Security Attacks and Solutions in Clouds

Kazi Zunnurhain and Susan V. Vrbsky
Department of Computer Science
The University of Alabama
Tuscaloosa, AL 35487-0290
kzunnurhain@crimson.ua.edu, vrbsky@cs.ua.edu

Abstract— **Cloud computing offers great potential to improve productivity and reduce costs, but at the same time it possesses many new security risks. In this paper we identify the possible security attacks on clouds including: Wrapping attacks, Malware-Injection attacks, Flooding attacks, Browser attacks, and also Accountability checking problems. We identify the root causes of these attacks and propose specific solutions.**

Keywords- FAT table; hypervisor; security attack

I. INTRODUCTION

In the field of computation, there have been many approaches for enhancing the parallelism and distribution of resources for the advancement and acceleration of data utilization. data clusters, distributed database management systems, data grids, and many more mechanisms have been introduced. Now cloud computing is emerging as the mechanism for high level computation, as well as serving as a storage system for resources. Clouds allow users to pay for whatever resources they use, allowing users to increase or decrease the amount of resources requested as needed. Cloud servers can be used to motivate the initiation of a business and ease its financial burden in terms of Capital Expenditure and Operational Expenditure.

There are many questions that arise as to whether a cloud is secure enough. Considering malicious intruders, there are many kinds of possible attacks, such as a Wrapping attack, Malware-Injection attack, Flooding attack and Browser attack. A Wrapping attack is done by duplication of the user account and password in the log-in phase so that the SOAP (Simple Object Access Protocol) messages that are exchanged during the setup phase between the Web browser and server are affected by the attackers. In a Malware-Injection attack, the attacker creates a normal operation, such as deleteUser, and embeds in it another command, such as setAdminRight. So, when the user request is passed to the server, rather than the server executing the command as if it were deleting a user account, it actually discloses a user account to the attacker. A Flooding attack occurs when an attacker generates bogus data, which could be resource requests or some type of code to be run in the application of a legitimate user, engaging the server's CPU, memory and all other devices to compute the malware requests. The servers finally end up reaching their maximum capacity, and thereby offload to another server, which results in flooding. A Browser attack is committed by sabotaging the signature and encryption during the translation of SOAP messages in between the web browser and web server, causing the browser to consider an adversary as a legitimate user and process all requests communicating with web server.

In addition, if any kind of failure occurs, it is not clear who is the responsible party. A failure can occur for various reasons: 1) due to hardware, which is in the Infrastructure as a Service (IaaS) layer of the cloud; 2) due to malware in software, which is in the Software as a Service (SaaS) layer of the cloud; or 3) due to the customer's application running some kind of malicious code, the malfunctioning of the customer's applications or a third party invading a client's application by injecting bogus data. Whatever the reason, a failure can result in a dispute between the provider and the clients. From the client point of view, data loss or interruption in computation can cost financially as well as affect a business reputation. From the provider point of view, the quality of service (QoS) is hampered, the Service Level Agreement (SLA) is not being satisfied and there can be unnecessary charges to the customers for which the customer is not responsible. These are all costly, affecting the provider's business reputation.

Considering the above issues, one of the main focuses of cloud computing is its security. In this paper, we identify some prime security issues in cloud computation, try to identify the root cause of the failures and propose some solutions. Our observations in this paper will be specific to each issue rather than imposing security as a whole.

The rest of this paper is organized as follows. In the next section we describe some related work. In Section III some security issues and the root causes are elaborated upon, followed by some approaches to solve these problems. Finally the conclusion is presented with thoughts for our future work and improvements in Section IV.

II. RELATED WORK

Factored operating systems (fos) are designed to address the challenges found in systems, such as cloud computing and many core systems. In reality there are several classes of systems having similarities to *fos*: traditional microkernels, distributed OS's and cloud computing infrastructure. Traditional microkernels include Mach [2] and L4. Instead of simple exploitation of parallelism between servers, *fos* seeks to distribute and parallelize within a server for a single high level function [1].

