iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 3 **Issue:** V **Month of publication:** May 2015

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Software Integrity Attestation for SaaS Cloud Systems Using KL Divergence

J. Thomas Gowtham[1], K.ArunKumar[2]
[1]PG Scholar, Department of Computer Science and Engineering,
[2]Assistant Professor, Department of Computer Science and Engineering,
SNS College of Engineering, Coimbatore-641035, Tamil Nadu, India

Abstract: ***SaaS Cloud systems provide efficient and cost-effective service hosting infrastructure for SaaS service providers. The infrastructures are often shared by multiple users from a variety of security domains, which make them liable to various malicious attacks. In addition, the cloud infrastructure normally hosts applications that deal with important data such as data processing applications. This provides opportunity for attackers to take advantage of the system susceptibility and carry out strategic attacks. There are various software integrity attestations solutions for this problem. This article focuses on a new software integrity attestation solution using KL Divergence. KL divergence The KL-divergence is the robust technique with respect to quantitative similarities identified in the output results of other genuine application service providers.***
Keywords: Integrity Attestation, cloud computing, SaaS, KL Divergence

I. INTRODUCTION

An open distributed SaaS cloud system supporting dataflow processing applications often consists of many domain-specific data processing application service providers. For any data processing application the integrity and accuracy of the results of such applications provide a vital role in determining the quality of such applications. With rapid recognition of the concepts of Software as a Service (SaaS) and Service Oriented Architecture (SOA), the Internet has evolved into a significant service delivery infrastructure instead of only providing host connectivity. The problem is that the attacker can act as a genuine service provider to provide counterfeit service components, and the service components provided by benign service providers may include security holes that can be deflated by attackers. In large scale multitenant cloud systems, many malicious attackers may initiate colluding attacks on particular service functions to invalidate the assumption.

II. RELATED WORK

A. Run Test

RunTest is a scalable runtime integrity attestation scaffold to promise the veracity of dataflow dispensation in cloud infrastructures. It provides light weight application level verification methods with dynamism and authenticate the integrity of data processing results and discover malevolent service providers when conflicting results are detected. It is a light weight application level corroboration scheme that can vigorously substantiate the integrity of data processing outcome in the cloud infrastructure and discover malicious service providers when inconsistent results are spotted. It validates service reliability by combining and analyzing result stability information more readily than evaluating memory footsteps of code execution as used by code verification. This method does not need trusted hardware or protected kernel co-subsisted with conciliator service providers in the cloud. The groundwork behind this approach is that dataflow processing applications are mostly concerned about the accuracy of final data results instead of the integrity of the code execution. Unlike usual agreement-based Byzantine fault discovery schemes, this method does not depend on full time majority voting on all service providers, which is inadequate for cloud infrastructures in terms of scalability. This method is the initial effort to offer effective runtime integrity attestation method for dataflow processing in the SaaS cloud systems. The contributions of Run Test to offer integrity attestation solution are as follows.

1) It provides a novel runtime integrity attestation solution that employs a new attestation graph model to detain corroboration results among various cloud systems. The design relies on a clique based corroboration graph analysis algorithm to recognize malevolent service providers and identify colluding attack models. This scheme can achieve runtime integrity substantiation for cloud dataflow processing systems using a diminutive number of verification data.

2) The RunTest is implemented inside IBM System dataflow processing system and tested on virtual computing lab (VCL), a production virtual cloud infrastructure. The prototype implementation indicates that this scheme can be effortlessly integrated into cloud dataflow processing system.

a) *AdapTest:* AdapTest is a novel adaptive runtime service integrity attestation framework for large scale cloud systems. AdapTest builds on top of our previously developed system RunTest that performs randomized probabilistic attestation and employs a clique-based algorithm to identify malicious nodes. However, randomized attestation still imposes significant overhead for high throughput multi-hop data processing services. In contrast, AdapTest dynamically verifies the trustiness of various services based on previous attestation results and adaptively selects attested services during attestation. Thus, AdapTest can considerably decrease the attestation overhead and cut down the detection delay.

