International Journal of Research

International Journal of Research (IJR)

e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 4, April 2015

Available at http://internationaljournalofresearch.org

Flexible Data-Driven reliable service authentication for Multi-Tenant Cloud Systems

D. Venkatesh; Dr. P. Venkateswarlu & B.V. Srikanth

[1]M.Tech (CSE), Department of Computer Science & Engineering
Nagole Institute of Technology & Science, Kuntloor (V), Hayathnagar (M), RR District, Hyderabad, India.
E-mail id: venkatesh572@gmail.com
[2]Professor & HOD, Department of Computer Science & Engineering.
E-mail id: venkat123.pedakolmi@gmail.com
[3]Associate Professor, Department of Computer Science & Engineering.
E-mail id: bvsrikanth123@gmail.com

Abstract:

Software-as-a service (SaaS) makes utilization of a distributed computing base to convey their applications to numerous clients paying little respect to their area. Due to this offering nature SaaS mists are powerless and give more chances to aggressors to endeavor the framework helplessness and perform vital assaults. In this paper, we exhibit IntTest, a successful administration trustworthiness verification system for SaaS mists. IntTest gives an incorporated diagram authentication examination system that can pinpoint noxious administration suppliers than existing routines. Additionally IntTest will consequently right the debased result that are delivered by the pernicious administration suppliers and supplant it with great results created by considerate administration suppliers. Our exploratory results demonstrate that our plan is powerful and can attain to higher exactness in pinpointing the aggressors than the current methodologies.

Keywords: Service Integrity Attestation, Cloud Computing

1. INTRODUCTION

Cloud computing relies on sharing of resources over a network. Cloud computing mainly focuses on maximizing the effectiveness of the shared resources. Software as a service describes any cloud service where consumers are able to access software applications over the internet. Clouds are providing many types of services like applications, infrastructures etc. Software-as-a-service (SaaS) clouds (e.g., Amazon Web Service (AWS) and Google AppEngine) build upon the concepts of software as a service and service oriented architecture (SOA) which enable application service providers (ASPs) to deliver their applications via the massive cloud computing infrastructure[1].Cloud computing infrastructures are shared by using ASPs from different security domains, because of that its vulnerable. Now days the cloud computing technology is popular because it is an attracting technology in computer science field. This paper concentrate on the integrity attacks on software as a service clouds and because of that the user will receive bad results after processing the data. Fig.1 shows the integrity attacks in software as a service clouds. Majority of software as a service cloud solutions are based on a multitenant architecture. In the previous research papers confidentiality and privacy protection problems are studied extensively but the service integrity attestation problem was not properly addressed. In software as a service cloud one of the most important problems that need to be addressed is this service integrity, no matter whether the data processing in cloud is public or private data. In the previous papers they are provided some software integrity attestation techniques but most of them requires special trusted hardware or secure kernel supports and because of these reasons that cannot be deployed in large scale cloud computing. This paper presents Intent, a new framework for multi tenant cloud systems. This technique provides the novel integrated attestation graph analysis technique that will provide a stronger attacker pinpointing power than the existing schemes. It will automatically enhance the result quality by replacing the bad results that are produced by the attackers by good results that are produced by the benign service providers. This can achieve higher attacker pinpointing accuracy than existing techniques Run Test and Adap Test. Figure 1: Software-as-a Service Specifically, RunTest and AdapTest as well as traditional majority voting schemes need to assume that benign service providers take majority in every service function. In large-scale multitenant cloud systems, large number of malicious attackers may launch colluding attacks on the targeted service functions to make them malicious. To address this challenge, IntTest takes a holistic approach by systematically examining both consistency and inconsistency relationships among different service providers within the entire cloud system. IntTest checks both per-function consistency and the global inconsistency graphs. An advantage of using this IntTest is it cannot only pinpointing the malicious attackers more efficiently but also it can suppress aggressive attackers and also limit the scope of damage that are caused by the attacks. The experimental result shows that IntTest can achieve more accuracy in pinpointing malicious attackers than any other existing schemes. Also this IntTest is more scalable and it will reduce overhead produced by the attestation more than the other voting schemes. This paper implements Efficient and distributed service integrity attestation framework for large scale cloud computing infrastructures. An integrated service integrity attestation scheme that can achieve higher pinpointing accuracy than existing techniques. A result auto correction technique is used that will automatically correct the corrupted results produced by malicious attackers and replace it with good results. The analytical study and experimental evaluation used to quantify the accuracy and overhead of the service integrity attestation method. The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 provides the proposed system in detail. Section 4 presents the main modules. Finally, the paper concludes in section 5.