In a cloud, the OS at the customer end is considered as the Virtual Machine (VM) and the application with a specific job or request is running as an instance in the provider's end. Virtual Machine Interface (VMI) was proposed to monitor VMs in [3]. *Manitou* [5] is a system that ensures that a VM can only execute authorized code by computing the hash of each memory page before executing the code. Manitou sets the executable bit for the page only if the hash belongs to a list of authorized hashes. *Lares* [6] is a framework that can control an application running in an untrusted guest VM by inserting protected hooks into the execution flow of a process to be monitored. The hooks' responsibility is to transfer control to a security VM that checks the monitored application using VMI and security policies.

Virtual machine image repositories such as VMware's Virtual Appliance Market Place [7] and Amazon's EC2 [8] have emerged. However, these repositories provide only basic services such as image store and retrieval. They do not provide security management of VM images within the repository. There is also work for imposing security, like VM introspection for security, Memory Protection, Secure Code Execution, and Secure Control Flow [4].

All of this work faces some drawbacks when security is considered in a cloud, which is our motivation for this work.

III. SECURITY ISSUE CAUSES AND SOLUTIONS

We will focus on specific problems for various kinds of attacks in the cloud: a) Wrapping attack, b) Malware-Injection attack, c) Flooding attack, d) Data stealing that can result from a Browser attack and e) Accountability checking. We describe each of these prime security issues in cloud systems and depict their root causes. We then present approaches to mitigate such attacks to ensure the integrity and security of cloud systems.

Wrapping Attack Problem: When a user makes a request from his VM through the browser, the request is first directed to the web server. In this server, a SOAP message is generated. This message contains the structural information that will be exchanged between the browser and server during the message passing. Before message passing occurs, the XML document needs to be signed and canonicalization has to be done. Also, the signature values should be appended with the document. Finally, the SOAP header should contain all the necessary information for the destination after computation is done.

For a wrapping attack, the adversary does its deception during the translation of the SOAP message in the TLS (Transport Layer Service) layer. The body of the message is duplicated and sent to the server as a legitimate user. The server checks the authentication by the Signature Value (which is also duplicated) and integrity checking for the message is done. As a result, the adversary is able to intrude in the cloud and can run malicious code to interrupt the usual functioning of the cloud servers.

Wrapping Attack Solution: Since an adversary can intrude in the TLS layer; we propose to increase the security during the message passing from the web server to a web browser by using the SOAP message. Specifically, as the signature value is appended, we can add a redundant bit (STAMP bit) with the SOAP header. This bit will be toggled when the message is interfered with by a third party during the transfer. When it is received in the destination, the STAMP bit is checked first and if it is found toggled, then a new signature value is generated in the browser end and the new value sent back to the server as recorded to modify the authenticity checking.

The adversary can no longer interrupt the customer request with a duplication of the SOAP body because the previous signature value is already altered. For this purpose, only a random signature value generator is needed in the browser end and only the extra message overhead of one bit is required for an authenticity check.

Malware-Injection Attack Problem: In a malware-injection attack, an adversary attempts to inject malicious service or code, which appears as one of the valid instance services running in the cloud. If the attacker is successful, then the cloud service will suffer from eavesdropping. This can be accomplished via subtle data modifications to change the functionality, or causing deadlocks, which forces a legitimate user to wait until the completion of a job which was not generated by the user. Here the attacker takes his first step by implementing his malicious service in such a way that it will run in Iaas or SaaS of the cloud servers, for example as mentioned in Section I, with deleteUser and setAdminRights. This type of attack is also known as a meta-data spoofing attack.

When an instance of a legitimate user is ready to run in the cloud server, then the respective service accepts the instance for computation in the cloud. The only checking done is to determine if the instance matches a legitimate existing service. However, the integrity of the instance is not checked. By penetrating the instance and duplicating it as if it is a valid service, the malware activity succeeds in the cloud.

