International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-4, April 2019

Cloud Runtime Framework for User-Level Security Auditing for the Cloud System

Tejas Sohani, S. Pandiaraj M.E, Jay Patel, Abhishek Samal, Aranish Dutta

Abstract-In recent years, For almost all IT services, cloud computing has emerged as an indispensable solution because of its ease of implementation and accessibility and the services that it provides connecting millions of users over the internet. However, the widespread adoption of cloud services into the technologies has its own difficulties starting from lack of transparency and accountability to threats and vulnerabilities. Traditional auditing techniques have its set of challenges which include lack of heterogeneity and data inconsistency. Runtime auditing techniques offer a solution to counter these problems although existing techniques cannot provide a practical solution when large clouds containing numerous datasets are concerned. This paper proposes a runtime auditing framework at user-level to improve security at the most vulnerable stage of the cloud architecture. It focuses on authentication and common access control mechanisms to implement an OpenStack based framework which is a widely deployed cloud management system. The focus is to reduce the response time and perform operations in a more effective and efficient manner. This helps to implement auditing on large cloud environments in a realistic time frame making the whole auditing process more efficient.

KEYWORDS - Auditing, Cloud-Security, Authentication and Runtime Framework.

I. INTRODUCTION

Cloud-based auditing presents various unique challenges related to data collection and processing (e.g., data format inconsistency, lack of correlation due to the heterogeneity of cloud infrastructure) and verification (e.g., restrictive overhead performance due to the sheer scale of cloud infrastructure and the requirement for runtime verification due to cloud dynamic nature). Existing auditing techniques do not offer a pragmatic response time to verify a wide range of user-level security properties in a large cloud for this purpose. It is possible to divide existing approaches into three categories.

- First, retroactive approaches that capture post - process security breaches.
- Second, intercept and verify security invariants for each user request before they are granted or denied.
- Third, proactive approaches pre - check user requests.

Manuscript published on 30 April 2019.
* Correspondence Author (s)
Tejas Sohani*, CSE Dept., SRM University, Ramapuram, Chennai, India
S. Pandiaraj M.E, CSE Dept., SRM University, Ramapuram, Chennai, India
Jay Patel, CSE Dept., SRM University, Ramapuram, Chennai, India
Abhishek Samal, CSE Dept., SRM University, Ramapuram, Chennai, India
Aranish Dutta, CSE Dept., SRM University, Ramapuram, Chennai, India

II. EXISTING SYSTEM

Cloud auditing gives rise to different challenges in data collection as well as data processing like data inconsistency, lack of correlation etc. and in verification and authentication. Existing techniques do not offer an optimal solution for verifying user-level security properties. The categories of existing approaches are

1. the retroactive approaches which catch security violations following the process
2. interceptand check approaches that verify security invariants for every user request to grant or deny it
3. proactive approaches that verify user requests in advance

The disadvantages of the existing systems include operational complexity, misconfigurations and vulnerabilities resulting in the violation of security properties.

III. PROPOSED SYSTEM

A user-level auditing framework for auditing at runtime for the cloud system is proposed here. It focuses specifically on user - level auditing using common methods of access control and authentication mechanisms such as RBAC, ABAC, SSO, and on OpenStack, a widely deployed cloud management system, the framework is evaluated.Response time of the system is reduced to an optimum level so that costly operations are executed just once and thus results in more efficient runtime verification. In a multi-domain cloud environment, the auditing framework is proposed. Based on authorization, authentication, and common cloud security standards, a set of security properties is compiled. During the initial phase, costly auditing operations are performed only once so that runtime operations can be performed in an incremental procedure to reduce the cost and delay of runtime operations Formal verification methods are used to enable automated reasoning and provide formal evidence or, rather, counter-compliance examples.

Advantages of the proposed system:

1. Can potentially prevent limitation of retroactive approaches
2. Requires no future change plan
3. Significantly reduces response time
4. Supports a lot of user-level security properties

Authentication Mechanisms:

RBAC Model

Focuses on multi - domain role - based access control (RBAC) verification that is generally used in cloud platforms in real - world. The extended RBAC model adds cloud support for multi - tenancy.

