A STUDY ON EXPLOITATION AND PREVENTION FOR PHP VULNERABILITY IN WEB APPLICATIONS

[1]KOTA SUBRAHMANYA KASHYAP, [2]KUDA NAGESWARA RAO, [3]MALLAVALLI SITHA RAM

[1]M.TECH STUDENT, [2]PROFESSOR, [3]RESEARCH SCHOLAR
Department of Computer Science & Systems Engineering,
Andhra University College of Engineering (A), Visakhapatnam, India

Abstract: Cyber-attacks became more refined with attackers using new techniques to exploit vulnerabilities present in existing systems. Even when developers try to remove these vulnerabilities, attackers find other ways to exploit systems. Applications with underlying vulnerabilities pave way to be exploited. Web Applications typically contain a vulnerability known as Cross-site scripting. This vulnerability allows attackers to send malicious code in form of scripts which get executed at client side into web pages which are viewed by different users. Attackers can use this vulnerability to access controls such as same origin policy. An attacker creates a payload which is basically a file (in this case a php file) which is uploaded into the server. When a user accesses a web page using browser, the malicious script is executed and the server is accessed by the attacker. Additionally, attacker can also send input which can be captured later by another script externally. The browser on user side will execute the script since it can't validate it. This script can access session tokens, cookies and sensitive information stored by the browser. This vulnerability can be forestalled with appropriate measures. Data Validation is one such measure which can be used to forestall cross-site scripting vulnerability.

***Key Terms*- Cyber-attacks, Web applications, Cross-site scripting, PHP vulnerability.**

I. Introduction

Cyber security consists of processes and technologies which are designed to guard systems and networks from unauthorized access. Effective cyber security controls involve the support of technology, processes and people. This approach helps organizations defend themselves from organized attacks and insider threats, like accidental breaches and human error. The need for cyber security has increased in recent times. Web Application security is branch of Cyber security. While developing web applications and web pages, if the developers aren't trained enough vulnerabilities arise. Even though the developers are well trained vulnerabilities may arise when the end users use the application for unintended purposes like accessing untrusted websites.
Attackers prowling on those websites can victimize the users. These attackers can exploit such untrusted websites which have vulnerable servers. One such vulnerability is Cross-site scripting.

Cross-site scripting is one of the most critical web application vulnerabilities. From 1990's Cross-site scripting attacks have been identified. In the past several social networking sites such as Facebook, Orkut, MySpace, Youtube, Twitter were affected by Cross-site scripting. Currently Cross-site scripting alongside SQL injection, Broken authentication, XML external entity, etc., are named as Top Ten Web Applications Security Risks by The Open Web Application Security Project.

Therefore appropriate measures need to be taken in development of Web application and Web pages. The end users should have sufficient knowlegde to use these web applications and web pages.

II. Related Work

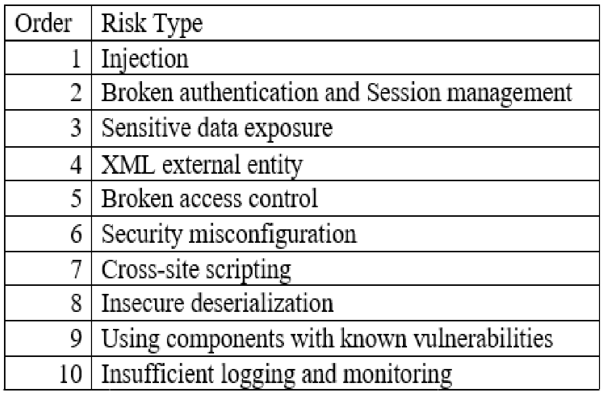
This section deals with the technologies and tools used to cause the vulnerability Cross-site scripting.

Metasploit Framework

A Linux based OS, Kali Linux, has preinstalled tools to perform penetration testing. One such tool is Metasploit Framework. It provides exploits for different platforms and applications. Using the Metasploit Framework vulnerable systems are exploited by installing payloads. These payloads by using various methods give access to those systems. Kali Linux is installed on VMware Workstation.

OWASP

OWASP stands for Open Web Application Security Project. This organization is dedicated in improving web application security. Several tools are provided by OWASP on application security for its users. OWASP provides a list of web application security risks which are most critical under the name "OWASP Top Ten".