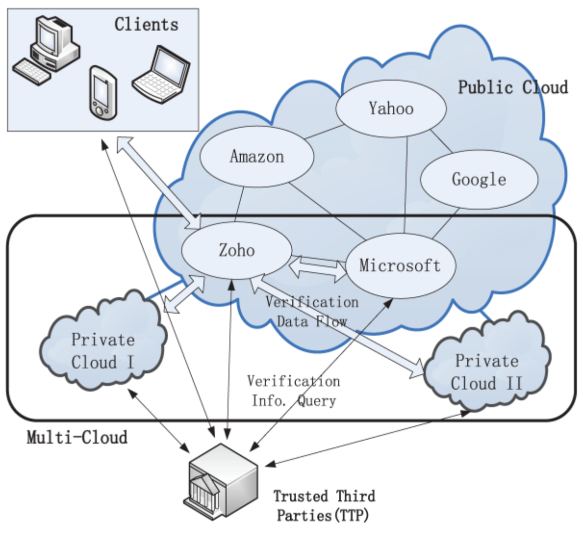

Figure 1: Service integrity attacks in clouds

2. RELATED STUDY

In recent years many integrity attestation schemes have been developed for software as a service clouds. For example the BIND technique, AdapTest technique, RunTest technique etc. but all of these are having some problems some of them needs secure kernel support and special trusted hardware components. In BIND (Binding Information and Data) technique is a verification method of integrity services that are provided by the software as a service cloud system. It was a fine grained attestation framework and can provide the verification through a secure kernel or by a third party. This technique uses the following steps: 1) attestation annotation mechanism 2) sandbox mechanism 3) verification of authenticator through hash. BIND method uses the DiffeeHellman key exchange for the purpose of integrity attestation. Another existing technique is TEAS (Timed Executable Agent System) this is used for protecting the integrity of cloud computing platforms. An agent generation and verification algorithm is used in this TEAS method. Another one existing technique is the runtest, it is a scalable runtime integrity attestation framework. It provides a light weight application level attestation method to assure the integrity of data flow processing in cloud. This will identify the untruthful data flow processing and will pinpoint malicious data processing service provider and at last it will detect the attacker's behavior. This RunTest will provide the benign service providers and will determine the malicious behavior of the attackers. But the disadvantage is its low performance. The AdapTet is another one existing technique, it provides a novel adaptive data driven runtime service integrity attestation framework. This method will significantly reduce the overhead of attestation and will shorten the delay. It treats all components as black boxes and it does not need any special hardware or software requirements. In this AdapTest it will reduce the attestation overhead and the detection of malicious attackers or service providers will be high when compared to other techniques. All the above methods that are used in the existing papers are having some disadvantages. And to overcome that disadvantages this IntTest is using. And by using this IntTest it will provides more integrity and it will provide more accuracy in pinpointing the malicious attackers and service providers. Also it will provide a result auto correction method and will correct the bad results and replace it with good results and also in this it does not require any special hard wares and secure kernel support.

3. PROPOSED SYSTEM

Software as a service and service oriented architecture are the basic concepts of SaaS clouds and this will allow the application service provider to deliver their application via cloud computing infrastructure. In our proposed method we are introducing a new concept called IntTest. The main goal of IntTest is, it can pinpoint all the malicious service providers. IntTest will treat all the service providers as black boxes and this does not need any special hardware or secure kernel support. When we are considering the large scale cloud system multiple service providers may simultaneously compromised by a single malicious attacker. In this we assume that the malicious nodes are not having any knowledge about the other nodes except those which they are directly interacting. In this proposed system we are making some assumptions. First of all we are assuming that the total number malicious service components are less than that of the total number of benign service providers in the entire cloud. This assumptions is very important because without this assumption, it would be difficult for any attack detecting scheme to work successfully. The second assumption is the data processing services are important deterministic. That is, the same input that is giving by a benign service component will always produce the same output. And finally we assume that the inconsistency caused by hardware or software faults can be excluded from malicious attacks. Fig. 2 shows the overall architecture of the proposed system. In this the user give request to cloud the service will be deployed in the cloud will forward the user request to the SaaS and the response will be send to the cloud by the SaaS. And then the IntTest process will be done. After that the result auto correction will be done. After that the result will be send to the user by the cloud. The architecture shows this IntTest module in detail.

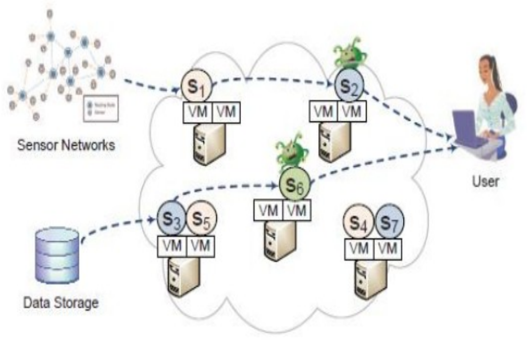

Fig 2. Overall System Architecture

4. TECHNIQUES FOR VERIFYING SERVICE INTEGRITY

In this paper, we will provide a broad overview of the different techniques for verifying the service integrity. We will provide a broad view of the major algorithms available for each method, and the variations on the different techniques. We will also discuss a combination of different concepts

4.1. The BIND Technique

In this section, we will discuss the BIND (Binding information and data) method for the verification of the integrity of services provided by the SaaS cloud system model. It consists of the fine grained attestation framework. It can provide the verification through the secure kernel or third party Auditor. BIND offers the following properties: 1. It attest only to the piece of code we are concerned about. 2. It narrows the gap between time of attestation, time of use and a measure a piece of code immediately before it is executed then uses a sand-boxing mechanism to protect the execution of the attested code.3. It ties the code attestation with data that the code produce, such that we can pinpoint what code has been run to generate that data.

4.1.1 Attestation Annotation Mechanism

In this mechanism, the programmer allowed to identify and annotate the beginning and end of this critical piece of code and every time this piece of code is executed. And this can be validated by checking and verifying checksum value.

4.1.2 SandBox Mechanism

In this, the execution of the critical code will be preserved. An integrity proof for a piece of the input data into the integrity statement of the code and output data. This enable us to achieve transitive integrity verification with constant overhead i.e. we only need to verify one signature to guarantee the integrity of the entire chain of process that transformed the data.

4.1.3 Verification of Authenticator through Hash

It consists of two steps to verify: 1. verify the signature 2. Verify the hash. Since verifying the signature is straightforward, we now explain how to verify the hash and how to enable different software version and software upgrades. BIND allows the application to register one or more legal hash values. We assume that for each application's trusted authority that signs certificate for legal hash values. When an application registers a hash value, it has to show a correct certificate. The public key of an applications trusted authority is included whenever BIND supports various software versions and software upgrades.

4.1.4 Conclusion

This BIND system uses the Diffee-Hellman key exchange for providing the integrity attestation. The existing system of this system is TCG style framework, it uses the coarse grain attestation where it provide attestation for the entire operating system. In this remote verification is difficult and the software will be compromised at runtime. Then next system is COPILOT system framework where it miss the short lived intrusion. Hence these things can be overcome through the BIND system.

4.2 The TEAS Technique

A software scheme for protecting the integrity of computing platforms using Timed Executable Agent Systems(TEAS). A trusted challenger issues an authenticated challenge to a perhaps corrupt responder. A new is that the issued challenge is an executable program that can potentially compute any function on the responder. The responder must compare not only the correct value implied by the agent, but also must complete this computation within time bounds prescribed by the challenger. It also needs some third party auditor to verify the integrity. The algorithm used here is agent generation and verification algorithm. In this both agent and the client generate the checksum value. The generated checksum of client and agent's checksum will be compared then verified by the auditor. It consists of two adversaries 1. off-line adversaries and 2. on-line adversaries.

Off-Line Adversaries In this class we assume that the adversary controlling a client will try to analyze the incoming programs without running them. Recall that in static analysis of program, it can be analyzed in isolation, without inputs and without the state of the machine where they will run. An off-line adversary will perform a similar type of analysis, except that it might also have access to inputs and state of the client.

On-Line Adversaries In this class we assume that the adversary controlling a client will also be able to run the incoming programs.

4.2.1 Conclusion

Finally it can verify the service integrity with the help of third party auditor or secure kernel. The existing system of Timed executable Agent System(TEAS) is verifies the agent through the genuinity system. It also uses the checksum composition. This system is corrupted by the substitution attack. After that this drawback can be overcome through the SWATT system and this system is tightly coupled between target system and trusted host. And then it needs some large memory to access is (O(n log n)). So these drawbacks can overcome through the TEAS system.