ABAC Model

Used for large scale infrastructures. Major cloud platforms implement ABAC. There are mainly 2 ABAC functions: User Attribute(UATT) and Object Attribute(OATT)

SSO Mechanism

A popular cloud authentication extension which is supported by major cloud platforms. SSO makes use of a single user action for granting access to every authorized computer systems. The two SSO protocols are OpenID and SAML.

IV. SYSTEM ARCHITECTURE

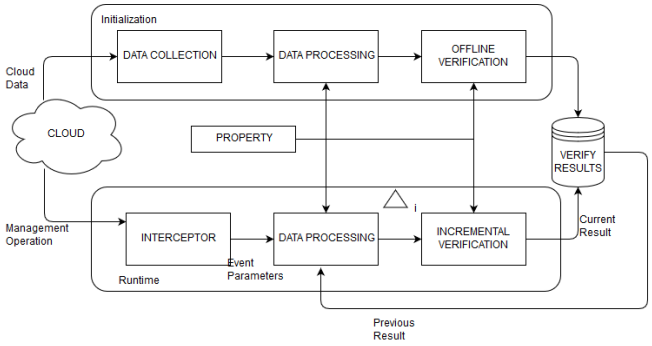

Fig .1 System Architecture

When data from the cloud is passed, it enters into an initialization phase and a runtime phase. This is referred to as data collection. This is followed by the data processing phase where the data is processed during runtime. This proceeds the data into the verification phase where offline verification and incremental verification takes place. Incremental verification ensures lesser costs of processing. The verified results that are audited are finally entered into the database.

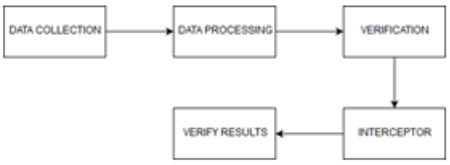

Fig .2 Data Flow representation

V. LITERATURE REVIEW

a) Cross Tenant Access Control (CTAC) Model for Cloud Computing: Formal Specification and Verification

Due to the cost-effective and location independent features of cloud computing, large scale sharing of resources on a cloud is considered practical. Yet, because of concerns in security for the sharing of resourceson cloud environments, organizations are often reluctant to adopt cloud environments for their businesses. In this paper, cloud service providers offer mediation services which act as a third entity between multiple tenants. In the presence of cloud resource mediation services, it defines a resource sharing mechanism between tenants. 4 Algorithms namely Activation, Delegation, Forward Revocation and Backward Revocation are demonstrated by the use of formal verification for the correctness of the authorisation, activation and delegation mechanism between the different tenants. On the basis of the performance analysis is states that resource sharing among different tenants can be performed securely and efficiently.

b) Auditing a Cloud Provider's Compliance with Data Backup Requirements: A Game Theoretical Analysis

Frequent challenges arise on the confidentiality, integrity,and availability of outsourced data with new developments in cloud computing. The cloud provider and user agree on a Service Level Agreement(SLA). There are many security mechanisms for checking outsourced data's integrity and availability. The tasks can be carried out either by the user or by an independent identity, although the task is accompanied by an additional cost that can often prevent the customer from performing data verification. Using game theory, the interaction between the verifier and the provider can be observed to find a practical strategy for verifying data. The issue is formulated as a non-cooperative two-player game. Depending on a set of parameters such as size and sensitivity, each type of data is replicated several times. The Nash Equilibrium helps to analyze strategies of boththe provider and the verifier to derive expected behaviors.