Order	Risk Type
1	Injection
2	Broken authentication and Session management
3	Sensitive data exposure
4	XML external entity
5	Broken access control
6	Security misconfiguration
7	Cross-site scripting
8	Insecure deserialization
9	Using components with known vulnerabilities
10	Insufficient logging and monitoring

Fig.1 OWASP Top Ten Web Application Security Risks List

DVWA

This paper deals with the Stored type of Cross-site scripting attack. OWASP provides a vulnerable web application known as Damn Vulnerable Web Application (DVWA) which is used for this attack. DVWA can be used by the personnel to run different exploits which in turn helps them to study the behavior of various exploits. Aspiring ethical hackers or amateur hackers can learn the need for web application security by using this DVWA.

III. Exploiting the vulnerability

This section deals with the attack Cross-site scripting and the steps used in exploitation.

Cross-site Scripting

In Cross-site scripting when a web application uses input from a user, an attacker can start an attack using that input, which can then spread to other users as well. The end user may trust the application, which the attacker can exploit in order to do things that aren't allowed under normal conditions. An attacker often uses different techniques to encode the malicious portion of the tag, making the request seem genuine to the user.

Cross-site scripting attacks are mainly classified as Reflected and Stored. In Reflected attacks the malicious code takes another route to the victim, either via an e-mail, or on a different server. When the user submits a form or clicks on a link, obtained via an e-mail, the code is injected into the vulnerable web server, and the attack is reflected to the user's browser. The browser executes the code thinking it came from a server which can be trusted. In Stored attacks the malicious code is stored in a message forum, database, target server, or visitor log permanently.

These steps are followed while exploitation
Step 1: Build PHP payload using Metasploit Framework in Kali Linux.
Step 2: Login to OWASP on VMware Workstation as it acts as Web server.
Step 3: Login to DVWA on any browser in Windows system.
Step 4: Upload the payload in DVWA.
Step 5: Start PHP Listener in Kali Linux.
Step 6: Set Security Level as low and Reset the Database in DVWA.
Step 7: In XSS stored menu enter the script and click the Sign Guestbook button.
Step 8: View the established Session in Metasploit.
Step 9: Exploit the server using various commands in Kali.

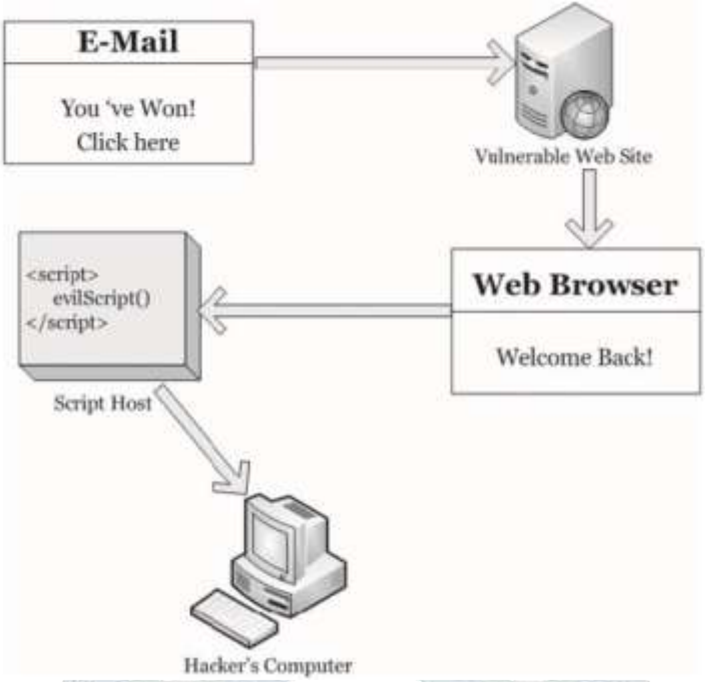


Fig.2 An example of Cross-site scripting attack

IV. Proposed Preventive Method

Data Validation

Data Validation consists of Data Filtration and Data Evasion

i) Data Filtration

Data Filtration is filtering the application data to allow data that doesn't cause damage the site, database, and users. This technique is normally related to SQL injection, but can be used to prevent Cross-site scripting. This technique is useful for preventing cross-site scripting in forms, because the special characters added by the user are stripped, instead of refusing the request. The stripped characters include null, backslashes, extra spaces etc. Escape Sequences like \b, \r, \n, \t, etc., are also filtered. Data Filtration isn't a primary prevention technique for vulnerabilities like Cross-site scripting and SQL injection, however it helps to cut back the consequences if their presence is known by an attacker.

