2013 International Conference on Computational and Information Sciences

Research on Developing an Attack and Defense Lab Environment for Cross Site Scripting Education in Higher Vocational Colleges

Huangcun Zeng
Department of Information Technology
Guangdong Teachers' College of Foreign Language and Arts
Guangzhou, China
zenghc@gtcfla.net

***Abstract*—Cross site scripting attacks are the most common network attacks nowadays. In order to let the students in higher vocational colleges master the attack and defense skills of cross site scripting, it's essential to let them have the opportunity to truly practice the attacking behaviors as hackers do, and to practice the defense countermeasures as web site administrators do. In this paper, we have developed an attacking and defense lab environment for cross site scripting education in higher vocational colleges. Our experience shows that by using this lab environment, students can better understand what cross site scripting vulnerabilities are, why they occur, how to exploit them and how to fix them. We suggest that a lot more researches need to be done in the area of developing attacking and defense lab environments in order to improve the outcomes of network security education in higher vocational colleges.**

Keywords-cross site scriping; lab environment; higher vocational college; network security; attack; defense

I. INTRODUCTION

Cross site scripting (XSS) attacks are the most common network attacks nowadays. These are attacks where improper input validation on a web site allows malicious users to inject code into the web site, which later is executed in a visitor's browser [1].

It is widely known that learning-by-doing can significantly enhance student's learning in many disciplines, including computer security education [2]. Pedagogical research has shown that effective laboratory exercises are critically important to the success of the courses like network security [3].

Du has developed a lab environment that can be used to practice some vulnerability attacks including XSS attacks [4]. Yet in general not much research has been done on developing lab environments to facilitate network security education.

In order to let the students in higher vocational colleges master the attacking and defense skills of XSS, it's essential to let them have the opportunity to truly practice the attacking behaviors as hackers do, and to practice the defense countermeasures as website administrators do.

In this paper, we have developed an attacking and defense lab environment for XSS education in higher vocational colleges.

Our experience shows that by using this lab environment, students can better understand what XSS vulnerabilities are, why they occur, how to exploit them and how to fix them. We suggest that a lot more researches need to be done in the area of developing attacking and defense lab environments in order to improve the outcomes of network security education in higher vocational colleges.

The paper is structured as follows: in Section II we describe the development of an XSS attacking and defense lab environment, in Section III we show how to utilize this lab environment to let students practice XSS attacks and defense against XSS attacks, in Section IV we conclude our paper.

II. DEVELOPMENT OF AN XSS ATTACKING AND DEFENSE LAB ENVIRONMENT

In this section, we describe our development of an XSS attacking and defense lab environment.

In this XSS attacking and defense lab environment, we should have a web server and a hacker computer. There will be a web site that has XSS vulnerabilities residing on the web server. The hacker computer should try to exploit XSS vulnerabilities present in the web site. And finally we will take countermeasures to protect the web site from XSS attacks.

We have installed two virtual machines using VMware WorkStation, one is Windows Server 2003, and the other is Windows XP. Windows Server 2003 mimics the web server, whose IP address is 192.168.248.132, Windows XP mimics the hacker computer, whose IP address is 192.168.248.133.

The design principles for the vulnerable web site are:

1) It has XSS vulnerabilities, thus allowing practice of XSS attacks and defense.

2) It should be as simple as possible. Since students in higher vocational colleges are not as proficient as university students in logical thinking, if the website is too complex and contains too many irrelevant technical details, it will hinder the understanding of higher vocational college students.

Following these design principles, we have developed an extremely simple guest book web application, and have deployed it on the virtual Windows Server 2003.

978-0-7695-5004-6/13 $26.00 © 2013 IEEE
DOI 10.1109/ICCIS.2013.515

The guest book web application has adopted Accss+Asp as its main technologies. The reason for this choice is that Accss+Asp are among the simplest dynamic web technologies, and since one of our design principles is that the web site should be as simple as possible. However, we can easily adapt our lab environment to adopting other technologies such as Sql Server+ASP.Net.

The guest book application needs to access an Access database, which includes two tables: tblUsers and tblMessages. The table tblUsers records the user names and passwords of website users, and tblMessages records the information of posted messages. Their structures are shown in Tables I and II.

TABLE I. THE STRUCTURE OF TBLUSERS

Column Name	Data Type	Length	Meaning
username	text	50	Username for login
password	text	50	Password for login

TABLE II. THE STRUCTURE OF TBLMESSAGES

Column Name	Data Type	Length	Meaning
ID	Int	4	ID of the posted message
poster	text	50	Poster
content	text	200	Content of the posted message
postDate	DateTime		Time at which the message was posted

The simple guest book application includes the following six web pages: conn.asp, login.asp, processLogin.asp, index.asp, post.asp and logout.asp. The web page conn.asp stores information concerning connecting the Access database; login.asp provides an interface for an user to log in the application; procssLogin.asp decides whether an user is authenticated, and if so redirects an user to index.asp, if not redirects an user to login.asp; index.asp is the main web page which shows all the posted messages; post.asp provides the functionality for an user to post a new message; and finally logout.asp lets an user log out.