Malware-Injection Attack Solution: Usually when a customer opens an account in the cloud, the provider creates an image of the customer's VM in the image repository system of the cloud. The applications that the customer will run are considered with high efficiency and integrity. We propose to consider the integrity in the hardware level, because it is very difficult for an attacker to intrude in the IaaS level. We utilize the File Allocation Table (FAT) system architecture, since its straightforward technique is supported by virtually all existing operating systems. From the FAT table we can know about the code or application that a customer is going to run. We can check with the previous instances that had been already executed from the customer's machine to determine the validity and integrity of the new instance. For this purpose, we need to deploy a Hypervisor in the provider's end. This Hypervisor will be considered the most secured and sophisticated part of the cloud system whose security cannot be breached by any means. The Hypervisor is responsible for scheduling all the instances, but before scheduling it will check the integrity of the instance from the FAT table of the customer's VM.

Another approach is to store the OS type of the customer in the first phase when a customer opens an account. As the cloud is totally OS platform independent, before launching an instance in the cloud, cross checking can be done with the OS type from which the instance was requested from with the account holder's OS type.

Flooding Attack Problem: In a cloud system, all the computational servers work in a service specific manner, with internal communication between them. Whenever a server is overloaded or has reached the threshold limit, it transfers some of its jobs to a nearest and similar service-specific server to offload itself. This sharing approach makes the cloud more efficient and faster executing requests.

When an adversary has achieved the authorization to make a request to the cloud, then he/she can easily create bogus data and pose these requests to the cloud server. When processing these requests, the server first checks the authenticity of the requested jobs. Because non-legitimate requests must be checked to determine their authenticity, checking consumes CPU utilization, memory and engages the IaaS to a great extent. While processing these requests, legitimate services can starve, and as a result the server will offload its services to another server. Again, the same thing will occur and the adversary is successful in engaging the whole cloud system just by interrupting the usual processing of one server, in essence flooding the system.

Flooding Attack Solution: For preventing a flooding attack, our proposed approach is to organize all the servers in the cloud system as a group of fleet of servers. Each fleet of servers will be designated for specific type of job, e.g. one fleet will be engaged for file system type requests, another for memory management and another for core computation related jobs, etc. In this approach, all the servers in the fleet will have internal communication among themselves through message passing. So when a server is overloaded, a new server will be deployed in the fleet and the name server, which has the complete records of the current states of the servers, will update the destination for the requests with the newly included server.

As mentioned in the above section, a Hypervisor can also be utilized for the scheduling among these fleets, determining the authenticity of the requests and preventing the fleets from being overloaded with bogus requests from an adversary. In this way the flooding attack can be mitigated to an extent (if the Hypervisor is locally breached, then further analysis and efforts will be required to secure the Hypervisor.)

Also, a PID can be appended in the messaging, which will justify the identity of the legitimate customer's request and be checked by the Hypervisor in the assignment of instances to the fleet of servers. This PID can be encrypted with the help of various approaches, such as implementing hash values or by using the RSA.

Data Stealing Problem: This is the most traditional and common approach to breach a user account. The user account and password are stolen by any means. As a result, the subsequent stealing of confidential data or even the destroying of data can hamper the storage integrity and security of the cloud. The providers face the first strike of such kind of problem.

Data Stealing Solution: At the end of every session, the customer will send an e-Mail about the usage and duration with a special number to be used for log in next time. In this way, the customer will be aware of the usage and charges as well as be availed with a unique number to be used every time to access the system. In Amazon EC2, a key pair is used to verify the authenticity of the customer, but this approach only needs the special number appended with the UserName. There will be an overhead for sending e-Mail to all the customers with a randomly generated number when their session will expire. Eventually, as mentioned earlier, the PID generator inside the Hypervisor can be appointed to commit the task.

Accountability Check Problem: The payment method in a cloud System is “No use No bill”. When a customer launches an instance, the duration of the instance, the amount of data transfer in the network and the number of CPU cycles per user are all recorded. Based on this recorded information, the customer is charged. So, when an attacker has engaged the cloud with a malicious service or runs malicious code, which consumes a lot of computational power and storage from the cloud server, then the legitimate account holder is charged for this kind of computation. Though the customer is not aware of the attack and until the main cause of the CPU usage is detected, the providers will charge the customers first. As a result, a dispute arises and business reputations are hampered. All the focus for charging is based on the recorded parameters.