c) b-SPECS+: Batch Verification for Secure Pseudonymous Authentication in VANET

Security and privacy preservation are considered pre-requisitesin the vehicular ad-hoc networks (VANETs). Recently, a scheme called SPECS (Secure and privacy enhancing communication schemes) has been proposed focusing on inter-vehicle communication. SPECS provides a software-based solution to meet privacy requirements along with lower overhead message delivery and higher success rates compared to previous messaging verification solutions.Also, it provides a protocol for group communication for authenticating and communicating securely with other vehicles in the group. In spite of these factors, It is susceptible to impersonation attacks. It has a flow which can help malicious vehicles send fake messages in a group. A secure scheme is provided for achieving privacy and security requirements and counter the vulnerabilities of SPECS. Also, the merits regarding the efficiency of the scheme are described through performance evaluations in terms of delay in verification and overhead transmission.

d) **Privacy-Preserving Public Auditing For Secure Cloud Storage**

With cloud storage mechanisms, users are able to access various internet services. They can benefit from on-demand services and applications from the resources. The cloud has to ensure data integrity and security for the user which is the main issue with respect to cloud storage. To take care of this issue, a public auditing process involving third-party auditors(TPA) is used. The proposed system also supports data dynamics and true public auditability. The auditing procedure is used to monitor data modifications, insertions,and deletions. The system supports public auditability, data dynamics and Multiple TPA which are used for the auditing process. The HARS scheme is used for ring signatures. Block level authentication is improved using Merkle Hash Trees. The TPA is enabled to perform audits for multiple users simultaneously through Batch auditing process.

e) Securing Loosely-coupled Collaboration in Cloud Environment through Dynamic Detection and Removal of Access Conflicts

Software-as-a-Service (SaaS) clouds have made online collaboration services very popular. It helps in sharing information between multiple domains and accessibility from remote locations. In the form of a set of permissions, access requests are sent. To enable resource access, the permissions are mapped into local roles. However, multiple requests for access may violate security principles. A secure collaboration framework distributed allows collaborative domains to detect and eliminate these conflicts. Somecharacteristics of the framework are: (i) requiresjust local information (ii) detection, removal of conflicts on the go. Formal proofs, experimental results,and qualitative comparison establishes correctness addresses the scalability requirements of cloud services.

VI. TEST DATA AND OUTPUT

The biggest challenge with the cloud security testing is the lack of application logging which deprives in focusing and enhancing the test efforts and the results. performing the security test in an isolated environment will only help us in generating logs for our considered attacks, which leaves us in only be able to gauge the attack success by the application's behavior. Cloud testing majorly focuses on Application, Network,and infrastructure. Testing of functions, Business workflows, data security, browser or launching environment compatibility etc. falls under the application security testing. Network security testing majorly focuses on bandwidths, protocols,and successful data transfer. whereas infrastructure testing covers disaster recovery. backups, connection security in terms of ciphering or encryption and storage policies.

a) Unit Testing: Unit testing verified the functional performance of each module of the cloud security component. a major focus was on the smallest unit considered by the internal team. White box testing techniques are vastlyimplementedin unit testing.

b) Functional Testing: In this type of testing, Test cases involve exercising nominal input values for which expected results are already known, as well as limit values and special values such as logically related inputs, identical element files and empty files. functional testing should produce the expected results for the valid inputs. Performance test, Stress test,and structure test also can be considered as types of functional test. While performance test helps in determining and inspecting the Time spent executing different parts of the unit, program output and resources utilized to carry out the process or the task and checks for the case where failure due to one user should not affect the other users using the system and system compatibility and performance across all the devices should remain the constant. A stress test is designed to test the unit in extreme conditions. It helps in learning the limitations of the system like a number of users can access the service provided by the cloud and how the system reacts over time under the specific load. Whereas the structured tests are concerned with the program logic and its paths of execution. The ways in which white box testing strategy is used to ensure that test cases can ensure that all independent module paths are exercised at least once. It works on both their true and false sides around all the logical decisions. To ensure their validity, execute each loop at their boundary and the internal data structure. It checks for attributes to verify their correctness and handles EOF conditions, I/O errors, buffer problems.

c) Security Testing: it is all about various security functions and parameters implemented in the system. it states that only authorized person can get access to the data, which must be encrypted in order to achieve protection against many eavesdropping attacks. there should not be inefficient or insufficient encryption. data belongs to the user should be deleted once the user account is expired or the data is not in use. also, it checks for various security parameters like firewall, VPN, anti-virus etc.

