Securing Data Integrity and Authentication Service Infrastructure for Service Clouds

SAIKRISHNA AKHANDAM
M.Tech Scholar (CSE) and Department of Computer Science Engineering, Kakinada Institute of Engineering and Technology, Korangi, AP, India.

RAMU VIKRUTI
Assistant Professor, Department of Computer Science and Engineering, Kakinada Institute of Engineering and Technology, Korangi, AP, India.

Abstract: **SaaS Cloud frameworks give proficient and financially savvy service facilitating foundation for SaaS service providers. The frameworks are regularly shared by different clients from an assortment of security areas, which make them obligated to different noxious assaults. Moreover, the cloud foundation ordinarily has applications that manage significant data, for example, data preparing applications. This gives chance to assailants to exploit the framework powerlessness and complete key assaults. In this paper, we present IntTest, a versatile and successful service honesty verification structure for SaaS clouds. IntTest gives a novel incorporated authentication diagram investigation conspire that can give more grounded aggressor pinpointing power than past schemes. In addition, IntTest can consequently improve result quality by supplanting awful outcomes created by noxious assailants with great outcomes delivered by kindhearted service providers. We have actualized a model of the IntTest framework and tried it on a creation cloud computing foundation utilizing IBM System S stream preparing applications. Our exploratory outcomes demonstrate that IntTest can accomplish higher assailant pinpointing precision than existing methodologies. IntTest does not require any exceptional equipment or secure piece support and forces little execution effect to the application, which makes it functional for huge scale cloud frameworks.**

Keywords: **Distributed Service Trustworthiness Verification; Cloud Computing; Secure Appropriated Data Handling;**

I. Introduction

Cloud computing is a model which empowers the service as on interest of the client. Services are conveyed pool of configurable assets specifically server, storage and system .These sorts of services are provisioned and kept up by the different service providers in the system with insignificant undertaking or collaboration. Cloud computing receives the"pay and use"; it diminished the calculation cost in the IT ventures. Cloud services are accessible on interest of the client. Client of cloud just pay according to utilize premise on the usage of the service, the amount they used that quite a bit of sum just they will pay. It is comparable like paying the power charge, how much power is expending client will pay for that as it were. Thusly cloud computing otherwise called the utility computing. Image to speak to the web is cloud from that cloud computing name is roused in light of the fact that in cloud computing everything dealt with over the web. As cloud speak to the accumulation of PCs and servers which is available by the clients over the web. These PCs and servers are overseen by an outsider in different areas. Numerous quantities of working frameworks can be controlled by these machines. Facilitated services are offered through the cloud-computing to its customer by utilizing web. Three cloud computing services are classified as IaaS (Infrastructure-as-a-Service), SaaS (Software-as-a-Service) and PaaS (Platform-as-a-Service). In Infrastructure – as – a - Service, Cloud – service - providers give the virtual server and different services to its customer. Virtual server is arranged by the customers with its gave storage doled out by the service providers case of it is Amazon's EC2. In Platform-as-a-Service, over the web cloud-service-providers gives the adaptability to their customers for structure their application/programming on the providers stage, model is Google App Engine. In Software-as-a-Service, customer of cloud-computing can utilize the different applications. According to require or as long as it is required and pay the sum for that product based on schedule. Case of SaaS application is the Gmail, Salesforce.com and MapQuest.