ii) Data Evasion

Evasion of the data means securing the data received by an application before it is presented to the end user. The important characters in the received data will be prevented from being modified by any malicious code in Data Evasion. If a web page doesn't allow the users to add any code to the page, it is best to evade the entities of URL, HTML and Javascript. Whereas if it allows to add text like forums or post comments; the special characters in the code can be converted to HTML entities which can be evaded. The characters which can be converted into HTML entities are (,) , < , > ; they are converted into &#40 , &#41 , &lt , &gt respectively.

Algorithm

The Proposed method to prevent Cross-site scripting attack is

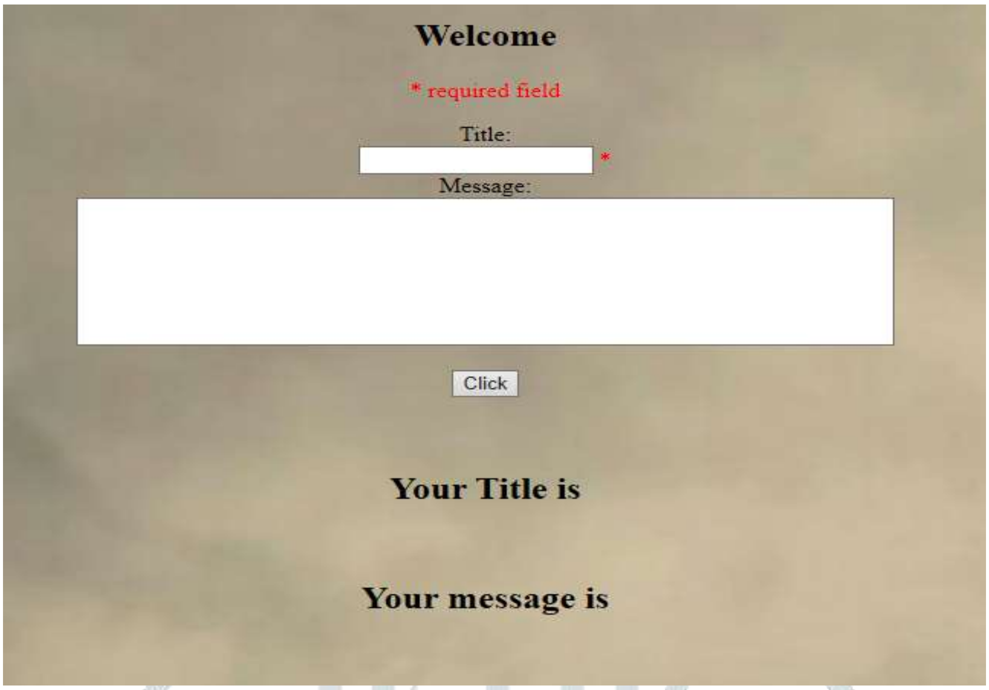
```
Step 1: Initialize
Step 2: Read the Input data
Step 3: Strip the Escape Sequences
Step 4: Strip backslashes, extra spaces and NULL characters
Step 5: Replace special characters with HTML Entities
Step 6: Return
```

V. Experimental Results

This section deals with the differences between a web page with vulnerability and a web page when the proposed preventive method is applied to it.

Create a web page which basically takes text as input and displays it. The page consists of a Title and a Message text area. The title field is a required field. After the Title and Message fields the Click button is pressed to display whatever text was entered.

