ACE Web Application Firewall Getting Started Guide

Software Release 6.0

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Text Part Number: OL-16663-01

The Cisco ACE Web Application Firewall is an application oriented networking product.

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

ACE Web Application Firewall Getting Started Guide

CONTENTS

Preface

This preface contains the following major sections:

Audience

This guide provides hands-on instruction that introduces the basic tasks you'll need to perform in order to use and administer the ACE Web Application Firewall. It is intended for users and administrators of the ACE Web Application Firewall. It assumes that you are familiar with networking, security, and Web application technologies.

Organization

This guide includes the following sections:

Part I: First Steps	Description
Chapter 1, "About the ACE Web Application Firewall"	Introduces the ACE Web Application Firewall and describes how to use it to implement web application security.
Chapter 2, "Before Starting"	Describes what you need to know and other prerequisites for following the steps in this guide.
Chapter 3, "Performing the Initial Setup"	Lists the procedures for installing the appliance on the network, and configuring its initial operating settings.
Chapter 4, "Accessing the Manager Web Console"	Describes how to log into the Manager web console, the graphical user interface for configuring and monitoring the system.
Chapter 5, "Virtualizing a Web Application"	Describes how to enable traffic routing through the Firewall by adding a virtual web application definition to the policy.

Part I: First Steps	Description
Chapter 6, "Deploying and Testing the Policy"	Provides instructions for deploying the policy from the Manager to the Firewall, and testing your first virtual web application.
Chapter 7, "Working with Firewall Profiles"	Describes how to use profiles to define traffic processing and security settings for a particular class of traffic.
Chapter 8, "Using the Logs"	Shows how to use the logging tools to develop, troubleshoot, and monitor your implementation.
Chapter 9, "Creating Request Match Filters"	Demonstrates the request matching criteria you can configure at the consumer interface for applications exposed at the ACE Web Application Firewall.
Chapter 10, "Monitor Mode"	Describes how to set up the ACE Web Application Firewall to apply message inspection rules without blocking traffic.
Chapter 11, "Fine-Tuning Firewall Rules"	Describes how to use modifiers to fine-tune traffic processing behavior.
Chapter 12, "Securing HTTP Cookies"	Provides an overview of the cookie security features in the profile.
Chapter 13, "Rewriting Response Content"	Details how to configure response content rewriting.
Chapter 14, "HTTP Error Response Mapping"	Provides instructions on mapping error responses from the backend to custom responses at the Firewall.
Chapter 15, "Server Header Masking"	Describes HTTP header rewriting capabilities of the system.

Related Documentation

For additional information on the Cisco ACE Web Application Firewall software, see the following documentation:

- *Cisco ACE Web Application Firewall Administration Guide*
- *Cisco ACE Web Application Firewall User Guide*

The following sections provide sources for obtaining documentation from Cisco Systems.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

Conventions

This document uses the following conventions:

Convention	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
`courier` font	Terminal sessions and information the system displays appear in `courier` font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means *reader take note*.

Caution Means *reader be careful*. In this situation, you might perform an action that could result in equipment damage or loss of data.

Warning **Means *reader be warned*. In this situation, you might perform an action that could result in bodily injury.**

Notices

The following notices pertain to this software license.

OpenSSL/Open SSL Project

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

This product includes software written by Tim Hudson (tjh@cryptsoft.com).

License Issues

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License:

Copyright © 1998-2007 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)”.
4. The names “OpenSSL Toolkit” and “OpenSSL Project” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5. Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment:

   “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)”.

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT “AS IS”' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License:

Copyright © 1995-1998 Eric Young (eay@cryptsoft.com). All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).

The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

   "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)".

   The word 'cryptographic' can be left out if the routines from the library being used are not cryptography-related.
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)".

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The license and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution license [including the GNU Public License].

PART I

Setup

This part introduces concepts that serve as the foundation for setting up and using the Cisco ACE Web Application Firewall. It describes the first steps in implementing the ACE Web Application Firewall—setting it up in your network environment.

The following topics are covered:

CHAPTER 1

About the ACE Web Application Firewall

Web-based attacks are among the most rapidly proliferating threats to enterprise data systems. Successful introduction of malformed data to a web application may allow a relatively unsophisticated attacker to compromise a system, possibly putting credit card numbers, medical records, and other sensitive data at risk.