II. Related work

A structure/framework for confided in storage of a client's data inside the cloud is created. The proposed framework is called A Trusted Storage System for the Cloud‟. As a tremendous amount of electronic data is being produced, there is a necessity of immense storage frameworks which can hold that data. The prerequisite isn't simply putting away the data however putting away it safely, i.e., the confidentiality and uprightness of the data ought to be kept up. The subject of confidentiality and honesty of data comes into the image when the owner‟s data is being put away in outsider storage frameworks like the cloud. National Institute of Standards and Technology (NIST) characterizes cloud computing as pursues: "Cloud computing is a model for empowering advantageous, on-request system access to a mutual pool of configurable computing assets (e.g., systems, servers, storage, applications, and services) that can be quickly provisioned and discharged with negligible administration exertion or service supplier association" . In spite of the fact that cloud computing gives cost effective storage services, it is an outsider service and thusly, a client/customer can't believe the cloud service supplier to store its data safely inside the cloud. Henceforth, numerous associations and clients may not be happy to utilize the cloud services to store their data in the cloud until certain security assurances are made. Constraints of the Current Cloud Computing Stack To persuade the requirement for cloud authentication, we should initially comprehend the dangers that cloud clients cause in the present cloud computing model. A streamlined model of existing cloud services can be spoken to by the outline in Figure 1. Regardless of the assorted variety and multifaceted nature of services and players that populate the cloud biological system, existing cloud services can be assembled by the deliberation layer at which services are conveyed to their individual customers: • Infrastructure-as-a-Service (IaaS) incorporates the fundamental framework services for virtual machine facilitating (e.g., Amazon EC2) and data storage (e.g., Amazon S3). Worked by cloud providers like Amazon and Google these services run legitimately on an equipment framework comprising of topographically scattered datacenters, every one of them facilitating a large number of cloud hubs and other equipment components. The product framework that actualizes IaaS executes on the cloud hubs and comprises of low-level programming parts, including a hypervisor or a working framework for virtual machine facilitating or data storage services. • Platform-as-a-Service (PaaS) sits over the physical framework or IaaS. Thus to IaaS, PaaS joins services for computing and putting away data. Be that as it may, these services are offered at a more elevated amount of reflection (e.g., databases, runtime and web application facilitating) and are bolstered by a more extravagant arrangement of helper services (e.g., message taking care of). Instances of PaaS services incorporate Google AppEngine [10] and Microsoft Azure. PaaS services are ordinarily actualized by middleware segments that work over the working framework and incorporate execution runtimes (e.g., Java), structures, and database servers.

III. Problem Formulation

For a given SaaS system, the goal of IntTest is to pinpoint any malicious service provider that offers an untruthful service function. IntTest treats all service components as black-boxes, which does not require any special hardware or secure kernel support on the cloud platform. We now describe our attack model and our key assumptions as follows Attack model: A malicious attacker can pretend to be a legitimate service provider or take control of vulnerable service providers to provide untruthful service functions. Malicious attackers can be stealthy, which means they can misbehave on a selective subset of input data or service functions while pretending to be benign service providers on other input data or functions.

The stealthy behavior makes detection more challenging due to the following reasons:

1) The detection scheme needs to be hidden from the attackers to prevent attackers from gaining knowledge on the set of data processing results that will be verified and therefore easily escaping detection;

2) The detection scheme needsto be scalable while being able to capture misbehavior that may be both unpredictable and occasional. In a large-scale cloud system, we need to consider colluding attack scenarios where multiple malicious attackers collude or multiple service sites are simultaneously compromised and controlled by a single malicious attacker. Attackers could sporadically collude, which means an attacker can collude with an arbitrary subset of its colluders at any time. We assume that malicious nodes have no knowledge of other nodes except those they interact with directly. However, attackers can communicate with their colluders in an arbitrary way. Attackers can also change their attacking and colluding strategies arbitrarily.

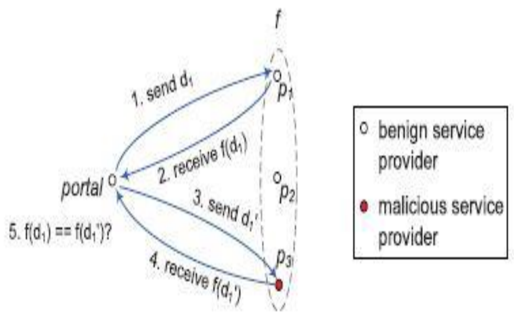

Fig.. Replay-based consistency check

Assumptions:

1. We first assume that the total number of malicious service components is less than the total number of benign ones in the entire cloud system. Without this assumption, it would be very hard, if not totally impossible, for any attack detection scheme to work when comparable ground truth processing results are not available. However, different from RunTest, AdapTest, or any previous majority voting schemes, IntTest does not assume benign service components have to be the majority

for every service function, which will greatly enhance our Pinpointing power and limit the scope of service functions that can be compromised by malicious attackers.

2. Second, we assume that the data processing services are input-deterministic, that is, given the same input, a benign service component always produces the same or similar output (based on a user defined similarity function). Many data stream processing functions fall into this category. We can also easily extend our attestation framework to support stateful data processing services, which however is outside the scope of this paper. Third, we also assume that the result inconsistency caused by hardware or software faults can be marked by fault detection schemes and are excluded from our malicious attack detection.