The Web page looks like this



Welcome

* required field

Title: *

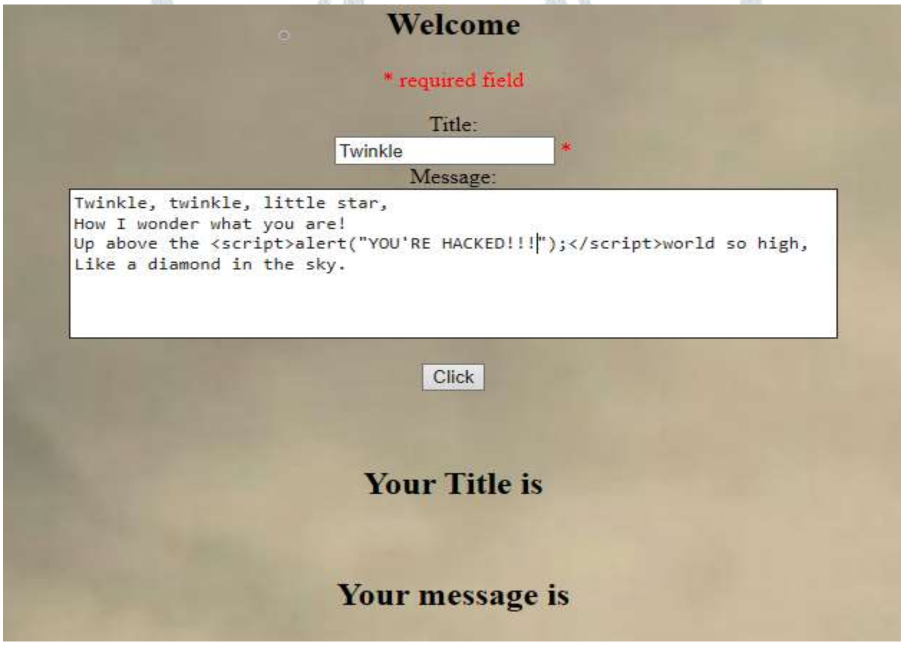
Message:

Your Title is

Your message is

Fig.3 Web page before text is entered

Fill the fields with some text. In the Message text area malicious script is inserted with normal text.



Welcome

* required field

Title: Twinkle *

Message:

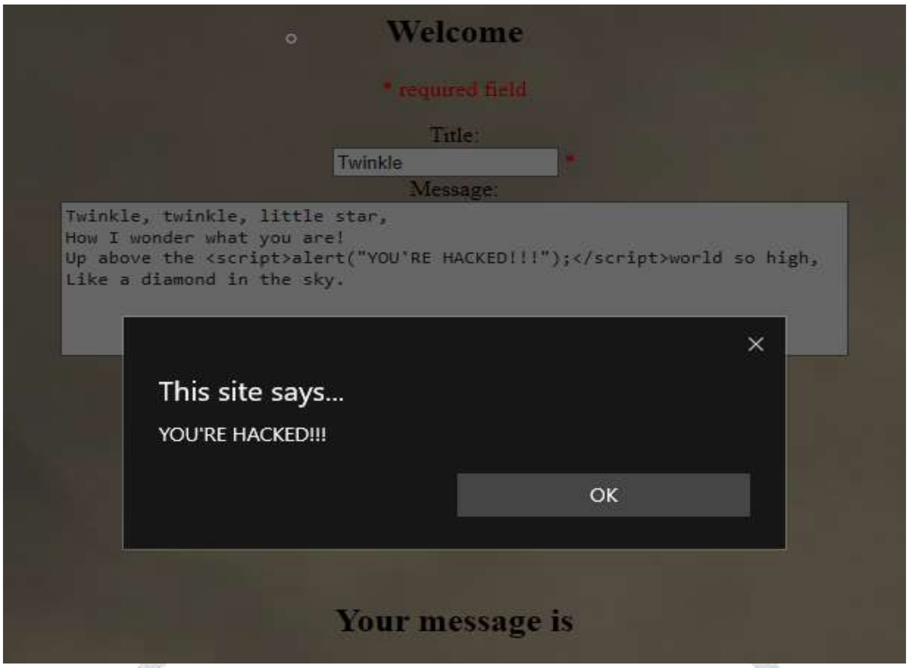
```
Twinkle, twinkle, little star,
How I wonder what you are!
Up above the <script>alert("YOU'RE HACKED!!!");</script>world so high,
Like a diamond in the sky.
```

Your Title is

Your message is

Fig.4 Web page filled with malicious script

Press the “Click” button and observe.