The Cisco ACE Web Application Firewall, a member of the Cisco Application Control Engine (ACE) family of products, helps to ensure the security of backend applications and resources. The Cisco ACE Web Application Firewall provides protection against many types of web application attacks, such as cross-site scripting (XSS), SQL injection, cross-site request forgery (CSRF), buffer overflow, cookie tampering, and more. It it designed to help you meet a wide range of web application security requirements, including those specified by the *Payment Card Industry Data Security Standard (PCI DSS) Version 1.1*.

While traditional firewalls operate primarily at the level of the message packet header, the Cisco ACE Web Application Firewall can look inside the message contents, thereby identifying and preventing harmful content-based attacks. It can also identify and mask sensitive content in responses (such as credit card numbers).

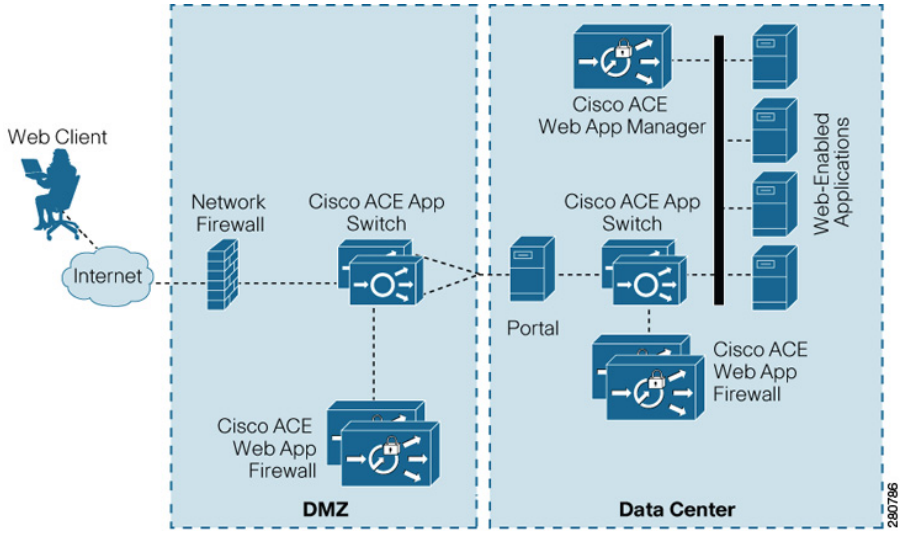
Because web-based attacks are ever-evolving, the ACE Web Application Firewall allows you to supplement its built-in threat prevention capabilities with your own custom rules and signatures. It provides extensive monitoring facilities that log and analyze server traffic in real time, enabling you to identify emerging or site-specific threats and quickly adapt the system to meet those threats.

Where it Fits in the Network

The ACE Web Application Firewall acts as a full reverse proxy. From the viewpoint of the backend application, the ACE Web Application Firewall appears to be the source of client requests. (Note that the ACE Web Application Firewall can forward information to the backend about the actual client, such as the client's IP address.)

The ACE Web Application Firewall generally operates in close proximity to the backend infrastructure, where it proxies HTTP and HTTPS traffic between external users and the backend systems. As shown in Figure 1-1, it can operate in the organization's network DMZ or in the protected network.

Figure 1-1 Deployment Topology



Operating in the DMZ, the ACE Web Application Firewall can validate traffic before it enters the protected network. Operating within the data center, it can safeguard applications from misuse from either inside or outside the organization, provide centralized monitoring, validate application input, and more.

The scale of a firewall deployment can be increased by clustering multiple ACE Web Application Firewall appliances behind a load-balancing device. Wherever deployed, the ACE Web Application Firewall provides application-level traffic security and control, ensuring uniform enforcement of security policies across network applications.

System Components

A Cisco ACE Web Application Firewall deployment consists of the following:

- ACE Web Application Firewall, the enterprise-class appliance for securing and managing application network traffic.
- ACE Web Application Firewall Manager, the administration server for the system. The ACE Web Application Firewall Manager serves the web-based graphical user interface used to configure and monitor the system.

The ACE appliances are delivered in a full-size, rack-mountable chassis that is suitable for production environments. A single appliance can operate as both Manager and Firewall (in what is called the standalone mode). However, this configuration is intended only for policy development or evaluation purposes—it is not recommended for production deployment.