d) Network Testing: Network testing is all about testing the protocols responsible for maintaining connectivity and ensuring security. It checks for the data integrity and while transferring the data which includes the data packet dropping due to node failure and firewall restrictions.

e) Integration Testing: A systematic technique used to construct the structure of the program while simultaneously carrying out tests to detect errors associated with interfacing. Integration testing is the complete testing of the product's set of modules. The goal is to take untested modules and build a tester to identify critical modules in the program structure. It is necessary to test critical modules as early as possible.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

One approach is to wait until the testing phase is completed by all units and then combine them before testing again. This approach has evolved from small programs' unstructured testing.Another strategy is to build the product on the basis of tested units in increments. Integrated and tested a small set of modules, adding and testing another module in combination, and so on. The advantage of this approach is that it is easy to find and correct the interface dispenses. The major error faced during the project is a link error, i.e. the link is not set correctly with all support files when all the modules are combined. Then the interconnection and the links are checked. The new module and its intercommunications are located with errors. Development of the product can be staged and modules integrated as unit tests are completed.When the last module is integrated and tested, testing will be stopped.

VII. TESTING TECHNIQUES

TESTING: Testing is a process in which a program is executed in order to find an error. A good test case is one that has a high likelihood of finding an error that has not yet been discovered. A successful test is one that detects an error that has not yet been detected. System testing is the implementation stage, which is designed to ensure that the system functions accurately and efficiently as expected before starting live operation. It checks that all the programs are hanging together. System testing requires a test consisting of several key activities and steps for running program, string, system and is important for the adoption of a new system that is successful. This is the last chance for user acceptance testing to detect and correct errors before the system is installed. Once the program is created, the software testing process starts designing the documentation and related data structures. To correct errors, software testing is essential. Otherwise, it is not said that the program or the project is complete. Software testing is a critical component of quality assurance of software and represents the ultimate review of design and coding specifications. Testing is the execution process of the program for the purpose of finding the error. A good design of a test case is one that is likely to find an undiscovered error. A successful test is one that reveals an error that has not yet been discovered. It is possible to test any engineering product in one of the two ways:

White Box Testing: This type of test is also referred to as Glass box testing. In this testing, it can be led by knowing the specific capacities that an item was intended to perform the test that each capacity is fully operational in the meantime, in each capacity, looking for blunders. It is an experiment plan strategy that utilizes the control structure of the procedural structure to infer experiments. Premise way testing is a white box testing. which fundamentally works with the Flow chart documentation, Cyclometric complexities and Graph frameworks control likewise it incorporates the deductions of the experiments.

Black Box Testing: In this testing, knowing the inner task of an item, the test can be directed to ensuring that "all appliances work," that is, the inward activity is carried out as indicated by particular and every single inner segment has been enough worked out. It on a very basic level spotlights on the utilitarian prerequisites of the product. discovery testing can be accomplished with Graph-based testing techniques, Equivalence apportioning, comparison testing, Boundary value analysis,. We often face many challenges while testing any cloud system. Which can be with respect to Data security and privacy. Where the risk of data theft is always there as we are considering the cloud for multi-tenant applications. Also, it practically very tough to consider all the scenarios which can lead to data theft or can take a form of attack. Maintaining the stability and the availability while providing the functionality upgrade or the interface upgrade. Data migration from one cloud to another cloud provider is not very frequent but a major challenge in terms of cloud testing.

VIII. SOFTWARE TESTING STRATEGIES

Strategies for software testing provide the software developer with a predefined plan. Testing is regarded as a set activity that is systematically planned and carried out in advance.So, a template for software testing consists of a set of steps where specific test case design methods can be implemented. Software Testingstrategiesare supposed to have these characteristics:

1. Testing starts at the level of the module and works outwards towards the full computer - based system integration.
2. It is possible to use different testing techniques at different times.
3. Testing should be carried out by the software developer as well as independent test groups.
4. Testing and debugging are two different activities, but each testing strategy must include debugging.

