Engineering Sciences International Research Journal
ISSN (Print) : 2330 – 4338 ; Volume 1 Issue 1 ǁ February. 2013 ǁ

An Analysis of Malware Injection Attacks and Their Impact on Cloud

TulasiRam N[1], Anusha K[2], Mary Saira Bhanu S[3]

Abstract: Cloud computing is an apparent solution for many organizations who look for satisfying their fluctuating business demands. The characteristics like scalability, reliability and high performance makes it highly adoptable. As the organization's data is stored off-premises, security is a threatening factor. Majority of the cloud services are accessible by the end users through a web interface, exposing them to malicious software (malware), which disrupts the intended functionality of the service. Malware can be any active content such as scripts, commands and other software codes. According to a recent survey conducted by SophosLabs (2012), 80% of overall infected sites are legitimate and injection attacks form the majority of security threats. This paper analyzes the following malware injection attacks: SQL injection, Cross-site scripting, Command injection and presents a procedure to understand their effect on cloud. Up on analyzing, we propose a cloud based generic framework that can prevent the above-mentioned attacks.

Keywords: Cloud computing, Command injection, SQL Injection, XSS.

1. INTRODUCTION

Cloud computing is an emerging technology which provides computing resources as services over Internet. It refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. Multi-tenancy and Virtualization are key ideas of this model. Multi-tenancy can be viewed as compartmentalizing different customers to share the same underlying resources. Virtualization refers to providing a layer of abstraction between computer hardware and software, offering a logical view of resources like cpu, storage, network etc.

Reference [6] states that 61% of the respondents would be using the cloud technology in near future. Contradicting the above statement [6] also mentions that 52% of the organizations do not use cloud in order to avoid data risk. Some well-publicized incidents in past emphasize the threats that may be possible on cloud. To quote one, Dropbox is one of the most popular cloud file sharing services with about 45 million users.

Researchers discovered at least three different ways of hacking it bypassing authentication. This indicates that security of cloud is an issue of high concern. There are various attacks that target the cloud at software service level and infrastructure level. Software service level attacks include SQL Injection (Sqlia), Cross site Scripting (XSS), Command injection (CI) etc. The consequences of these attacks range from service level sensitive data theft to injecting worms on single/multi tenant services. At infrastructure level the attacks try to gain control over the hypervisor.

Injection attacks exploit the vulnerabilities in input sanitization. Sqlia embeds harmful SQL statements in the data context of an application. XSS attacks breaches security by injecting code, which executes on client side web browser. CI attacks the web application by injecting platform specific system commands. In this paper, we experiment the above mentioned malware injection attacks on a cloud test bed, analyze the degree of their impact and propose a system that can counteract the above mentioned attacks by deploying the solution as "Security as a service".

Section II gives the related study about injection attacks on a network. Section III identifies the attack surfaces of cloud. Section IV presents the attack generation steps designed to launch various attacks and their impact. Section V proposes a generic framework for security-as-a-service. Section VI explains the experimental setup. The remainder of this paper concludes and gives the future work.

2. RELATED WORK

The migration of networks and servers to cloud also migrates the security threats. This section gives the various injection attacks that are possible on plain network which are also applicable to the cloud environment. The first kind of attacks is Sqlia. They can be introduced into the web browser either through input fields, cookies or through server side parameters. They aim at database fingerprinting, extracting data, adding or modifying data present in database. Reference [10] gives a detailed survey of the Sqlia and their prevention methods. Many of the Database Managements Systems come up with inbuilt protection mechanisms against Sqlia so that the programmer to write an efficient code can use them.

Reference [4] describes various built in functions provided by Oracle Systems. Reference [9] gives a solution that counter acts the attack in two stages: static analysis, run time analysis. Static analysis helps in building legitimate query models that an application produces and run time enforcement of these models can filter the injected code. Reference [7] gives the idea of encrypting the users data that avoids Sqlia, but this requires additional fields to be stored in the database and consumes additional time to encrypt and decrypt frequently. The second type of attack is the Command Injection attack whose purpose is to inject and execute unwanted system commands through a vulnerable application. The applications are vulnerable if the user level input is used at the system level. If this vulnerability occurs in a privileged prog- ram it can specify the commands that are normally not accessed by user. Reference [1] describes several examples on how to produce this attack. The third type is the Cross Site Scripting attack where malicious scripts are injected into a trusted web application through a client site web browser. XSS attacks are classified as stored and reflected depending on how the code is injected into the web application. A variety of solutions exist to avoid XSS over plain network. Reference [3] describes several filter-based mechanisms. Other methods include computation of hash values before and after the injection to detect the attack, proxy based client side solutions, and solutions that can be embedded at kernel level.

3. ATTACK SURFACES

In order to access the services deployed on the cloud the end-user has to undergo below mentioned levels of authentication depending on deployment model.

Public cloud requires two levels of authentication where as in private cloud, user can directly access services through

level #1 : Authenticate himself as a valid cloud user.

level#2 : Authenticate himself as a valid service user.

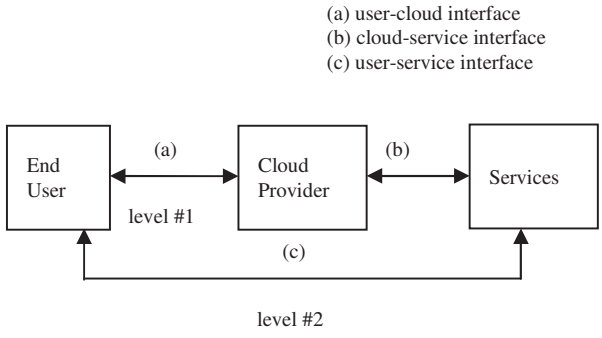

Fig. 1. Two level authentication

level #2. The above two levels can be exploited by a malicious user to launch various attacks. The malware injection attacks are viable at both levels differing in the impact they create at each level.

4. MALWARE INJECTION ATTACKS

A. SQL Injection Attack

Sqlia targets the database underlying an application through a user input field. A destructive sql command is given as a part of the input field which when substituted into the sql query makes it a valid one but performs a unexpected harmful action.

Attack generation steps

1. Malicious user authenticates himself to the cloud provider

2. He authenticates himself to the subscribed service which is vulnerable

3. He injects the SQL queries in the input fields of the subscribed service and gathers the required database information.

4. He successfully views the sensitive data related to victim service without the need to subscribe to it.

Impact on cloud:

Based on vulnerability identified on one cloud application the Sqlia is able to view

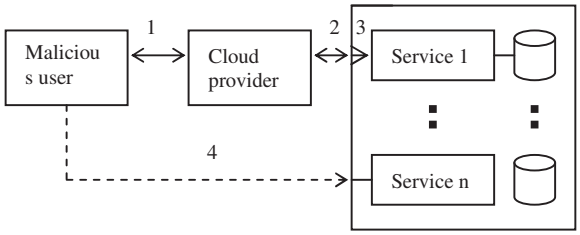

Fig. 2. Sql injection attack scenario in cloud

view the sensitive data pertaining to other applications that are deployed on the same infrastructure. In this way Sqlia can be used to produce a service-to-service attack on the applications.

B. Cross site scripting attack (XSS)

XSS deals with injecting code into data context of HTML based documents at client and gaining access to sensitive information from server. It allows an attacker to execute

scripts in victims' web browser. OWASP classifies XSS attacks as stored and reflected.

According to the WHID (2011), 12.58% of the overall attacks on the web are associated with XSS. The variety of attacks based on XSS is almost limitless.

Attack generation steps

1. Malicious user authenticates himself against cloud provider.
2. Malicious user tries to access a vulnerable web application.
3. Web application gets loaded onto the malicious users browser.
4. Malicious user injects malware from an attackers' server via an URL.
5. Contents get updated on server side upon submit action.
6. Victim user authenticates himself against cloud provider.
7. Victim user tries to access the same web application that malicious user accessed.
8. Malicious code gets loaded onto Victim users browser.
9. Malware sends almost every detail that it can extract via browser to an attacker server.