In post.asp, we have purposely allows the user to post any messages he/she wants to make this web application vulnerable to XSS attacks.

In the following we show the detailed source codes of the above mentioned web pages.

The source code of the web page conn.asp:

```
<%
Const Db = "guestbook.mdb"
dim ConnStr
ConnStr = "Provider = Microsoft.Jet.OLEDB.4.0;Data Source =" & Server.MapPath(Db)
%>
```

The source code of login.asp:

```
<form action="processLogin.asp" method="post">
User Name : <input name="txtUserName" type="text"><br>
Password : <input name="txtPassword" type="password"><br>
<input name="btnLogin" type="submit" value="Login">
</form>
```

The source code of procssLogin.asp:

```
<!--#include file="conn.asp"-->
<%
dim conn, strSql, rs, userName, password
set conn = Server.CreateObject("ADODB.Connection")
conn.open ConnStr
userName=request("txtUserName")
password=request("txtPassword")
strSql="select * from tblUser where username='" & userName & "' and password='" & password & "'"
set rs=conn.execute(strSql)
if not (rs.bof and rs.eof) then
        response.cookies("userName")=userName
        response.cookies("password")=password
        rs.close
        conn.close
        set rs=nothing
        set conn=nothing
        response.redirect("index.asp")
else
        rs.close
        conn.close
        set rs=nothing
        set conn=nothing
        response.redirect("login.asp")
end if
%>
```

The source code of index.asp:

```
<!--#include file="conn.asp"-->
<%
if request.cookies("userName")="" or request.cookies("password")="" then
        response.redirect("login.asp")
else
        dim conn, strSql, rs
        set conn = Server.CreateObject("ADODB.Connection")
        conn.open ConnStr
        strSql="select * from tblMessages order by postDate desc"
        set rs=conn.execute(strSql)
        response.write("<html><body><table><tr><td>")
        response.Write("<a href='post.asp'>post a message</a>")
        response.Write("</td><td>")
        response.Write("<a href='logout.asp'>logout</a>")
        response.Write("</td></tr>"")
        while not rs.eof
                response.write("<tr>")
                response.write("posted by " & rs("poster") & " at time:" & rs("postDate"))
```

```
                response.write("</tr><tr>")
                response.write( rs("content"))
                response.write("</tr><tr><br></tr>")
                rs.moveNext
        wend
        response.write("</table></body></html>")
        rs.close
        conn.close
        set rs=nothing
        set conn=nothing
    end if
    %>
```

The source code of post.asp:

```
<!--#include file="conn.asp"-->
<%
if request.cookies("userName")="" or request.cookies("password")="" then
        response.redirect("login.asp")
end if
if request.form("post")="post" then
        dim conn, strSql, rs
        set conn = Server.CreateObject("ADODB.Connection")
        conn.open ConnStr
        strSql="select * from tblMessages where ID=0"
        set rs=Server.CreateObject("ADODB.RecordSet")
        rs.open strSql,conn,1,3
        rs.addnew
        rs("poster")=request.cookies("userName")
        rs("content")=request.form("txtContent")
        rs.update
        rs.close
        set rs=nothing
        response.redirect("index.asp")
end if
%>
<form action="post.asp" method="post">
conent : <textarea name="txtContent" cols=50 rows=10></textarea><br>
<input name="btnPost" type="submit" value="post">
<input type="hidden" name="post" value="post">
<a href="index.asp">return</a>
</form>
```

The source code of logout.asp:

```
<%
response.cookies("userName")=""
response.cookies("password")=""
response.redirect("login.asp")
%>
```

From the above source codes, we should see that our guest book web application is extremely simple. It only includes the essential codes, and ignores other non relevant technical details, which makes it easier for higher vocational college students to understand, analyze and master the essentials of XSS attacks and defense.

III. USE OF THE LAB ENVIRONMENT TO FACILITATE EDUCATION OF XSS ATTACKS AND DEFENSE

In this section, we will describe how we will teach XSS attacks and defense utilizing the lab environment described in Section II.

First we will ask the students to access the guest book application. Then we will analyze the web page post.asp whose functionality is to post a message. This page has XSS vulnerabilities since it allows an user to post any messages he/she wants.

Then we will instruct the students to post the following message to test whether the web site has XSS vulnerabilities:

```
<script>alert('XSS');</script>
```

If the web site has XSS vulnerabilities, it won't validate user's input, and the above script will be executed when browsing the posted message, thus a dialog showing "XSS" will pop up. Otherwise, user's input is validated, the above script will be forbidden to execute, so no pop up dialog will be shown.

And then in order to further demonstrate XSS attacks, we will instruct the students to steal cookies from the web site.

We will instruct the students to construct a dynamic web site on their virtual Windows XP. This dynamic web site includes a cookie stealing web page named getCookie.asp and a plain text file cookie.txt which will store the cookies stolen from the victim web site. The source code of getCookie.asp is as follows:

```
<%
cookie=request("cookie")
set fs=Server.CreateObject("Scripting.FileSystemObject")
set file=fs.OpenTextFile(Server.MapPath("cookie.txt"),8,Ture)
file.writeline("cookie:" & cookie)
file.close
set file=nothing
set fs=nothing
%>
```

The functionality of this script is to store cookies stolen from the victim web site in the file cookie.txt.