Accountability Check Solution: The provider does not know the details of the customer’s applications and it does not have the privilege to test the integrity of the application running in the cloud. On the other hand, customers do not know the infrastructure of the provider’s cloud. If a customer is charged due to a malware attack or a failure, then the customer has no option to defend himself.

So when there is an unusual phenomenon, before charging the customer, an investigation should take place. In our approach there will be some features to be ensured in the provider’s end before launching any instance of a customer: 1) Identities, 2) Secure Records, 3) Auditing and 4) Evidence. Firstly, before starting the instance, the identity of the legitimate customer should be checked, which will be done by the Hypervisor. Secondly, all the message passing and data transfer in the network will be stored securely and uninterrupted in that specific node. Hence, when the auditing takes place, all the necessary information can be retrieved. Also, the evidence must be strong enough to clarify the recorded events, so the AUDIT will have the following properties: Completeness, Accuracy and Verifiability. These properties ensure that when there is a security attack it should be reported immediately, no false alarm will be reported and the evidence can be scrutinized by a trusted third party who will commit the task of AUDIT from a neutral point of view.

Also in our approach, we will keep the privacy of both the providers as well as customers, by keeping the log of all the records available to only the victim customers and not to all the customers in the cloud. The log available to customers will not have any confidential information about the infrastructure of the provider from which the IaaS can be inferred by the customer. However, details will be disclosed to the third party step by step until the problem is detected and the dispute is resolved.

IV. CONCLUSIONS AND FUTURE WORK

Cloud computing is revolutionizing how information technology resources and services are used and managed, but the revolution always comes with new problems. We have depicted some crucial and well known security attacks and have proposed some potential solutions in this paper, such as utilizing the FAT table and a Hypervisor.

In the future, we will extend our research by providing implementations and producing results to justify our concepts of security for cloud computing. The concepts we have discussed here will help to build a strong architecture for security in the field of cloud computation. This kind of structured security will also be able to improve customer satisfaction to a great extent and will attract more investors in this cloud computation concept for industrial as well as future research farms. Lastly, we propose to build strong theoretical concepts for security in order to build a more generalized architecture to prevent different kinds of attacks.

REFERENCES

[1] D. Wentzlaff, C. Gruenwald III, N. Beckmann, K. Modzelewski, A. Belay, L. Touseff, J. Miller, and A. Agarwal. Fos: A Unified Operating System for Clouds and Manycore. *Computer Science and Artificial Intelligence Laboratory TR*, Nov. 20, 2009.

[2] M. Accetta, R. Baron, W. Bolosky, D. Golub, R. Rashid, A. Tevanian, and M. Young. Mach: A new kernel foundation for UNIX development. In *Proc. of the USENIX Summer Conference*, pp. 93-113, June 1986.

[3] T. Garfinkel, M. Rosenblum, A virtual machine introspection based architecture for intrusion detection. *Proc. 2003 Network and Distributed Systems Symposium*, 2003.

[4] M. Christodorescu, R. Sailer, D. L. Schales, D. Sgandurra, D. Zamboni. *Cloud Security is not (just) Virtualization Security*, CCSW’09, Nov. 13, 2009, Chicago, Illinois, USA.

[5] L. Litty and D. Lie. Manitou: a layer-below approach to fighting malware. In *ASID ’06: Proc. of the 1st workshop on Achitectural and system support for improving gsoftware dependability*, pages 6-11, New York, NY, USA, 2006. ACM.

[6] B.D. Payne, M. Carbone, M. Sharif, and W. Lee. Lares: An architecture for secure active monitoring using virtualization. *Security and Privacy, IEEE Symposium on*, 0:233-247, 2008.

[7] VMware. Virtual Appliance Marketplace. http://www.vmware.com/appliances/.

[8] amazon. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2.