Typically, a production deployment consists of an ACE Web Application Firewall Manager and multiple ACE Web Application Firewalls, which are organized by clusters.

Understanding Security Policies

The ACE Web Application Firewall processes traffic as specified by its policy. The policy is the complete set of rules and processing instructions applied by the ACE Web Application Firewall. Implementing the system primarily involves developing the policy in the ACE Web Application Firewall Manager web console. The web console includes tools and wizards that ease the task of securing web applications.

The version of the policy under development in the Manager web console is called the *working policy*. To enforce a policy on network traffic, you use the web console to transmit the policy to the ACE Web Application Firewall, a process known as deploying the policy. The policy actively running on the Firewall is the *active policy*.

In developing the policy, you may choose to configure:

- validation of the content, arguments, or other properties of the message
- screening of response content
- verification of digital certificates that identify message senders
- verification and generation of digital signatures to protect cookies
- encryption and decryption of cookies

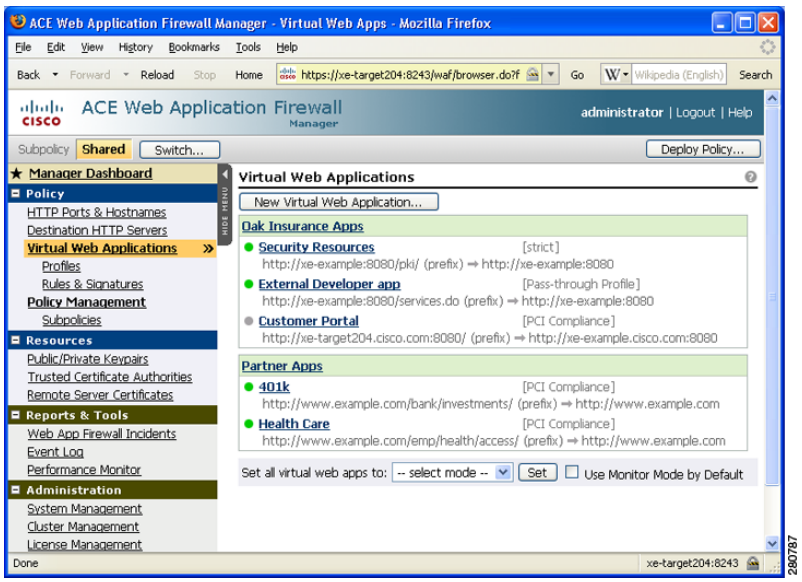
The traffic processing measures are applied to a class of network traffic through a policy object called a *virtual web application*, as described in the next section.

Virtualizing Web Applications

A virtual web application encapsulates settings for a particular backend application. It specifies the consumer interface for the application at the Firewall, the connection settings for the backend application, and the rules to be applied to traffic for the application.

As shown in Figure 1-2, a policy can contain numerous virtual web applications.

Figure 1-2 Components of an ACE Web Application Firewall Policy



Requests are matched to a virtual web application based on its consumer interface definition. The consumer interface can select requests based on various properties of the request, such as the request URL, POST or GET parameter values, or HTTP headers in the request. This allows you to apply specialized processing or security measures to traffic for a given web application at a highly granular level.

The virtual web application processes selected messages as determined by a *profile*. A profile defines a particular configuration of the web application security rules and processing features. The ACE Web Application Firewall includes two built-in firewall profiles:

- The *Pass-through* profile provides all firewall options in a disabled state; that is, all traffic that matches the request filter expression passes through to the backend service uninspected and unmodified.
- The *PCI Compliance* profile provides preconfigured rule settings to help meet the requirements of the *Payment Card Industry Data Security Standard Version 1.1*.

You cannot modify the settings of built-in profiles—they are meant to be applied directly or used as a starting point for the customized profiles you create. Each profile specifies a complete set of firewall rules, including each feature's enabled/disabled status and the values of feature-specific settings.

About this Document

This document provides tutorial-style instruction for setting up and using the ACE Web Application Firewall against a backend application. It provides hands-on experience with using the ACE Web Application Firewall Manager to develop a security policy and deploying it to the Firewall. It guides you through the process of setting up a virtual web application, assigning a firewall profile to it, and testing its effect on live traffic. It also describes how to return customized error codes, mask server headers, prevent SQL injection attack, secure cookies, and prevent cross-site scripting attack.