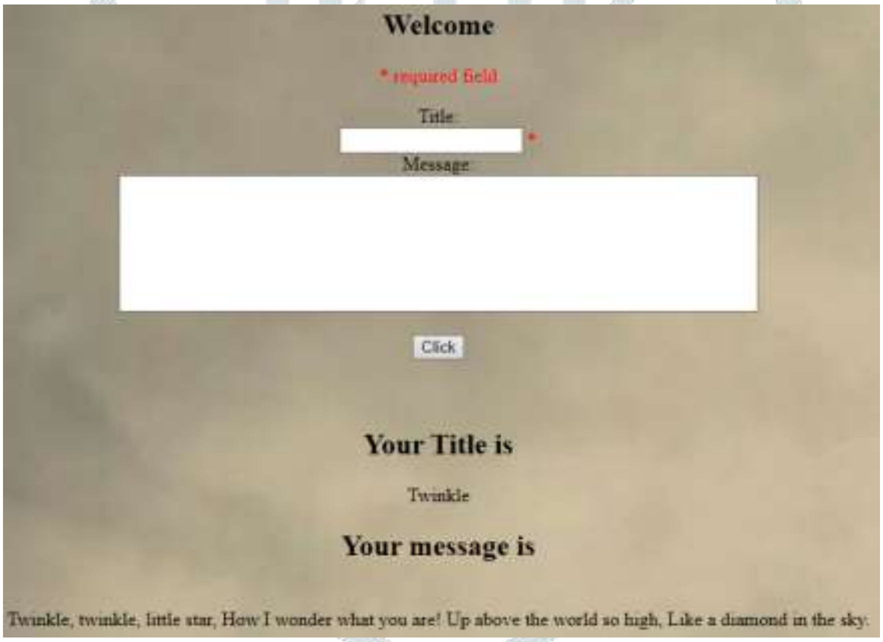
Twinkle

```
Twinkle, twinkle, little star,
How I wonder what you are!
Up above the <script>alert("YOU'RE HACKED!!!");</script>world so high,
Like a diamond in the sky.
```

This site says...

YOU'RE HACKED!!!

Fig.5 Script gets executed which generates an alert



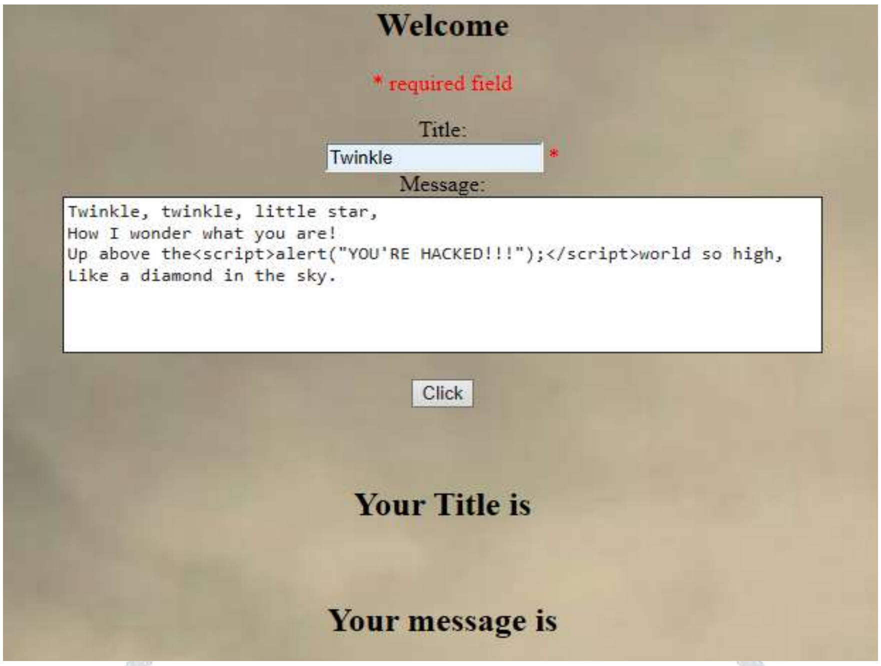
Twinkle

Twinkle, twinkle, little star, How I wonder what you are! Up above the world so high, Like a diamond in the sky.

Fig.6 The script doesn’t get displayed

The script which was executed generated an alert; since the purpose of the script is to generate an alert. The text is displayed without the script.

Apply Data validation and Fill the form again with the malicious script.