IV. SaaS Cloud System Model

It develops upon the concepts of Software as a Service (SaaS) and Service Oriented Architecture (SOA) which allows application service providers (ASPs) to deliver their applications via large-scale cloud computing infrastructures. Amazon Web Service (AWS) and Google App Engine are examples to provide a set of application services supporting enterprise applications and big data processing. A distributed application service can be dynamically composed from individual service components provided by different ASPs. For example, a disaster assistance claim processing application consists of voice-over-IP (VoIP) analysis component, email analysis component, community discovery Component, and clustering and joins components. Our work focuses on data processing services which have become increasingly popular with applications in any real world usage domains such as business intelligence, security surveillance, and scientific computing. Each service component, denoted by ci, provides a specific data processing function, denoted by fi, such as sorting, filtering, correlation, or data mining utilities. Each service component can have one or more input ports for receiving input data tuples, denoted by di, and one or more output ports to emit output tuples. In a large-scale SaaS cloud, the same service function can be provided by different ASPs. Those functionally equivalent service components exist because: i) service providers may create replicated service components for load balancing and fault tolerance purposes; and ii) popular services may attract different service providers for profit. To support automatic service composition, we can deploy a set of portal nodes that serve as the gateway for the user to access the composed services in the SaaS cloud. The portal node can aggregate different service components into composite services based on the user's requirements. For security protection, the portal node can perform authentication on users to avoid malicious users from disturbing normal service provisioning. Different from other open distributed systems such as peer-to-peer networks and volunteer computing environments, SaaS cloud systems possess a set of unique features. First, third-party ASPs typically do not want to reveal the internal implementation details of their software services for intellectual property protection. Thus, it is difficult to only rely on challenge-based attestation scheme where the verifier is assumed to have certain knowledge about the software implementation or have access to the software source code. Second, both the cloud infrastructure provider and third-party service providers are autonomous entities. It is impractical to impose any special hardware or secure kernel support on individual service provisioning sites. Third, for privacy protection, only portal nodes have global information about which service functions are provided by which service providers in the SaaS cloud. Neither cloud users nor individual ASPs have the global knowledge about the SaaS cloud such as the number of ASPs and the identifiers of the ASPs offering a specific service function.

V. Cloud Security Challenges

When a company mitigates to intense cloud services, and particularly public cloud services, abundant of the automatic data processing system infrastructure can currently below the management of cloud service supplier. These management initiatives can needs clearly delineating the possession and responsibility roles of each the cloud supplier and therefore the organization functioning within the role of client. Security managers should be able to confirm what detective and preventative controls exist to obviously outline security posture of the organization. Though correct security controls should be implement supported quality, threat, and vulnerability risk assessment matrices. Encryption: the sensitivity of information might need that the network traffic to and from the virtual machine be encrypted, victimization encoding at the host OS computer code.

• Physical security: keep the virtual system and cloud management hosts safe and secure behind carded doors, and environmentally safe.

• Authentication and access control: the authentication capabilities among your virtual system ought to copy the approach your different physical systems evidence. Only once watchword and life science ought to all be enforced within the same manner. Conjointly authentication needs whereas you're causing information or message from one cloud to different cloud. To produce message authentication we'll use digital signatures.

• Separation of duties: as system get additional complicated, misconfiguration occur, as a result of lack of experience in addition to meager communication. Take care to enforce least privileges with access controls and responsibleness.

• Configuration, modification management, and patch management: this is often important and typically unnoted in smaller organizations. Configuration, modification management, patch management, and updated processes ought to be maintained within the virtual world moreover as physical world.

• Intrusion detection and prevention: what's returning into and going out of your network must recognize. a bunch primarily based} intrusion bar system in addition to a hypervisor based resolution may examine for virtual network traffic.

VI. 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 are 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. shows the overall architecture of the proposed system. In this the user give request to cloud the serve will be deployed in the cloud 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.