AdapTest makes the following offerings:

i. This model provides a new adaptive multi-hop integrity attestation framework based on a new weighted attestation graph model. We obtain both per-node trust scores and pair-wise trust scores to efficiently guide probabilistic attestation.

ii. AdapTest is implemented on the IBM System S stream processing system and tested it on the virtual computing lab (VCL), a production virtualized computing cluster that operates in a similar way as Amazon EC2. Our experimental results show that AdapTest can considerably decrease attestation overhead for reaching the 100% detection rate by up to 60% and cut down detection time by up to 40% compared to previous randomized attestation approaches.

b) *IntTest:* IntTest provides a novel integrated attestation graph analysis scheme that can provide stronger attacker pinpointing power than previous schemes. Moreover, IntTest can automatically enhance result quality by replacing bad results produced by malicious attackers with good results produced by benign service providers. IntTest can not only pinpoint attackers more efficiently but also can suppress aggressive attackers and limit the scope of the damage caused by colluding attacks. Moreover, IntTest provides result auto correction that can automatically replace corrupted data processing results produced by malicious attackers with good results produced by benign service providers.

B. Attack Model

In a shared cloud infrastructure, malicious attackers can pretend to be legitimate service providers to give fake service instances or compromise vulnerable benign service instances by exploiting their security roles. It focuses on detecting the service integrity attack where a malicious (or compromised) service instance gives deceptive data processing results.

To escape detection, malicious attackers may want to perform selective cheating. That is, they can misbehave on a selective subset of received data while pretending to be benign on other received data. Thus, the attack detection scheme must be able to capture misbehavior that are both unpredictable and occasional without losing scalability. Although we can perform integrity attestation on all service instances all the time, the overhead of integrity attestation would be very high, especially for high throughput data processing services in large-scale cloud systems. Thus, an effective attack detection scheme must perform sneaky attestation, which can prevent attackers from gaining knowledge about our attestation scheme (i.e., when and which set of data will be attested.). Otherwise, the attacker can compromise the integrity of selective data processing results without being detected at all.

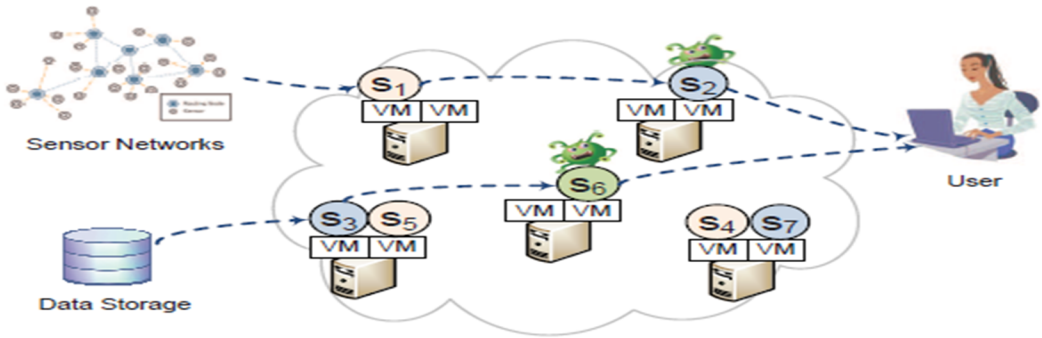


Fig 1. Integrity attack in cloud-based data processing.

Furthermore, cloud computing infrastructures often comprise a large number of hosts running many more VMs and application service instances. It creates new opportunities for colluding attacks where multiple malicious attackers launch coordinated attacks or multiple benign service instances are simultaneously compromised and controlled by a single malicious attacker. Colluders can communicate with each other in an arbitrary way and produce the same incorrect results on the same input. Attackers can also change their attacking and colluding strategies arbitrarily. However, we assume that attackers do not have knowledge of other benign service instances that they do not interact with.

Amongst the three software integrity attestation techniques for multi tenant cloud system, IntTest is the recent solution for the problem. Although both RunTest and AdpaTest have certain advantages, IntTest overcomes the limitations of the two previous approaches. The IntTest is evaluated with an additional metric of false alarm rate hence providing more accurate results than the previous approaches

C. Attestation Solution Using KL Divergence

Although the IntTest is the recent technique for the software integrity in the SaaS cloud system, there are two major drawbacks in this technique. They are, IntTest is inefficient for the SaaS applications that are input deterministic. It is not suitable for the applications which produces results that will vary based on timestamp or different inputs. This drawback can be overcome by using the KL Divergence. The divergence is calculated for all the service functions that are provided by various application service providers. For this some properties of the KL Divergence is used. From following expression KL[f(a),f(b)] ~ div[f(a),f(b)], the divergence between the two service functions is calculated.

