ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Scalable Distributed Service Integrity Attestation for Software-as-a-Service Clouds

Y.Navajyothi
M.Tech Student,
Dept of CSE,
KITS for Women's, Kodad, T.S, India.

N.Lakshmi Priya
Associate Professor,
Dept of CSE,
KITS for Women's, Kodad, T.S, India.

Abstract:

Software-as-a-service (SaaS) cloud systems allow application service providers to deliver their applications via massive cloud computing infrastructures. SaaS is becoming an increasingly prevalent delivery model as underlying technologies that support Web services and service-oriented architecture (SOA) mature and new developmental approaches, such as Ajax, become popular. SaaS clouds are vulnerable to malicious attacks because of their sharing nature. Many security frameworks have been developed toaddress cloud security issues like IntTest, Privacy Proxy, Trusted virtual data center,Placement and Extraction method for Exploring Information Leakage, Stateful Dataflow Processing, Building Privacy-Conscious Composite Web Services, Anomaly Extraction and Mitigation using Efficient- Web Miner Algorithm.Breif Study on the above frame worksare explained below.

Keywords:

Distributed Service, Integrity attestation, Cloud computing, Multitenanat\.

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 a 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 multi-tenant 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 IntTest, 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.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

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.

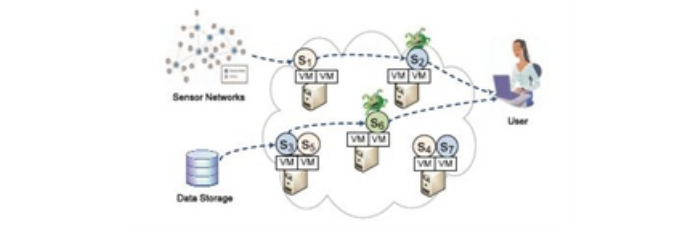

Figure 2: Service integrity attacks in clouds

TABLE 1
Notations

notation	meaning
p_i	service provider
f_i	service function
c_i	service component
d_i	application data tuple
P_a	attestation probability
r	number of copies for a tuple
K	Max number of malicious service providers
C_G	minimum vertex cover of graph G
N_p	the neighbor set of node p
G_r^c	the residual graph of G
Ω	the set of malicious service providers identified by the global inconsistency graph
M_i	the set of malicious service providers identified by consistency graph in service function f_i

2 PRELIMINARY:

In this section, we first introduce the software-as-a-service cloud system model. We then describe our problem formulation including the service integrity attack model and our key assumptions. Table 1 summarizes all the notations used in this paper.

2.1 SaaS Cloud System Model:

SaaS cloud builds upon the concepts of software as a service and service-oriented architecture which allows application service providers to deliver their applications via large-scale cloud computing infrastructures. For example, both Amazon Web Service and Google AppEngine 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 (pi) For example, a disaster assistance claim processing application consists of voice-over-IP (VoIP) analysis component, e-mail analysis component, community discovery component, and clustering and join components. Our work focuses on data processing services which have become increasingly popular with applications in many real-world usage domains such as business intelligence, security surveillance, and scientific computing.

2.2 Problem Formulation:

Given an SaaS cloud 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.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

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..

3 DESIGN AND ALGORITHMS:

In this section, we first present the basis of the IntTest system: probabilistic replay-based consistency check and the integrity attestation graph model. We then describe the integrated service integrity attestation scheme in detail. Next, we present the result autocorrection scheme.

3.1 Baseline Attestation Scheme:

To detect service integrity attack and pinpoint malicious service providers, our algorithm relies on replay-based consistency check to derive the consistency/inconsistency relationships between service providers.

3.2 Integrated Attestation Scheme:

We now present our integrated attestation graph analysis algorithm.

Step 1: Consistency graph analysis. We first examine perfunction consistency graphs to pinpoint suspicious service providers. The consistency links in per-function consistency graphs can tell which set of service providers keep consistent with each other on a specific service function. Given any service function, since benign service providers always keep consistent with each other, benign service providers will form a clique in terms of consistency links.

