010.

[6] H. Wang, "Proxy Provable Data Possession in Public Clouds," IEEE Trans. Services Computing, DOI: 10.1109/TSC.2012.35.

[7] Y. Zhu, H. Hu, G.J. Ahn, and M. Yu, "Cooperative Provable Data Possession for Integrity Verification in MultiCloud Storage," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231-2244, Dec. 2012.

[8] L. Fortnow, J. Rompel, and M. Sipser, “On the Power of Multi-Prover Interactive Protocols,” Theoretical Computer Science, pp. 156-161, 1988.

[9] D. Boneh and M. Franklin, “Identity-Based Encryption from the Weil Pairing,” Proc. 21st Ann. Int’l Cryptology Conf. Advances in Cryptology (CRYPTO ’01), pp. 213-229, 2001.

[10] A. Miyaji, M. Nakabayashi, and S. Takano, “New Explicit Conditions of Elliptic Curve Traces for FR-Reduction,” IEICE Trans. Fundamentals, vol. 5, pp. 1234-1243, 2001.

[11] D. Boneh, H. Shacham, and B. Lynn, “Short Signatures from the Weil Pairing,” J. Cryptology, vol. 17, no. 4, pp. 297-319, 2004.