1) *Accessibility Matrix Construction:* The first step is to create an accessibility matrix. This will be useful in identifying which Service provider provides which service function. It is done by using a two dimensional array. This array will hold the result values produced by the service functions. These results will then be exposed to the divergence function. The KL method has two arguments, the results of the service functions will be passed as arguments and the divergence is calculated. The construction of the accessibility matrix is done for all the service providers who are providing that particular service function. In our implementation the cloud environment is simulated using CloudSim package. For a set of ten service providers who provide four simple arithmetic service functions the accessibility matrix will be as follows

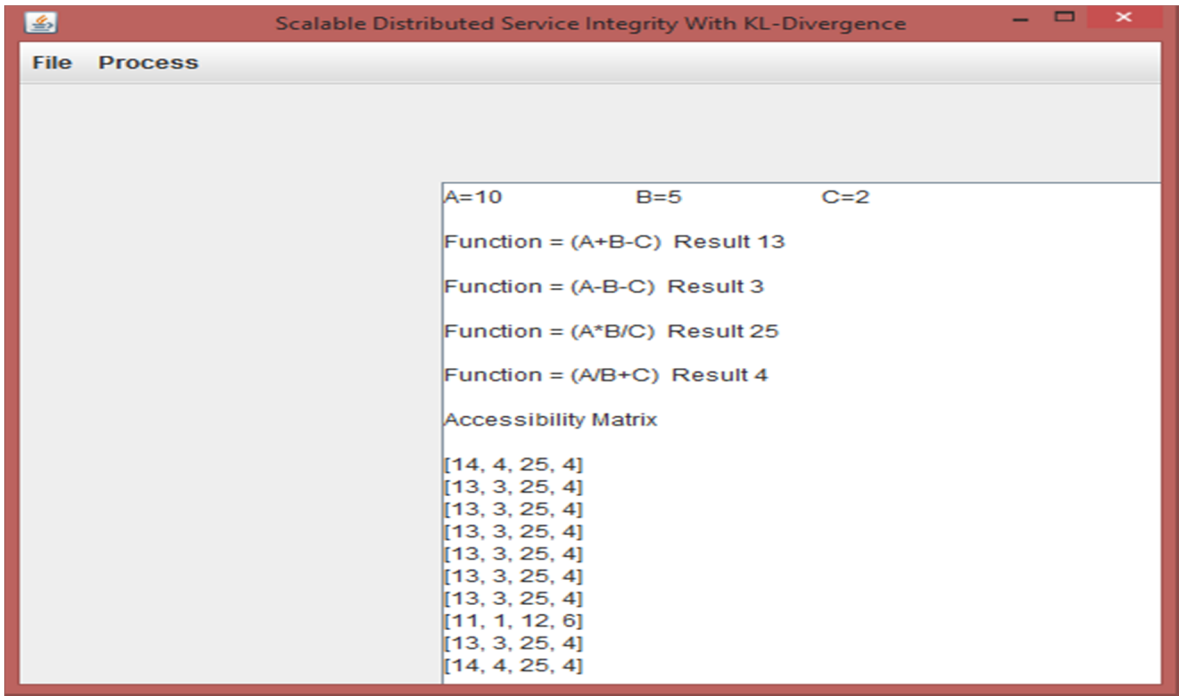


Fig 2. Accessibility Matrix construction

The accessibility matrix holds the results produced by all the ten service providers. These results will be used for calculating the

divergence.

2) *KL Divergence Calculation:* For the above service functions the divergence is calculated. Here the property of KL divergence is used, the property states that the two functions f(a) and f(b) provide zero divergence if f(a)=f(b). Hence for the service functions which provide similar result the divergence will be zero. For the above accessibility matrix the divergence is as follows.