4. RELATED WORK

A. Run Test:

RunTest is a scalable runtime integrity attestationframework to guarantee the integrity of dataflow processing in cloud infrastructures. It provides light-weight application level attestation methods to dynamically authenticate the integrity of data processing results and identify malicious service providers when inconsistent results are detected. It is a light weight application level attestation scheme that can dynamically confirm the integrity of data processing outcome in the cloud infrastructure and discover malicious service providers when inconsistent results are spoted.

It validates service integrity by combining and analyzing result consistency information more willingly than evaluating memory footprints of code execution as used by code attestation. This approach does not need trusted hardware or secure kernel co-subsisted with intermediary service providers in the cloud. The foundation behind this approach is that dataflow processing applications are mostly apprehensive about the accuracy of final data results instead of the integrity of the code execution. Unlike customary agreement-based Byzantine fault detection schemes, this approach does not rely on full time majority voting on all service nodes, which falls short for cloud infrastructures in terms of scalability. This work makes the first effort to offer efficient runtime integrity attestation method for dataflow processing in the cloud infrastructure. Run Test makes the following contribution:

•It provides a new runtime service integrity attestation method that employs a novel attestationgraph model to capture attestation results amongstdifferent cloud nodes. The design is a clique basedattestation graph analysis algorithm to identify malicious service providers and recognize colluding attack models. Our scheme can attain runtime integrity confirmation for cloud dataflowprocessing services using a small number of attestation data.

•The RunTest is implemented within IBM Systemdataflow processing system and tested it on NCSU virtual computing lab (VCL), a production virtual cloud infrastructure. The prototype implementation indicates that our scheme can be effortlessly integrated into cloud dataflow processing system.

B. 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.

5.SECURITY ANALYSIS:

We now present a summary of the results of our analytical study about IntTest. Additional details along with a proof of the proposition presented in this section can be found in Section 2 of the online supplemental material. Proposition 3. Given an accurate upper bound of the number of malicious service providers K, if malicious service providers always collude together, IntTest has zero false positive.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Although our algorithm cannot guarantee zero false positives when there are multiple independent colluding groups, it will be difficult for attackers to escape our detection with multiple independent colluding groups since attackers will have inconsistency links not only with benign nodes but also with other groups of malicious nodes. Additionally, our approach limits the damage colluding attackers can cause if they can evade detection in two ways. First, our algorithm limits the number of functions which can be simultaneously attacked. Second, our approach ensures a single attacker cannot participate in compromising an unlimited number of service functions without being detected.

6 EXPERIMENTAL EVALUATION:

In this section, we present the experimental evaluation of the IntTest system. We first describe our experimental setup. We then present and analyze the experimental results.

6.1 Experiment Setup:

We have implemented a prototype of the IntTest system and tested it using the NCSU's virtual computing lab , a production cloud infrastructure operating in a similar way as Amazon EC2 . We add portal nodes into VCL and deploy IBM System S stream processing middleware to provide distributed data stream processing service. System S is an industry-strength high performance stream-processing platform that can analyze massive volumes of continuous data streams and scale to hundreds of processing elements (PEs) for each application. In our experiments, we used 10 VCL nodes which run 64bit CentOS 5.2. Each node runs multiple virtual machines (VMs) on top of Xen 3.0.3. The data-flow processing application we use in our experiments is adapted from the sample applications provided by System S. This application takes stock information as input, performs windowed aggregation on the input stream according to the specified company name,and then performs calculations on the stock data. We use a trusted portal node to accept the input stream, perform comprehensive integrity attestation on the PEs, and analyze the attestation results.

6.2 Results and Analysis:

We first investigate the accuracy of our scheme inpinpointing malicious service providers.