Welcome

* required field

Title:

Twinkle *

Message:

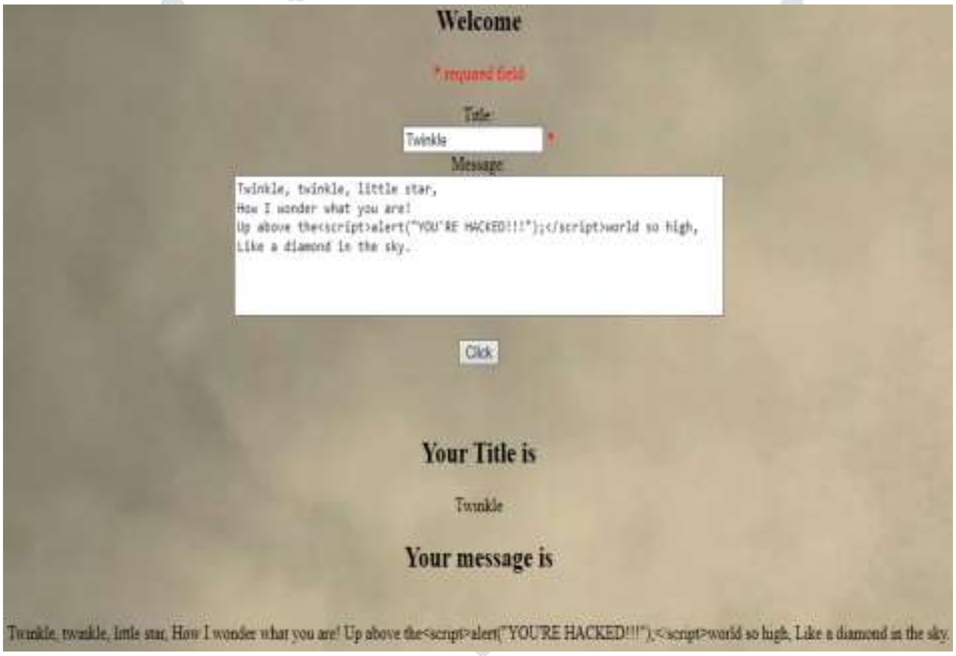
```
Twinkle, twinkle, little star,
How I wonder what you are!
Up above the<script>alert("YOU'RE HACKED!!!");</script>world so high,
Like a diamond in the sky.
```

Your Title is

Your message is

Fig.7 Web page filled with malicious script after applying proposed preventive method

Press the “Click” button and observe.



Welcome

* required field

Title:

Twinkle *

Message:

```
Twinkle, twinkle, little star,
How I wonder what you are!
Up above the<script>alert("YOU'RE HACKED!!!");</script>world so high,
Like a diamond in the sky.
```

Your Title is

Twinkle

Your message is

Twinkle, twinkle, little star, How I wonder what you are! Up above the<script>alert("YOU'RE HACKED!!!");<script>world so high, Like a diamond in the sky.

Fig.8 The script gets displayed in text since it wasn’t executed

The script does nothing. The script is displayed with the normal text.

VI. CONCLUSION

This paper dealt with the cross-site scripting attack and need for web application security. Some measures to forestall is also discussed. Cross-site scripting is named one of the most critical web application threat by OWASP. This paper presented the stored attack of cross-site scripting using DVWA. In near future, a software tool can be developed which automatically detects and prevents the malicious script from running in the web application. A browser extension can also be developed to forbid the execution of script in web application.

VII. REFERENCES

[1] Twana Assad Taha, Murat Karabatak, "A proposed approach for preventing cross site scripting", 6th International Symposium on Digital Forensic and Security (ISDFS), 2018.

[2] Ding Lan, Wu ShuTing, Ye Xing, Zhang Wei, "Analysis and prevention for cross site scripting attack based on encoding", IEEE 4th International Conference on Electronics Information and Emergency Communication, 2013.

[3] About OWASP. https://www.owasp.org/index.php/About_The_Open_Web_Application_Security_Project.

[4] Rafay Baloch, "Ethical Hacking And Penetration Testing Guide", CRC Press, 2015.

[5] Wille L. Pritchett, David De Smet, "Kali Linux Cookbook", Packt Publishing, 2013.

[6] Juned Ahmed Ansari, "Web Penetration Testing with Kali Linux", Packt Publishing, 2015.

[7] Gilberto Nájera-Gutiérrez, "Kali Linux Web Penetration Testing Cookbook", Packt Publishing, 2016.

[8] Nipun Jaswal, "Mastering Metasploit", Packt Publishing, 2014.

[9] David Kennedy, Jim O'Gorman, Devon Kearns, Mati Aharoni, "Metasploit The Penetration Tester's Guide", No Starch Press, 2011.

[10] Alan Forbes, "The Joy of PHP – A Beginner's Guide", Plum Island Publishing LLC, 2012.