IX. TOOLS AND APPLICATIONS

a)STRUCTURED QUERY LANGUAGE (SQL)

SQL or Structured Query Language is a data control programming language that is utilized for managing relational databases and perform different activities to manage and control information present in the databases. SQL is particularly utilized by database administrators, yet additionally by database engineers/ developers working on data integration and similar platforms and data analyst to run analytical queries. SQL can modify the database table and their structures. It inserts, updates, deletes rows of data and retrieves information from databases when needed for the transactional process and analytical applications. Queries and other SQL operations make up the commands which are written as query statements. The commonly used SQL statements include DDL, DML and TCL queries. SQL is the standard programming language for relational databases known as SQL databases. Relational systems comprise ofa collection of tables containing rows and columns containing data in a structured format.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

Everycolumn in a database table corresponds to a unique category of data.

b)**SQL STANDARD AND PROPRIETARY EXTENSIONS**

The American National Standards Institute (ANSI) in 1986 followed by International Organization for Standardization, known as ISO, in 1987 adopted an official SQL standard w More than half a dozen of joint updates were released by both the standards development bodies SQL:2018 is the latest version of SQL. Both proprietary and open source RDBMS built with SQL and are available for use by organizations which includes Oracle database, Microsoft SQL Server, SAP HANA,IBM DB2, SAP Adaptive Server,MySQL,and PostgreSQL.

c)**SQL COMMANDS AND QUERIES**

SQL statements are of different types, (DML) data manipulation language statements and (DDL) data definition language statements, (TCL) transaction control language which comprises of transaction controls and security measure statements.DML statementsare used to manipulate data in the database, and DDL statements alter the structure of the database. The transaction controls help manage transaction includes commit and rollback functionalities. The security statements are used to control user roles and permissions.

d)**PYTHON PROGRAMMING LANGUAGE**

Python is a high-level, interpreted language. Created by Guido released in 1991, Python enhances code readability. It provides constructs that enable distinguished programming at every scale.

Python has automatic memory management and dynamic type system. It supports multiprogramming and threading, including object-oriented procedural programming with a large standard library.

Python interpreters are compatible with many operating systems. CPythonis open source software references Python implementation and has a community-based development model managed by Python Software Foundation.

e)**LiCLIPSE EDITOR**

It is a typeof light-weight editor for theming and also usability improvements used for Eclipse. It provides a different experience for Eclipse users.

It provides the following features:

- It has a fast editor supporting many languages.
- Provide support for Text Mate Bundles.
- Gives a simpler way for adding support for a new language
- Usability improvements on eclipse editors include:
- Multiple cursors
- Vertical indent guides
- Themed scrollbars
- Improved search with Lucene support for external folders, index-based searching, additional filtering on page results and open editors.
- Gives a preview of HTML for RST, Markdown and HTML editors
- Native installers
- Support for theming based on eclipse for improvements