Fig. 8a compares our scheme with the other alternative schemes (i.e., FTMV, PTMV, and RunTest) when malicious service providers aggressively attack different number of service functions. In this set of experiments, we have 10 service functions and 30 service providers. The number of service providers in traverses good nodes. If it is true, we will use the result of the attestation data to replace.

7 LIMITATION DISCUSSION:

Although we have shown that IntTest can achieve better scalability and higher detection accuracy than existing\ schemes, IntTest still has a set of limitations that require further study. A detailed limitation discussion can be found in Section 4 of the online supplementary material. We now provide a summary of the limitations of our approach. First, malicious attackers can still escape the detection if they only attack a few service functions, take majority in all the compromised service functions, and have less inconsistency links than benign service providers.

However, IntTest can effectively limit the attack scope and make it difficult to attack popular service functions. Second, IntTest needs to assume the attested services are input deterministic where benign services will return the same or similar results defined by a distance function for the same input. Thus, IntTest cannot support those service functions whose results vary significantly based on some random numbers or time stamps.

8. CONCLUSION:

In this paper, we have presented the design and implementation of IntTest, a novel integrated service integrity attestation framework for multitenant software-as-a-service cloud systems. IntTest employs randomized replay-based consistency check to verify the integrity of distributed service components without imposing high overhead to the cloud infrastructure. IntTest performs integrated analysis over both consistency and inconsistency attestation graphs to pinpoint colluding attackers more efficiently than existing techniques. Furthermore, IntTest provides result autocorrection to automatically correct compromised results to improve the result quality. We have implemented IntTest and tested it on a commercial data stream processing platform running inside a production virtualized cloud computing infrastructure.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Our experimental results show that IntTest can achieve higher pinpointing accuracy than existing alternative schemes. IntTest is lightweight, which imposes low-performance impact to the data processing services running inside the cloud computing infrastructure.

9.ACKNOWLEDGMENTS:

I am y.navajyothi. and would like to thank the publishers, researchers for making their resources material available. I am greatly thankful to Associate Prof N.Lakshmi Priya for their guidance. We also thank the college authorities, PG coordinator and Principal for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to friends and family members
.

10.REFERENCES:

[1] Amazon Web Services, http://aws.amazon.com/, 2013.

[2] Google App Engine, http://code.google.com/appengine/, 2013.

[3] Software as a Service, http://en.wikipedia.org/wiki/Software asa Service, 2013.

[4] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, Web ServicesConcepts, Architectures and Applications (Data-Centric Systems andApplications). Addison-Wesley Professional, 2002.

[5] T. Erl, Service-Oriented Architecture (SOA): Concepts, Technology, andDesign. Prentice Hall, 2005.

[6] T.S. Group, "STREAM: The Stanford Stream Data Manager," IEEEData Eng. Bull., vol. 26, no. 1, pp. 19-26, Mar. 2003.

[7] D.J. Abadi et al., "The Design of the Borealis Stream Processing Engine," Proc. Second Biennial Conf. Innovative Data Systems Research (CIDR '05), 2005.

[8] B. Gedik et al., "SPADE: The System S Declarative StreamProcessing Engine," Proc. ACM SIGMOD Int'l Conf. Managementof Data (SIGMOD '08), Apr. 2008.

[9] S. Berger et al., "TVDc: Managing Security in the Trusted VirtualDatacenter," ACM SIGOPS Operating Systems Rev., vol. 42, no. 1,pp. 40-47, 2008.

[10] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, YouGet Off My Cloud! Exploring Information Leakage in Third-PartyCompute Clouds," Proc. 16th ACM Conf. Computer and CommunicationsSecurity (CCS), 2009.

Author's Details:

Miss y.navajyothi. MTech student, in M.Tech Student, Dept of CSE in KITS for women's,kodad, T.S, India

N.Lakshmi Priya working as a Associate at CSE in KITS for women's,kodad, T.S, IndiaJNTUH Hyderabad. He has 2 years of UG/PG Teaching Experience