And we will instruct the students to post a message shown as follows:

```
<script>
img=new Image();
img.src="http://192.168.248.133/getCookie.asp?cookie=" + document.cookie;
</script>
```

Notice that 192.168.248.133 is the IP address of the virtual Windows XP mimicking a hacker computer. When the above posted message is browsed by other users, the browser will access the getCookie.asp on Windows XP and send cookies to this page without user notice.

By mimicking the hacker's behaviors of stealing the cookies from the victim website, the students get a clearer understanding of XSS attacks. And it's now easier to teach them how to protect against XSS attacks. The key is to validate user's input and forbids script from being executed.

We will instruct the students to edit an anti XSS attack web page named antiXSS.asp. The source code of antiXSS.asp is as follows:

```
<%
function encode(str)
    if isnull(str) or trim(str)="" then
        encode=""
    else
        str=replace(str,">","&gt;")
        str=replace(str,"<","&lt;")
        encode=str
    end if
end function
%>
```

The function encode replaces symbols "<" and ">", thus scripts won't be executed anymore.

And in order to make XSS.asp in effect, we instruct students to include this page in index.asp and instead of simply writing "response.write(rs("content"))" in index.asp, we use "response.write(encode(rs("content")))". After taking these countermeasures, we will ask the students to check that they can no longer attack the victim web site using above mentioned methods.

IV. DISCUSSION

Learning-by-doing can significantly enhance student's learning in computer security education [2]. Effective laboratory exercises are critically important to the success of the courses like network security [3].

In our previous research, we have developed lab environments for SQL injection attacks and defense. Our experience has shown that using these simple lab environments can facilitate SQL injection education [5, 6].

In order to teach higher vocational college students XSS attacks and defense, we have designed and implemented an extremely simple guest book web application. We use a virtual machine as a web server and deploy the guest book web application on it. We use another virtual machine as a hacker machine, and let the students use this machine to mimic hacker attacks on the target web site by exploiting XSS vulnerabilities. And later we ask students to take countermeasures to protect the web site from XSS attacks.

Our motivation of designing and implementing such a simple web application is we believe that students learn best by doing. By using our simple web application, the students can practice how to exploit XSS vulnerabilities from the victim web site, and how to protect the web site from XSS attacks. After these practices, they will have a better understanding of what XSS vulnerabilities are, why they occur, how to exploit them and finally how to fix them.

Our web application is extremely simple and ignores many irrelevant technical details. The reason why we do this is that we think that higher vocational college students are not as proficient as university students in logical thinking, so that too many technical details will hinder their comprehension. On the other hand, our web application only includes essential source codes for XSS attacks and defense education and ignores other non essential technical details. In so doing, the students can more easily understand and master the skills.

The reason why we didn't use the full fledged web application such as DvBBS that have XSS vulnerabilities is the same as the reason that we stated above. These full fledged web applications are too complex and have many technical details that are not relevant to XSS attacks and defense.

We have used this simple web application in our network security class. And the experience shows that using this simple web application improves the outcomes of the XSS attacking and defense education.

We suggest that in order to improve the outcomes of network security education in higher vocational colleges, the teachers should develop a lot more lab environments that can be used by students to practice relevant attacks and defense. Only by practicing attacks as hackers do, and then practicing defense as web site administrators do, can students truly understand the attacking and defense skills. Yet we haven't seen much research done in this area, so we think there is still much work that needs to be done.

In this paper, we have demonstrated how a higher vocational college teacher can develop a simple web application that has XSS vulnerabilities, and how to use this web application to let students practice XSS attacks and defense. Though the web application is simple, we believe that it will be beneficial to cultivate the attacking and defense skill of higher vocational college students.

Our future work is to design and implement more lab environments that will facilitate the education of network security in higher vocational colleges.

REFERENCES

[1] M. Goodrich, R. Tamassia, Introduction to Computer Security. Beijing: Tsinghua University Press, 2013, pp.357.

[2] W. Du, K. Jayaraman, and N. Gaubatz, "Enhancing Security Education with Hands-on Laboratory Exercises,".in Proceedings of the 5th Annual Symposium on Information Assurance (ASIA '10). June 16-17, 2010, Albany, New York.

[3] W. Du, M. Shang, and H. Xu, "A Novel Approach for Computer Security Education using Minix Instructional Operating System," Computer & Security, Vol. 25, 2006, pp. 190-200.

[4] W. Du, "The SEED Project: Providing Hands-on Lab Exercises for Computer Security Education," IEEE Security and Privacy Magazine, September/October, 2011. Invited paper.

[5] H. Zeng, "Research on the Development and Use of an Lab Environment for SQL Injection Education in Higher Vocational Colleges," Journal of Vocational Education Research, in press.

[6] H. Zeng, "Research on the Development and Use of an Lab Environment for Logging on a Web Site Using SQL Injection Attacks," Journal of Vocational Education Research, in press.