REFERENCES

1. OpenStack, “OpenStack user survey,” 2016, available at:https://www.openstack.org.
2. E. Aguiar, Y. Zhang, and M. Blanton, “An overview of issues andrecent developments in cloud computing and storage security.” in High-Performance Cloud Auditing and Applications, 2014.
3. K. Ren,C. Wang, and Q. Wang, “Security challenges for the publiccloud.” IEEE Internet Computing, no. 1, pp. 69–73, 2012.
4. T. Madi, S. Majumdar, Y. Wang, Y. Jarraya, M. Pourzandi, and L. Wang,“Auditing security compliance of the virtualized infrastructure in thecloud: Application to OpenStack.” in CODASPY, 2016.
5. OpenStack, “OpenStack open source cloud computing software.” 2015,available at http://www.openstack.org.
6. OpenStack, “OpenStack Congress.” 2015, available at:https://wiki.openstack.org/wiki/Congress.
7. V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, andK. Scarfone, “Guide to attribute based access control (ABAC) definitionand considerations.” NIST SP, vol. 800, 2014.
8. N. Pustchi and R. Sandhu, “MT-ABAC: A multi-tenant attribute-basedaccess control model with tenant trust.” in NSS, 2015.
9. R. Sandhu, “The authorization leap from rights to attributes: maturationor chaos?” in SACMAT, 2012.
10. OASIS, “Security assertion markup language (SAML).” 2016, availableat: http://www.oasis-open.org/committees/security.
11. OpenID Foundation, “OpenID: the internet identity layer.” 2016, availableat: http://openid.net.
12. T. Groß, “Security analysis of the SAML single sign-on browser/artefactprofile.” in ACSAC, 2003.
13. H.-K. Oh and S.-H. Jin, “The security limitations of SSO in OpenID.” in10th International Conference on Advanced Communication Technology,2008.
14. A. Armando, R. Carbone, L. Compagna, J. Cuellar, and L. Tobarra,“Formal analysis of SAML 2.0 web browser single sign-on: breakingthe SAML-based single sign-on for Google apps.” in ACM FMSE, 2008.
15. ISO Std IEC, “ISO 27002:2005.” Information Technology-Security Techniques,2005.
16. A. Gouglidis, I. Mavridis, and V. C. Hu, “Security policy verification formulti-domains in cloud systems.” Int. Jour. of Info. Sec., 2014, 13(2).
17. getcloudify..org, “OpenStack in numbers - the real stats.” 2014, availableat: http://getcloudify.org.
18. OpenStack, “OpenStack audit middleware.” 2016,available at:http://docs.openstack.org/developer/keystonemiddleware
19. F. Doelitzscher, “Security audit compliance for cloud computing.” Ph.D.dissertation, Plymouth University, 2014.
20. D. Petcu and C. Craciun, “Towards a security SLA-based cloud monitoringservice.” in CLOSER, 2014.
21. C.Wang, S. S. Chow, Q.Wang, K. Ren, and W. Lou, “Privacy-preservingpublic auditing for secure cloud storage.” IEEE TC, 2013.
22. H. Kai, H. Chuanhe, W. Jinhai, Z. Hao, C. Xi, L. Yilong, Z. Lianzhen,and W. Bin, “An efficient public batch auditing protocol for data securityin multi-cloud storage.” in ChinaGrid, 2013.
23. Z. Ismail, C. Kiennert, J. Leneutre, and L. Chen, “Auditing a cloudprovider’s compliance with data backup requirements: A game theoreticalanalysis.” IEEE TIFS, 2016.
24. Y. Wang, Q. Wu, B. Qin, W. Shi, R. H. Deng, and J. Hu, “Identity-baseddata outsourcing with comprehensive auditing in clouds.” IEEE TIFS,2017.
25. K. Fisler, S. Krishnamurthi, L. A. Meyerovich, and M. C. Tschantz,“Verification and change-impact analysis of access-control policies.” inICSE, 2005.
26. S.-J. Horng, S.-F. Tzeng, Y. Pan, P. Fan, X. Wang, T. Li, and M. K. Khan,“b-SPECS+: Batch verification for secure pseudonymous authenticationin VANET.” IEEE TIFS, 2013.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

BIOGRAPHY

S. Pandiaraj is an assistant professor in the Computer Science and Engineering Department, SRM Institute of Science and Technology, Chennai, India. His research interests are Cloud Technology, Image Processing and Networking.

Tejas Sohani is a student in the Computer Science and Engineering Department, SRM Institute of Science and Technology, Chennai, India. His research interests are Cloud computing, Web Development, and Database Management System.

Jay Patel is a student in the Computer Science and Engineering Department, SRM Institute of Science and Technology, Chennai, India. His research interests are networking and security.

Aranish Dutta is a student in the Computer Science and Engineering Department, SRM Institute of Science and Technology, Chennai, India. His research interests are Web backend Development and Database Management System.

Abhishek Samal is a student in the Computer Science and Engineering Department, SRM Institute of Science and Technology, Chennai, India. His research interests are Cloud Computing and networking.