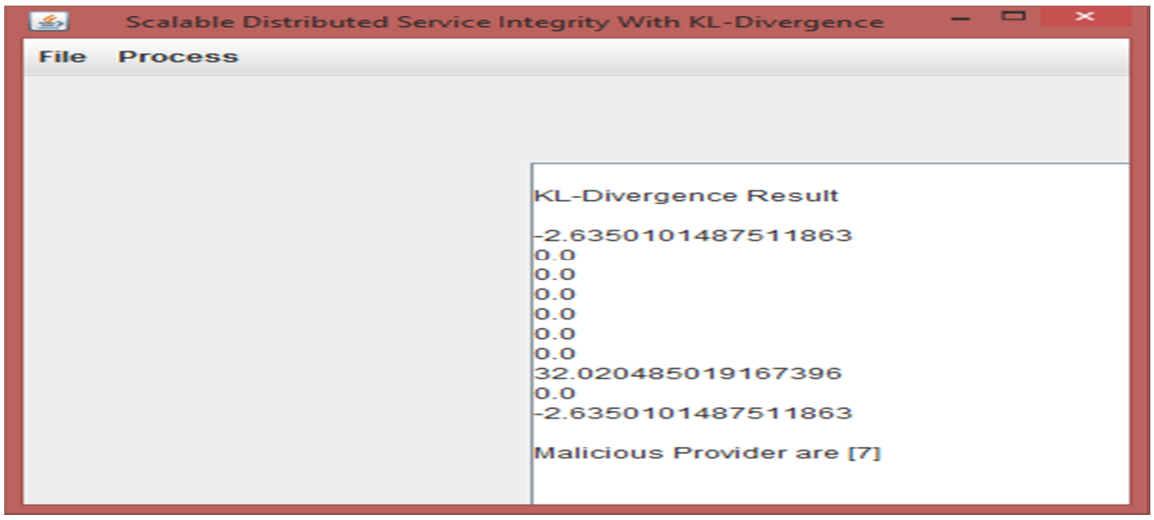


Fig 3. KL Divergence Calculation

The threshold value for the divergence is solely based on the service function. The threshold values must be determined by a standard organization or third party auditors. Here service providers are simulated such a way that some service providers fetch input from an array and the results must be within that permutation. The malicious providers will provide results that are not within those permutations.