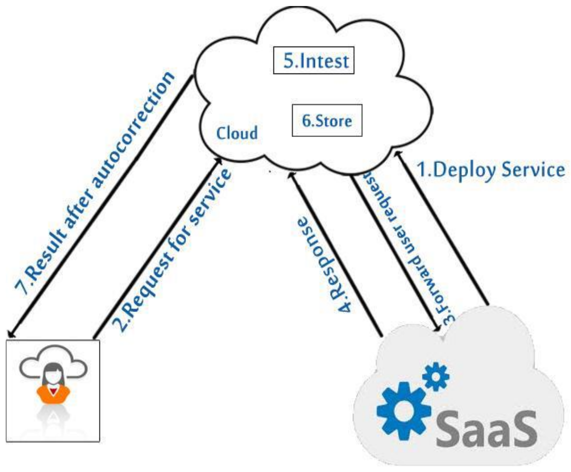

Figure: Over all architecture of the proposed method

Signature verification algorithm

For receiver to authenticate senders signature, he must have a copy of her public-key curve point Q_A. Receiver can verify Q_A is a valid curve point as follows:

1. Check that Q_A is not equal to the identity element O, and its coordinates are otherwise valid
2. Check that Q_A lies on the curve
3. Check that Q_A

   $n \times Q_A = O$

1. Verify that r and s are integers in $[1, n-1]$. If not the signature is invalid.
2. Calculate $e = HASH(m)$, where HASH is the same function used in the signature generation.
3. Let z be the L_n left most bits of e
4. Calculate $w = s^{-1} \bmod n$
5. Calculate $u_1 = zw \bmod n$ and $u_2 = rw \bmod n$
6. Calculate the curve point $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$.
7. The signature is valid if $r \equiv x_1 (\bmod n)$, invalid otherwise.

VII. Conclusion

This paper, discussed about various approaches and techniques used in providing the service integrity of SaaS cloud model. Each techniques has its own advantages and dis-advantages. Most integrity attacks can be effectively destroyed by the advanced techniques and approaches. All methods are approximate to our goal of providing the service or search results with integrity, we need to further perfect those approaches or develop some efficient methods.

FUTURE WORK

In the future, allow verifying functional properties of cloud services, they have not yet matured. Multiple recent works have tackled specific concerns that arise in the context of cloud storage, and promising techniques have emerged.

References

[1] Garay.J and Huelsbergen.L, “Software integrity protection using timed executable agents,” in Proceedings of ACM Symposium on Information, Computer and Communications Security (ASIACCS), Taiwan, Mar. 2006.

[2] Juan Du Daniel J. Dean, Yongmin Tan, XiaohuiGu, Senior and Ting Yu Scalable Distributed Service Integrity Attestation for Softwareas-a-Service Clouds .

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

[4] Du.J, Shah.N, and Gu.X, “Adaptive Data-Driven Service Integrity Attestation for Multi-Tenant Cloud Systems,” Proc. Int’l Workshop Quality of Service (IWQoS), 2011. Virtual Computing Lab, http://vcl.ncsu.edu/, 2013.

[5] Ho etal.T, “Byzantine Modification Detection in Multicast Networks Using Randomized Network Coding,” Proc. IEEE Int’l Symp. Information Theory (ISIT), 2004.

[6] Hwang.I, “A Survey of Fault Detection, Isolation, and Reconfiguration Methods,” IEEE Trans. Control System Technology, vol. 18,no. 3, pp. 636-653, May 2010.

[7] Lamport.L, Shostak.R, and Pease.M, “The Byzantine Generals Problem,” ACM Trans. Programming Languages and Systems, vol. 4,no. 3, pp. 382-401, 1982

[8] Shi.E, Perrig.A, and Doorn.L.V, “Bind: A fine-grained attestation service for secure distributed systems,” in Proceedings of the IEEE Symposium on Security and Privacy, 2005.

[9] Xu.W, Venkatakrishnan.V. N, Sekar.R, and Ramakrishnan .I. V, “A framework for building privacy-conscious composite web services,” in IEEE International Conference on Web Services, Chicago, IL, Sep. 2006, pp. 655–662.

[10] Zhang.H, Savoie.M,Campbell.S, Figuerola.S, von Bochmann.G, and Arnaud.B.S, “Service-oriented virtual private networks for grid applications,” in IEEE International Conference on Web Services, Salt Lake City, UT, Jul. 2007, pp. 944–951.

Authors Profile

Mr.SaiKrishna Akhandam is a studying M.Tech in Kakinada Institute of Engineering and Technology, Korangi, East Godavari, A.P. He Received his B.Tech in Electronics and Communications Engineering from AGCET, Tadepalli Gudem.

Mr.RamuVikruti is an asst. professor in Computer Science at Kakinada Institute of Engineering & Technology Korangi, KKD, Affiliated to JNTU Kakinada. He holds an M.Tech degree in Computer Science.