SCALABLE SERVICES DISTRIBUTED TO SOFTWARE-AS-A-SERVICE CLOUDS

[#1]**Ms.KOTHAPALLY HARINI PRIYA,** *Assistant Professor*

[#2]**Mr.PURAM SRINIVAS,** *Assistant Professor*

Department of Computer Science and Engineering,

SREE CHAITANYA INSTITUTE OF TECHNOLOGICAL SCIENCES, KARIMNAGAR, TS.

ABSTRACT: Application service providers can now offer their programs on enormous computer networks in the cloud thanks to software-as-a-service (SaaS) cloud platforms. Because SaaS clouds are shared, they can be attacked by malicious actors. In this paper, we will look at the Int Test. It is a scalable and successful technique for demonstrating the trustworthiness of SaaS cloud services. Int Test's new integrated attestation graph analysis method, when compared to prior techniques, provides for more exact identification of attackers. Furthermore, Int Test can rapidly improve the quality of results by replacing bad results from hostile attackers with good results from reliable service providers. The IBM System S stream processing applications were used to develop and evaluate a variant of the Int evaluate system in a real-world cloud computing environment. Our findings show that Int Test can pinpoint an attacker with more accuracy than existing methods. Int Test can be used in large-scale cloud systems because it does not require specialist hardware or secure kernel support and does not dramatically slow down the software.

***Index Terms*:** Distributed service integrity attestation, cloud computing, secure distributed data processing.

1. INTRODUCTION:

Cloud computing is a new, low-cost, and advantageous means of renting computing resources. It removes the requirement for consumers to operate their own complicated physical computer infrastructures. SaaS clouds, such as Amazon Web Service (AWS) and Google App Engine, enable application service providers (ASPs) to offer their applications across the vast cloud computing infrastructure. The basic concepts for these clouds are SOA and software as a service (SaaS).

Our work primarily focuses on data stream processing services, which are viewed as a type of "killer" cloud application that may be used in a number of real-world scenarios such as security surveillance, scientific computing, and business intelligence. ASPs from a number of security disciplines, on the other hand, are vulnerable to attack because they commonly leverage cloud computing infrastructures. Assailants, for example, can act as reliable service providers in order to sell counterfeit service components. These phony service components may have security weaknesses that attackers can use against legitimate service providers. Our study primarily focuses on service integrity attacks that result in inaccurate data processing results for customers. problems around service integrity authentication have not been effectively addressed, but privacy and confidentiality problems have previously been thoroughly researched. Whether the cloud system is handling public or private data, the most critical issue that must be addressed is service security. A few approaches for verifying software integrity have been proposed in the past, but they require either trusted hardware or secure kernel support, making their deployment on large cloud computing systems difficult. All replicas in normal Byzantine fault tolerance (BFT) systems use full-time majority voting (FTMV). Although this detects random faults, it severely slows down the cloud system. Section 5 of the online supplement is accessible in the Computer Society

Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2013.62. It provides a thorough summary of the linked work. Int Test is a unique tool that may be used to validate the integrity of multitenant cloud systems. Within Int Test, there is a handy way for testing the stability of a service that does not involve application updates or the assumption of trustworthy groups on third-party service provisioning sites. Int Test is a continuation of our previous work, Run Test and A dap Test, but it is more effective at identifying malevolent users.

It is critical to believe that the bulk of service functions are managed by nice service providers for Run Text and A dap Test, as well as typical majority voting systems. Multiple hostile actors may combine to attack individual service components in large, multitenant cloud systems to demonstrate that the premise is untrue. In order to resolve the problem, Int Test investigates all of the consistent and inconsistent techniques through which different cloud service providers communicate with one another. For each function, the Int Test evaluates both the generic inconsistency graph and the consistency graph. The global inconsistency graph can be used to identify attackers who are attempting to compromise several service functions. The per-function consistency graph analysis, on the other hand, can restrict the damage caused by collaborating assailants. This suggests that Int Test may still be able to detect hostile attackers, even if they are in the majority for specific service activities.

2. RELATED WORK:

Numerous novel integrity assurance strategies have been developed in recent years for clouds that provide software as a service. For example, the BIND method, the A dap Test method, the Run Test method, and so on all have flaws. Some of them necessitate dependable kernel support as well as specialized hardware components. The BIND (Binding Information and Data) method is used to validate the integrity services of a software as a service cloud system. It was an authenticating device capable of providing proof from a third party or a secure kernel. The steps in this technique are as follows:

- A method of incorporating evidence and annotation
- How the action works
- For verification, a hash-based authenticator is used.

To assure data accuracy, the BIND approach leverages the Diffee-Hellman key exchange. Another mechanism presently utilized to ensure the dependability of cloud computing systems is the Timed Execution Agent System (TEAS). An agent creation and testing technique is used in this TEAS method. Another contemporary solution is the run test, which is a framework for scalable runtime integrity assurance. It offers a straightforward approach of application-level certification to ensure the dependability of cloud flow processing. This will identify the company that provides bad data processing services, show instances of dishonest data flow processing, and then identify an intruder's activity. This Run Test will produce a list of trustworthy service providers and demonstrate how bad actors act.

It's unfortunate that it doesn't work well. The A dap Test, a novel approach for validating that runtime services remain reliable in the face of changing data, is now in use. This strategy will shorten the time and lower the cost of attesting. Its components are all considered black boxes, and it does not require any specialist hardware or software. This A dap Test detects more malicious attackers and service providers while requiring less work and attestation than earlier methods.

All of the aforementioned strategies, which are used in published publications, have drawbacks. And the INT Test is utilized to get around these problems. Using this Int Test will also help you differentiate between service providers and malicious attackers with more consistency and precision. It will also contain a way for automatic result rectification, which will convert poor results into good ones without the need for specialist hardware or safe kernel support.

EXISTING SYSTEM:

App service providers (ASPs) would be allowed to run their applications on the huge cloud computing infrastructure. Our work primarily focuses on data stream processing services, which are viewed as a type of "killer" cloud application that may be used in a number of real-world scenarios such as security surveillance, scientific computing, and business intelligence. ASPs from a number of security disciplines, on the other hand, are vulnerable to attack because they commonly leverage cloud computing infrastructures. Assailants, for example, can act as reliable service providers in order to sell counterfeit service components. These phony service components may have security weaknesses that attackers can use against legitimate service providers.

DISADVANTAGES OF EXISTING SYSTEM:

- These approaches typically require secure kernel support or reliable hardware to function.
- As a result, configuring them on very big cloud computer systems is complicated.

3. PROPOSED SYSTEM:

Int Test, a new approach for ensuring the integrity of cloud services for numerous users, is discussed in this study. Within Int Test, there is a handy way for testing the stability of a service that does not involve application updates or the assumption of trustworthy groups on third-party service provisioning sites. Int Test is a continuation of our previous work, Run Test and A dap Test, but it is more effective at identifying malevolent users. It is critical to believe that the bulk of service functions are managed by nice service providers for Run Text and A dap Test, as well as typical majority voting systems. Multiple hostile actors may combine to attack individual service components in large, multitenant cloud systems to demonstrate that the premise is untrue. In order to resolve the problem, Int Test investigates all of the consistent and inconsistent techniques through which different cloud service providers communicate with one another. Int Test looks at the generic consistency graph as well as the consistency graph for each function.

4. SAAS CLOUD SYSTEM MODEL:

The software as a service (SaaS) cloud is built on the notions of service-oriented architecture by allowing application service providers to distribute their programs across enormous cloud computing networks. Google App Engine and Amazon Web Service, for example, both provide a set of application services that allow the execution of enterprise applications as well as the management of vast amounts of data. By mixing on-demand service components from many ASPs, a distributed application service (pi) can be created. Program components used to handle disaster aid claims include voice-over-IP (VoIP) analysis, email analysis, community discovery, and clustering and joining.

Our primary focus is on data processing services, which are becoming increasingly popular and have a variety of real-world applications, such as scientific computing, business intelligence, and security surveillance (www.jreecs.com). Ci and fi are two distinct service components. Each one provides a distinct method of processing data, such as sorting, filtering, correlating, or data mining tools. Each service component, denoted by di, may have one or more output ports for transmitting output tuples and one or more input ports for receiving input data tuples. Several ASPs may offer the same service function in a large SaaS cloud. These parts of the service are nearly identical because service providers may build redundant server components for fault tolerance and load balancing; and many people may offer and profit from common services.

To facilitate the establishment of autonomous services, a network of portal nodes has been developed. This is how the integrated SaaS cloud services are accessed by the user. The portal node can integrate numerous service components into composite services based on the user's preferences. Table 1 illustrates a security compromise in a cloud-based service. VM stands for virtual machines, and Si stands for numerous service components. The portal node can utilize user authentication as a security precaution to prevent malicious users from tampering with standard service configuration. Volunteer computing environments and peer-to-peer networks distinguish SaaS cloud platforms from other open distributed systems. First, third-party ASPs often do not want to divulge the inner workings of how their software services are deployed in order to protect their intellectual property.

As a result, when the verifier is expected to understand how the program works or to have access to the software's source code, challenge-based attestation solutions are not necessarily reliable. The cloud infrastructure provider and third-party service providers are separate companies. It is not practical to require a specific hardware component or secure kernel support for every service provisioning point. Third, to protect privacy, only portal nodes can access global data identifying which SaaS cloud service providers are in charge of specific service duties. Cloud consumers and individual ASPs are both uninformed of the number of ASPs or their distinct identities among those who provide a certain service function.

ADVANTAGES OF PROPOSED SYSTEM:

- ➢ Framework for ensuring the integrity of distributed services in big cloud computing infrastructures that is both effective and scalable.
- ➢ This is a brand-new integrated service integrity attestation mechanism that finds problems with more accuracy than previous methods.
- ➢ Restoring results that have been tampered with by malicious attackers automatically.
- ➢ Analytical research and experimental evaluation to identify the precision and time requirements of the integrated service integrity attestation method.

5. DESIGN AND ALGORITHMS:

This section will begin by going through the basic components of the Int Test system. Examples include the probabilistic replay-based consistency check and the integrated service integrity attestation approach. The auto-correction method used by the result will then be displayed.

Baseline Attestation Scheme:

To evaluate the linkages between service providers' consistency and inconsistency, we use a consistency check based on replays. This aids in identifying malicious service providers and preventing service integrity attacks. Figure shows the technique for comparing three service providers (p1, p2, and p3) that supply the same service (f). The gateway gets the output, fd1, after transferring the initial data item, d1, to p. The gateway then sends d01, a duplicate of d1, to p3 and receives fd01 in response.

Our method is based on the idea that if two service providers differ on what should happen with the same input, one of them must be bad. The link then checks fd1 and fd01 to see if p1 and p3 are the same. Please keep in mind that we do not send two copies of the same input data item (attestation data) at the same time. Rather, once the processing results for the original data are available, we replay the authentication data on other service providers. As a result, malicious people who modify the original data risk being found. Regardless of the potential of a delay, a

Fig. Use replays to establish consistency.

The next tuples in the data stream might be processed while attestation is still in progress to hide the attestation delay from the user. A consistency relationship exists between two service providers if they consistently produce same output outcomes for all input data. Their relationship is uneven if they respond differently to at least one input. Because two outstanding service providers may supply us with similar but not identical outcomes, we do not limit the consistency relationship to the equality function. For example, if credit ratings are acquired from different credit companies, they may not be similar for the same person. Allow the user to choose a distance function to establish the maximum allowable difference between results.

Integrated Attestation Scheme:

We shall now go over our full attestation graph analysis technique. First, look at the homogeneity graph. To begin, we employ per function consistency plots to identify suspect service providers. The consistency linkages in per-function consistency graphs can be used to determine which service providers are consistent with one another in terms of a specific service function. Certain service providers will always be consistent with one another, forming a consistency connection clique. In particular, see Fig. 3a, P1, P3, and P4 are always together as providers of goodwill services. In prior work, we created a clique-based method to detect dishonest service providers. Assuming that there are more benign than malicious service providers, a benign node will always remain in a clique of solely benign nodes. This coterie has a greater number of members than bk=2c, where k is the number of service providers executing the service function. As a result, we may find odd nodes by looking for those that are not members of any cliques greater than bk=2c and are not members of the clique of size 3.

6. RESULTS AND ANALYSIS:

To begin, we must verify that our strategy appropriately identifies substandard service providers. When malicious service providers actively target numerous service functions, Fig. 8a compares our system to the other alternatives (FTMV, PTMV, and Run Test). This collection of exams includes 30 service providers and 10 service tasks. Each service role has between one and eight service providers. Each type of service provider provides two randomly selected service functions. The stream receives 300 tuples of information every second. We have classified 20% of service providers as untrustworthy. When the gateway receives the processing result of a new data tuple, it decides whether to execute data attestation at random. The attestation data is duplicated twice (r14 3 total number of data copies, including the original data), and each tuple has a 0.2% probability of being testified (Pu 14 0:2). Each trial is run three times. The average detection and false alarm rates for various devices are provided. Keep in mind that once the randomized probabilistic attestation embraces all attested service providers and determines the majority group, Run Test can achieve the same detection accuracy as approaches based on majority vote.

Before making a decision, Int Test thoroughly checks both the global inconsistency graph and the per-function consistency graph. Int Test, on the other hand, has a substantially greater recognition rate and a reduced false-positive rate. Furthermore, when malicious service providers attack many processes, Int Test can produce more precise results. Furthermore, keep in mind that even if malicious service providers only target a subset of service operations, our system can still detect them if they launch an aggressive attack.

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

Figure 1

Scalable Distributed Service Integrity Attestation for Software-as-a-Service Clouds
USER INBOX :

Figure 2

7. CONCLUSION

They created and deployed Int Test, a cutting-edge technique for assuring that multi-tenant cloud systems deliver integrated service integrity assurance. Int Test uses randomized replay-based consistency testing to ensure that distributed service components are correct without unnecessarily complicating the cloud infrastructure. Int Test uses an integrated analysis of attestation graphs to check for continuity and irregularity in order to more precisely identify attackers who are collaborating than existing methods.

Furthermore, Int Test offers a "result auto correction" option that can correct inaccurate results and improve the quality of the results. Int Test was configured and tested in a live virtualized cloud computing infrastructure utilizing a commercial data stream processing platform. This experiment shows that Int Test performs more precisely than other native techniques currently in use. Int Test has no effect on the processing performance of cloud computing data processing services due to its small size and light weight.

REFERENCES:

1. Amazon Web Services, http://aws.amazon.com/, 2013.
2. Google App Engine, http://code.google.com/ app engine/, 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 Services Concepts, Architectures and Applications (Data Centric Systems and Applications). Addison-Wesley Professional, 2002.
5. T. Erl, Service-Oriented Architecture (SOA): Concepts, Technology, and Design. Prentice Hall, 2005.
6. T.S. Group, "STREAM: The Stanford Stream Data Manager," IEEE Data 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 Stream Processing Engine," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '08), Apr. 2008.
9. S. Berger et al., "TV Dc: Managing Security in the Trusted Virtual Datacenter," ACM SIGOPS Operating Systems Rev., vol. 42, no. 1,pp. 40-47, 2008.
10. T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, You Get Off My Cloud! Exploring Information Leakage in Third-Party Compute Clouds," Proc. 16th ACM Conf. Computer and Communications- Security (CCS), 2009.