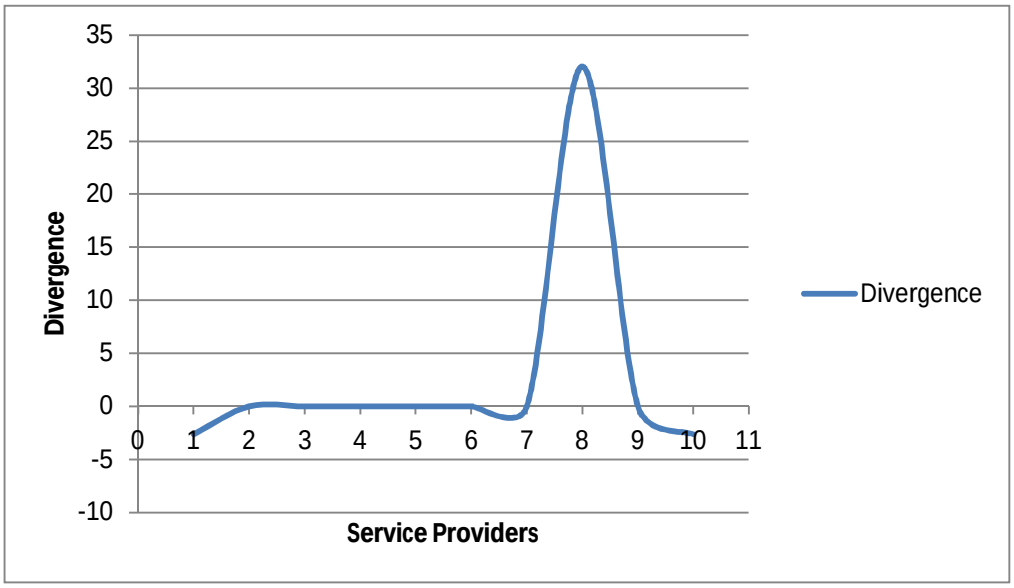


Fig 4. KL Divergence vs Service Providers

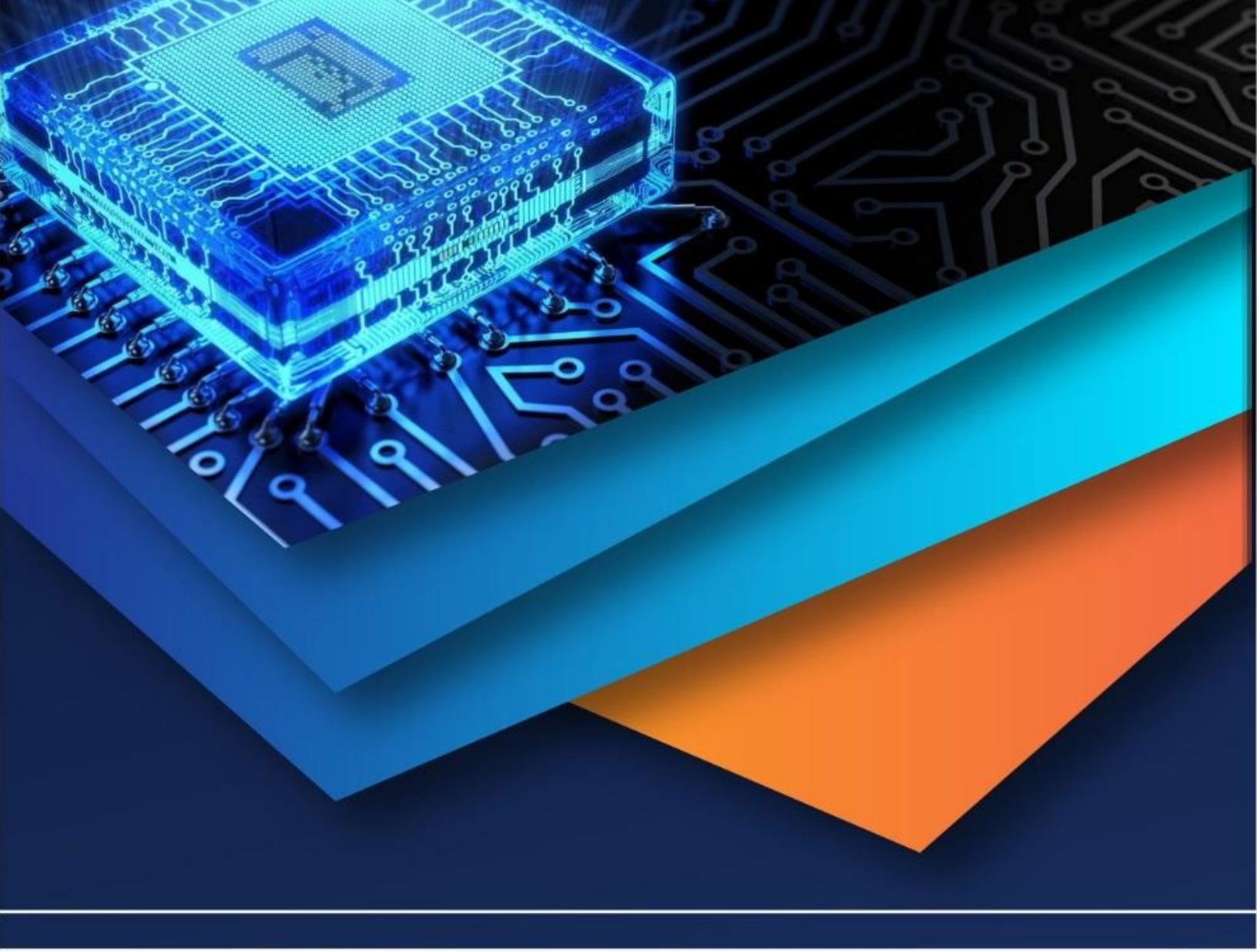
The Service functions that have zero divergence are benign service providers and they could be attested by using the existing IntTest. This KL divergence method can be implemented on top of the existing technique.

III. CONCLUSION

The proposed model of software integrity attestation using KL Divergence can overcome the drawbacks of the existing model of IntTest. Here the SaaS applications that use different inputs such as timestamps and the applications that uses inputs that vary based on time can be attested using this technique. To implement this in real time live servers, one must have the entire details about all the service functions and application service providers. This algorithm must be implemented on a portal node that authenticates the user for accessing the service functions provided by the application service providers.

REFERENCES

[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] Thomas Gowtham J, Arun Kumar K. A Survey on Software Integrity Attestation Solutions for Multi Tenant Cloud Systems. Discovery, 2015, vol. 29 (114),164-167
[5] Software as a Service, http://en.wikipedia.org/wiki/Software as a Service, 2014.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY