Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 85 (2016) 198 – 205

Procedia Computer Science

International Conference on Computational Modeling and Security (CMS 2016)

CSSXC: Context-Sensitive Sanitization Framework for Web Applications against XSS Vulnerabilities in Cloud Environments

Shashank Gupta[#], B. B. Gupta[*]

Department of Computer Engineering, NIT Kurukshetra, India

Abstract

This paper presents a context-sensitive sanitization based XSS defensive framework for the cloud environment. It discovers all the hidden injection points in HTML5-based web applications deployed on the platforms of cloud and sanitizes the XSS attack payloads injected in such points in a context sensitive manner. The identification of such injection points permits our technique to retrieve each possible web page of application, allowing a wider exploration and accelerating the process of applying the sanitizers on the untrusted variables of web application. The XSS attack mitigation capability of our framework was evaluated on web applications deployed for the cloud users in the cloud environment. The experimental results reveal that this technique detects the XSS attack payloads with minimum rate of false negatives and less runtime overhead.

*Keywords:*Cloud Computing; Cross-Site Scripting (XSS) attacks; JavaScript Code Injection Attacks; HTML5; Cloud Security.

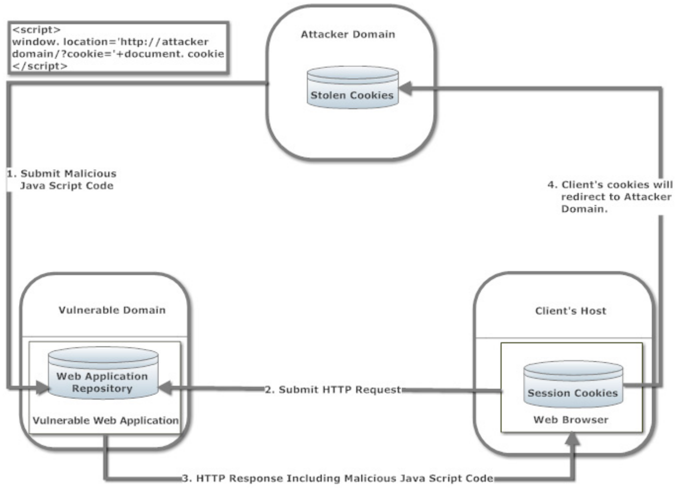
1. Introduction

Currently, cloud computing is apparently considered to be the utmost outlook technologies because of its high flexibility as well as its cost effectiveness (Almorsy, 2010). Instead of referring the outdated Internet settings for constructing an expensive setup, nowadays numerous IT enterprises utilize the capabilities of cloud techniques (Tiwari, 2015). Therefore, several IT organizations install the setup of their web applications in the infrastructures of cloud. However, it is clearly known that the cloud settings are installed on the backbone of Internet. Therefore, numerous web application vulnerabilities in the conventional Internet infrastructures also exist in the backgrounds of cloud-based environments. XSS vulnerability is considered to be one of topmost web application vulnerability (Gupta, 2015a) and have now turned out to be a critical security concern, as web applications are facing this problem from the time when XSS first time discovered in the year 2000.

E-mail address: gupta.brij@gmail.com [*](Tel. +91-1744-233488), mr.shashank.gupta@ieee.org[#]

doi:10.1016/j.procs.2016.05.211

This attack generally occurs due to the injection of malicious scripts in the vulnerable injection points of web application (Gupta, 2014).Fig. 1 highlights the abstract view of XSS attack. XSS falls third in the list according to the statistics of OWASP Top 10 2013 (OWASP, 2013). Recently, numerous approaches have been proposed to detect and mitigate the effect of XSS vulnerabilities from real world web applications. XSS Auditor (Bates, 2010) is a filter that realizes equally extraordinary performance as well as high accuracy via jamming scripts following the HTML parsing and prior to execution. Noncespaces (Gundy, 2009) is an end-to-end mechanism that facilitates web browsers to differentiate between benign and malicious content to apply the techniques from Instruction Set Randomization (ISR) for thwarting the exploitation of XSS vulnerabilities. XSS-Guard (Bisht, 2008) is a server-side solution for defending against the XSS attacks by discovering the collection of scripts that a web application intends to create for any HTML web request. BLUEPRINT (Louw, 2009) is a server-side solution for thwarting XSS attacks where the web application transfers two replicas of output HTML document to a web browser for detecting any deviation, one with user inputs and other with legitimate values. (Saxena, 2010) proposed a hybrid and dynamic analysis methodology, which examine the JavaScript-based web applications for the discovery of input validation vulnerabilities. This technique was executed in a tool known as FLAX, a taint enriched black-box fuzzer, which is capable of finding the client side validation bugs in Java script programs. However, the testing of FLAX has not focused on the complexity of sanitization errors, which still remain in the client-side Java script code. (Livshits, 2013) proposed an automated technique of sanitizer placement by statically analyzing the stream of infected data in the program. However, this technique presumes every possible source, sinks and sanitizers to be identified in advance, and as a result suffers from non-tolerable runtime overhead.



```
<script>
window. location='http://attacker
domain/?cookie='+document. cookie
</script>
```

Attacker Domain
Stolen Cookies
1. Submit Malicious Java Script Code
4. Client's cookies will redirect to Attacker Domain.
Vulnerable Domain
Web Application Repository
Vulnerable Web Application
2. Submit HTTP Request
Client's Host
Session Cookies
Web Browser
3. HTTP Response Including Malicious Java Script Code

Fig. 1. Abstract View of XSS Attacks

With the escalating boom in the field of cloud computing, the investigation on the security of web applications deployed in the cloud environment turn out to be a significant challenge. XSS attacks are turned out to be plague for the modern HTML5-based web applications (Gupta, 2012). Therefore, the defense of XSS attack has now become a pre-requisite for the web applications deployed on the platforms of cloud computing. In addition to this, the existing XSS defensive solutions are inappropriate for the cloud platforms and cannot be easily integrated in the cloud environments. As such solutions demand countless modifications in web browsers and in the source code of web applications. In addition to this, existing XSS defensive solutions do not provide real time protection (Gupta, 2016).

To address these issues, this paper presents a robust XSS defensive framework for the HTML5-based web applications deployed in the cloud environment. It discovers all the hidden injection points in the cloud-based web applications and performs context-sensitive sanitization on the XSS attack payloads injected in such points for mitigating the effect of XSS vulnerabilities from these web applications. This framework does not need any alterations required on the web browser and in the source code of web applications. The testing and evaluation of our work was done on real world web applications deployed in the cloud environment. The observed results