This guide assumes that you are familiar with networking, security, and web applications technologies. For general information, see the *Cisco ACE Web Application User Guide*. From the web console, you can view information on web console settings by clicking the help link on the console page.

CHAPTER 2

Before Starting

To follow the steps in this guide, you'll need access to an ACE Web Application Firewall and Manager installation. This guide provides an overview of how to install the appliance on your network. For complete information on installing the ACE Web Application Firewall and Manager, see the *Cisco ACE Web Application Firewall Administration Guide*. The following sections describe what you'll need in order to perform the appliance set up and other policy configuration steps described in this guide.

Set Up Requirements

Before you can start configuring a web application security policy, the ACE Web Application Firewall appliance needs to be installed on your network, as described in Chapter 3, "Performing the Initial Setup." To perform this task, you will need the following information:

- The IP address of the default IP gateway for the network
- The IP address of the primary DNS server for the network
- A hostname for the ACE Web Application Firewall registered with the local network's DNS server (recommended)
- The password for the root account on the ACE Web Application Firewall appliance

  If this is a new installation and you do not know the root password, please contact your support representative.
- A static IP address for the ACE Web Application Firewall appliance

  You only need one IP address to run the ACE Web Application Firewall and Manager on a single appliance chassis (the configuration described in this guide). However, you will need two IP addresses to run the Manager and Firewall on separate appliances.

Tools for Generating Traffic

Configuring and testing the policy as described in this guide requires several third-party tools. After you configure the ACE Web Application Firewall, you'll want to send test requests to it. You can use your preferred web browser to send the requests described in this book.

Several examples involve HTTP cookie editing. To edit cookies, you can use the "Add N Edit Cookies" add-on for Firefox (https://addons.mozilla.org/en-US/firefox/addon/573). For Internet Explorer, tools such as IECookiesView (http://www.nirsoft.net/utils/internet_explorer_cookies_view.html) or CookieEditor (http://www.proxoft.com/CookieEditor.asp) are available.

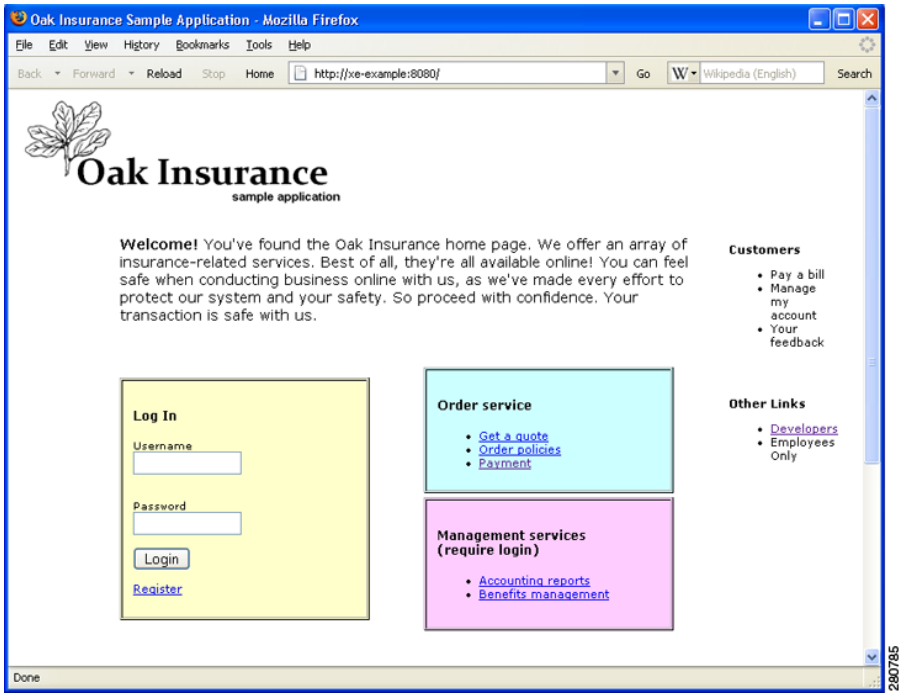
You'll also need a tool that allows you to view HTTP headers. The examples in this book use the Live HTTP Headers add-on for Firefox. For Internet Explorer, the Fiddler tool is available from Microsoft (http://www.fiddlertool.com/fiddler/).

