CHAPTER 16

Preventing Cross-Site Scripting Attacks

Cross-site scripting involves the injection of malicious scripts into web pages, where they can be used to gain access to user's systems or to sensitive information. The Cisco ACE Web Application Firewall provides rules that inspect messages for JavaScript, ECMAScript, VBScript and other types of code artifacts that could indicate a cross-site scripting attack.

In this chapter, you'll demonstrate a cross-site scripting vulnerability in the Poison Oak sample application, and then configure the Firewall to prevent such attacks.

Simulating a Cross-Site Scripting Attack

In this section, you'll simulate a cross-site scripting attack by using your browser to inject a JavaScript command into an HTTP argument to the Poison Oak sample application.

Before trying this attack, you may need to clear your browser's cache to ensure that you do not view a cached response.

Step 1 From the Poison Oak sample application home page, click the **Developers** link.

Step 2 Click the **Retrieve Quote** link.

A page appears detailing the interface for retrieveQuote, one of the SOAP operations available from the Poison Oak sample application. You can view the interface for each operation in the application by passing the operation name as a URL argument to the details page, as evident in the browser's address bar, details.do?op=retrieveQuote

Step 3 In the browser's navigation bar, replace the op argument value, retrieveQuote, with the following text:

<script>alert('XSS attack in progress!')</script>

When you submit the request, a dialog box appears with the text "XSS attack in progress." It can be inferred from the success of this attack that the application does not sanitize input to protect against script attacks.

In the next section, you'll configure the **Cross Site Scripting** firewall rule to protect against cross-site scripting vulnerability.

Blocking XSS Attacks

To configure the ACE Web Application Firewall policy to protect against cross-site scripting attack, follow these steps:

Step 1 Click the **Profiles** link in the Manager navigation menu, and then click on the name of the profile you created, **Poison Oak Traffic Validation**.

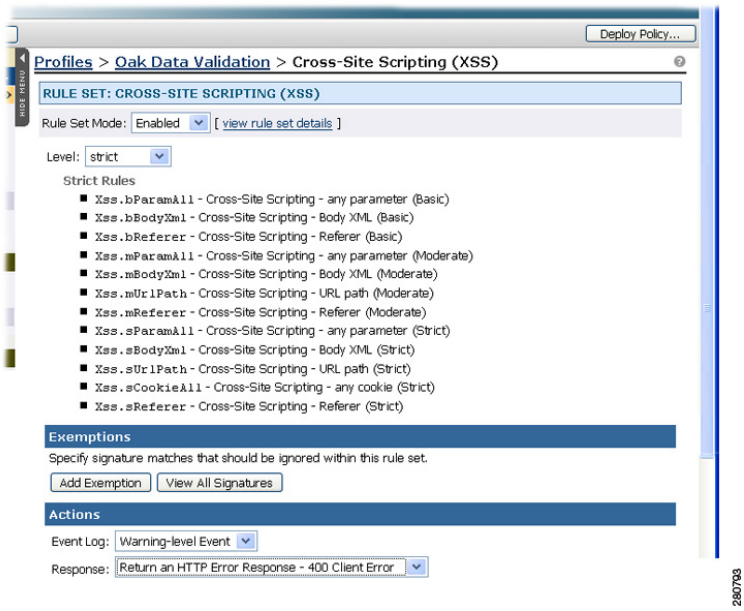
Notice that the Cross-Site Scripting (XSS) rule set is currently disabled in this profile. The Cross-Site Scripting (XSS) rule set is listed with the Message Inspection rules under the Firewall Configuration settings.

Step 2 Click the **edit** link next to the Cross Site Scripting rule.

Step 3 In the Cross-Site Scripting (XSS) rules page, choose **Enabled** from the **Rule Set Mode** menu.

The settings for the rule set appear, as shown in Figure 16-1.

Figure 16-1 Cross-Site Scripting Rule editor



Step 4 For the **Level** menu, keep the default value, **strict**.

Step 5 In the **Actions** section, choose the **Return A Custom HTTP Error Response** item from the **Response** menu.

Step 6 Type **XSS Error blocked by Poison Oak Traffic Validation** in the **Body** field. This text will appear as the response body for requests that trigger the rule.

Step 7 Click **Save Changes**.

The profile page should now indicate that the cross site scripting rule is enabled for this profile.

Step 8 Deploy the policy to have the changes take effect.

After deploying your changes, you can test cross site scripting security.

Testing XSS Attack Prevention

To try out the new cross site scripting security rule, follow these steps:

Step 1 In your browser, go to the Poison Oak home page through the ACE Web Application Firewall.

Step 2 On the home page, click the **Developers** link.

Step 3 Click the **Retrieve Quote** link.

Step 4 In the browser's navigation bar, again replace "retrieveQuote" with the following text:

<script>alert("XSS attack in progress!")</script>

This time, the injected JavaScript command never reaches the backend service—the ACE Web Application Firewall blocks the command and returns your custom error message.

For more information on the incident, check the event log and incident report in the Manager web console.