4.3 The RunTest Technique

This technique provides a scalable runtime integrity attestation framework. To assure that the integrity of data flow processing in cloud it provides a light weight application level attestation methods. To identify the untruthful data flow processing results and pinpointing malicious data processing service provider and finally detect a colluding attack behavior.

4.3.1 Generation of Integrity Attestation Graph

The malicious can be determined through Integrity Attestation Graph. To capture the aggregated cross node integrity attestation results. It also includes the statistical output of consistency and inconsistency information from different data flow processing nodes. It uses the BronKerbosch(Bk) Clique finding algorithm for finding the consistency cliques in the attestation graph. Proposition 1: All benign service providers always form the consistency clique in the integrity attestation graph. Proposition 2: Any node that is outside of all maximal cliques of size larger than [k/2] in a per-function attestation graph must be a malicious node.

4.3.2 Security Analysis

Our scheme of pinpointing malicious service provider is based on attestation graph analysis. We claim that the scheme preserves the following properties:

Property 1: No false positive: A benign service provider will not be pinpointed as malicious.

Property 2: Non-Repudiation: For any pinpointed malicious service provider ,the trusted portal node can present evidence to prove it is malicious

4.3.3 Data Quality

It defines data quality as the percentage of proposed data with correct results. This scheme can detect the tempered data results probabilistically and report data quality close to actual data quality. Qr =1-(c/n) Qr be the data quality and 1-(c/n) be the actual quality.

4.3.4 Conclusion

Hence this technique can provide the good service provider and also determine malicious behavior. It provides non-repudiation and its demerit is low performance.

4.4 The AdapTest Technique

This presents a novel adaptive data driven runtime service integrity attestation framework for Multitenant cloud system. It can significantly reduce attestation overhead and shorten delay by adaptively selecting attested node based on dynamically derived trust scores. It treats the attested service as Black Boxes and does not impose any special hardware or software requirements on the cloud system or application service provider.

4.4.1 Generation of Weighted Attestation Graph

This graph will provide the trust score for a single node or a pair node.

Definition 1: A weighted Attestation graph is an undirected graph consisting of all functionality equivalent service instances as nodes. The weight of each edge consists of a pair of counters denoting the number of consistent results respectively.

Definition 2: The trust score of the node Si, is defined by αi , is defined as the fraction of consistent results returned by the node Si when attested with all the other nodes. Node trust scores range within [0,1] and are initialized to be 1.

Definition 3: The pair wise trust score between two services instances Si and Sj, denoted by β(i,j),is calculated by the fraction of consistent results when Si is attested against Sj . The pair wise trust score ranges within[0,1] and are initialized to be -1, which means that Si and Sj have not been attested with each other yet. A. Per-Hop Adaptive attestation: In this, the attestation will be provided through clique based algorithm. Initially it selects the suspicious nodes that have low trust scores and attest those suspicious nodes more frequently. B. Multi-Hope Attestation: It also provides the attestation through clique based algorithm. Complicated data processing services often from comprise multiple data processing functions called service hops. Malicious attacker can attack any of the service hops to compromise the final data processing results

5. CONCLUSION

Amongst the three software integrity authentication procedures for multi inhabitant cloud framework, IntTest is the late answer for the issue. Albeit both RunTest and AdpaTest have notable points of interest, IntTest conquers the impediments of the two past methodologies. The IntTest is assessed with an extra metric of false alert rate thus giving more precise results than the past methodologies.

REFERENCE

[1] J. Du, W. Wei, X. Gu, and T. Yu, "Runtest: Assuring Integrity of Dataflow Processing in Cloud Computing Infrastructures," Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS), 2010.

[2] J. Du, N. Shah, and X. Gu, "Adaptive Data-Driven Service Integrity Attestation for Multi-Tenant Cloud Systems," Proc. Int'l Workshop Quality of Service (IWQoS), 2011

[3] J. Du, D. J. Dean, Y. Tan, X. Gu, Ting Yu, "Scalable Distributed Service Integrity Attestation for Software-as-a- Service Clouds" IEEE transactions on parallel and distributed systems, vol. 25, no. 3, march 2014 [4] Software as a Service, http://en.wikipedia.org/wiki/Software as a Service, 2014.

ABOUT AUTHOR

D. Venkatesh pursuing M.Tech (CSE) from Department of Computer Science & Engineering, Nagole institute of technology & science", Hyderabad.