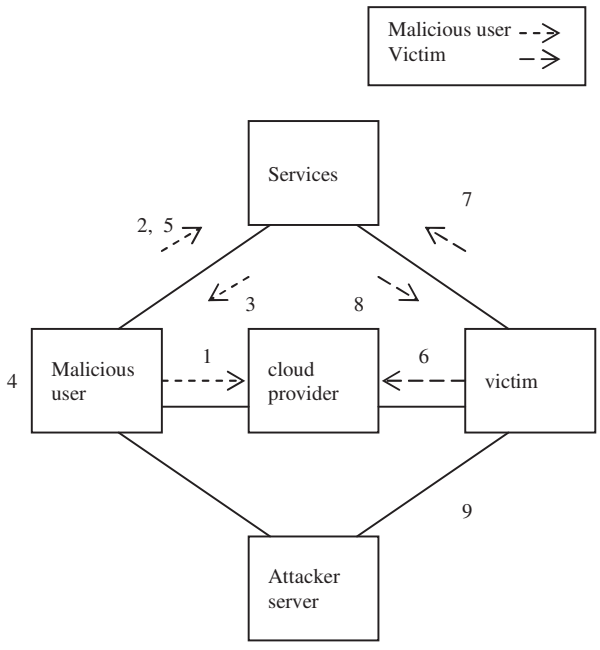

Fig. 3. Cross site scripting attack scenario in cloud

Impact on cloud

At level#1, XSS attack on user-to-cloud interface can create considerable impact by hijacking cloud users active session and can exploit the user's privileges to the full extent. A successful attack on cloud interfaces results in consequences such as full access to the victim user's data, creation and deletion of virtual machine images, control over administrative settings etc. breaching the cloud properties like virtualization and multi-tenancy.

A well-known real world example "The Amazon EC2 Hack":

XSS attack in combination with XML Signature wrapping attack on control interfaces: user-to-cloud interface and cloud-to-service interface leads attacker to gain control over legitimate user account and thus allowing him to perform denial of service to other legitimate users.

At level#2, XSS attack on user-to-service interface has relatively less impact compared to the impact on level#1. It targets a specific vulnerable service. A successful attack on such a service can lead to service-to-service attacks. It can steal sensitive information like document cookies, browsing history etc. thus breaching the victim user's privacy.

Command Injection attack

Command injection is a type of code injection where the commands are injected in identified vulnerable applications. It allows such inputs to get executed on shell or in the respective runtime environment. The injected commands like ls, ps, cat etc. get executed in the runtime environment with the same privileges that a targeted application posses. One of the major consequences of the above attack is increased waiting time for the other users who makes use of applications running on the same VM in which vulnerable application runs.

Attack generation steps:

1. Malicious user authenticates himself against cloud provider.
2. Malicious user tries to access a vulnerable web application.
3. Web application gets loaded onto the malicious users browser.
4. Malicious user injects command by modifying HTTP request (GET or POST) parameters.
5. Victim user authenticates himself against cloud provider.
6. Victim user tries to access a web application.
7. Web interface gets loaded in the victim user's browser and the user experiences long waiting time as a result of step #4.

ISBN: 978-93-81583-90-6

Impact on the cloud

CI attack has considerable impact on user-to-service interface. Malicious user

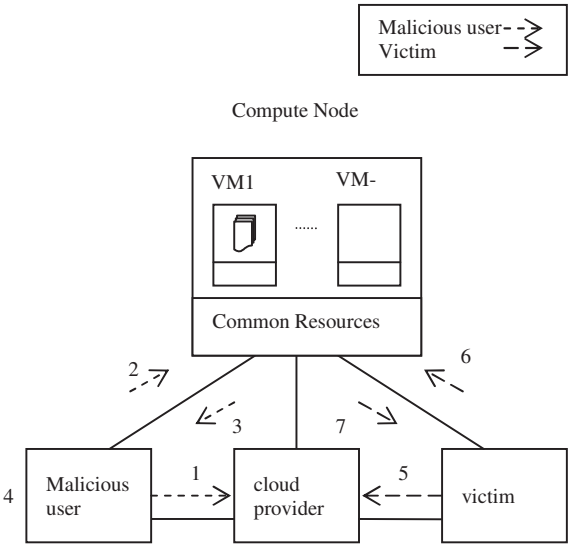

Fig. 4. Command injection attack scenario in cloud

can run the commands with the same privileges the vulnerable application posses. A successful attack results in consequences such as, listing the files, displaying the contents of a file, viewing the list of processes currently being executed etc.

Simple attack vector: "while(true); do <cmd-1..cmd-N>;done" result in consump- tion of CPU cycles, memory resources which sometimes lead to denial of service.

5. PROPOSED SYSTEM

Existing solutions over network provide either vulnerability checkers with static rules or solutions that address specific class of attacks. In cloud, wide range of services are being deployed day-to-day, therefore the solutions should be able to handle new attack vectors. Deployment of such solutions should not create bottlenecks. This necessitates a generic framework which can be deployed as a service so that it is scalable.

Every incoming HTTP request passes through the following detection modules: Sql injection, CI, XSS. Each module employs a detection mechanism to deal with its corresponding attack. The Sql injection module maintains for each application, a set of query models. A query model is a state diagram of an sql query dictating what should be there in the query. The models are enforced on the incoming input and if any mismatch is found the request is discarded. If no Sqlia is detected in this module the query is forwarded to the CI detection module. The request is first parsed into tokens then is checked against the database containing a set of system commands and user defined commands. If any of them is identified the request is discarded, if not it is forwarded to the XSS module. It initially employs filter based mechanisms on the input and then the parser extracts the links present in the request. It checks each link against the black listed database, if present the request is discarded, if not it is checked for any suspicious keywords, if identified the request is added to the blacklist and discarded, if not the request is accepted.

6. EXPERIMENTAL SETUP

The attacks are tested on Eucalyptus cloud. Eucalyptus is a software platform that provides an on-premise private cloud.

7. CONCLUSION

Vulnerabilities in web applications enable unauthorized access to cloud. Even a minute vulnerability in any one of the applications may result in a security breach of other multi-tenant services on cloud. Hence effective security mechanisms play vital role. To understand the impact of malware injection attacks and the necessity to provide counter measures, we have devised and tested various attack vectors in each class of malware injection attacks. A generic solution to prevent injection attacks is provided as a service on cloud. Our future work includes, building a self-learning system that detects the type of attack and can sustain the possible attack vectors.

Experimental setup configuration details

	Server#1	**Server#2**	**Client**
Functionality	**CLC,CC, Walrus**	**NC**	**Web UI client**
No. of NICs	eth0 - Enterprise N/W eth1- Eucalyptus N/W	eth0 - Eucalyptus N/W	eth0 - Enterprise N/W eth1 -Eucalyptus N/W
IP Addresses	eth0 - 10.1.60.2 eth1 - 10.1.50.4	eth0 - 10.1.50.3	eth0 - 10.1.60.3 eth1 - 10.1.50.5
VT enabled	No	Yes	Yes
CPU	64bit,3GHz,Multi core	64bit,3GHz,Multicore	64bit,3GHz,Multicore
Networking	100Mbps	100Mbps	100Mbps
Memory	4GB	4GB	4GB

8. REFERENCES

[1] CWE: Individual Dictionary Definition. “Improper Neutralization of Special Elements used in a command” https://cwe.mitre.org/data/definitions/77.html

[2] Juraj Somorovsky, Mario Heiderich, Nils Gruschka, Luigi Lo Iacono, "All Your Clouds are Belong to us– Security Analysis of Cloud Manage- ment Interfaces" CCSW'11, October 21, 2011, Chicago, Illinois, USA.

[3] Mario Heiderich, Marcus Niemietz, Felix Schuster, Thorsten Holz, Jörg Schwenk ,"Scriptless Attacks Stealing the Pie Without Touching the Sill" CCS'12,North Carolina, USA.

[4] Oracle Developers,"An Introduction to SQL Injection Attacks for Oracle Developers" March 2007.

[5] OWASP. "Command Injection"https:// www.owasp.org/ index.php/ Command_Injection.

[6] Sophos security threat report 2012, accessed on http://www.sophos.com/ medialibrary/PDFs/other/Sophos SecurityThreatReport2012.pdf

[7] Sunita Rani, Ambrish Gangal," Cloud Security with Encryption using Hybrid Algorithm and Secured Endpoints",(IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (3) , 2012,4302 - 4304

[8] Sql injection prevention cheat sheet. Https://www.owasp.org/ index.php/SQL_Injection_Prevention_Cheat_Sheet, 2012.

[9] William G.J. Halfond and Alessandro Orso," AMNESIA: Analysis and Mo- nitoring for Neutralizing SQL Injecti- on Attacks", ASE'05, USA.ACM.

[10] William G.J. Halfond, Jeremy Viegas, and Alessandro Orso "A Classfication of SQL Injection Attacks and Counte- rmeasures", IEEE 2006

* * *

[1]*Tulasiram N, M.Tech, C.S.E, National Institute of Technology Trichy, 206111002@nitt.edu,*
[2]*Anusha K, M.Tech, C.S.E, National Institute of Technology Trichy, 206111009@nitt.edu,*
[3]*Dr.Mary Saira Bhanu S, Associate Professor, C.S.E, National Institute of Technology Trichy, msb@nitt.edu*