About the Sample Application

This guide describes how to configure the ACE Web Application Firewall for a particular backend application. The sample application used in this guide (Figure 2-1) is the web portal for a fictional insurance company, Poison Oak Insurance Company. The application is designed for Poison Oak's customers, partners, and internal employees. For instance, customers can use the application to request an insurance quote, submit an order, or pay a bill. Employees can retrieve business accounting reports or benefits information.

Unfortunately for Poison Oak's imaginary users, the web application is riddled with security vulnerabilities, including vulnerabilities to buffer overflow, SQL injection, and cross-site scripting attack. The vulnerabilities are used in this guide to demonstrate the capabilities of the ACE Web Application Firewall. Following the steps in this guide, you will configure the ACE Web Application Firewall to secure the Poison Oak web application and its users.

Figure 2-1 The Poison Oak Insurance home page



If you have access to the Poison Oak sample application, you will be able to follow the steps described in this guide closely. However, you can also follow these steps with your own application or using an application that, like the Poison Oak Sample application, is intended to demonstrate web vulnerabilities, several of which are available on the web.

CHAPTER 3

Performing the Initial Setup

The first step in implementing the ACE Web Application Firewall system is installing the appliance on your network. This involves configuring its basic network settings, including its IP address, default gateway, name servers, and so on.

You configure network settings for the ACE Web Application Firewall and Manager from the appliance's operating system environment. To access the environment, connect a monitor and USB keyboard (optionally a KVM switch) to the appliance directly. You can also connect by serial connection from a computer with VT100-compatible terminal emulation software or from a dumb terminal (as described in "Accessing the Appliance by Serial Connection" section on page 3-10). However, note that the appliance writes boot up messages to KVM output by default.

After configuring the network settings for the appliance, you will be able to access its operating system environment remotely from a secure shell (ssh) client.

Preparing the Appliance

If not done already, you must first physically prepare the appliance for installation, as follows:

Step 1 Remove the appliance from its packaging.

If evaluating the appliance, be sure to keep the packing materials for use when returning the system. The packaging usually includes a return shipping label for use in returning the appliance.

Step 2 Connect a monitor and keyboard to the back of the appliance. (If you would rather access the appliance shell environment by serial connection from another computer, see the "Accessing the Appliance by Serial Connection" section on page 3-10.")

Step 3 Plug a power cable into the power supply port on the appliance.

Step 4 Attach an Ethernet cable to a network interface port on the back panel of the appliance.

Rack-mounted appliances may have as many as five Ethernet interface ports. Connect the cable to any of the connectors except the Integrated Lights-Out connector, which is on the top-left side of the back panel. Make a note of which interface you are using.

Step 5 Turn on the appliance by pressing the power button on the front panel.

The system starts up. When the startup procedure finishes, a login prompt appears on the monitor.

Accessing the Appliance by Serial Connection

Instead of a direct KVM connection, you can connect to the appliance using a laptop or personal computer connected by serial cable to the DB-9-type serial connector.

To access the appliance by serial connection, the laptop or personal computer must have VT-100-compatible terminal emulation software, such as Microsoft HyperTerminal. Configure your serial terminal or terminal-emulation software to use the following settings for the connection:

- Bits per second: 9600
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow Control: None

By default, boot-up messages of the appliance are written to KVM output only, not to serial output. For information on changing this behavior, see the *Cisco ACE Web Application Firewall Administration Guide*.

Performing the Initial Configuration

After connecting to the appliance, configure its basic settings as follows:

Step 1 At the appliance login prompt, log in as `root` user with the password provided to you.

The factory default password for the root user is *swordfish*.

Step 2 When logging in with the default password, you will be prompted to change the password. To bypass the password change prompt at this time, press the Esc key. To change the password:

1. Click **OK** or press enter.
2. In the **Please enter the new root password** screen, type a new password and click **OK**.
3. Confirm the password when prompted.

The main menu of the appliance administration interface appears, as shown in Figure 3-1.

Figure 3-1 Main Menu of the Shell

Step 3 Choose the second item, **Network Configuration**, and click **OK**.

Step 4 In the **Network Configuration** screen, configure the hostname for the appliance as follows:

1. Select the first item, **Hostname**, and click **OK**.
2. In the hostname screen, enter the fully qualified hostname of the ACE Web Application Firewall (such as acewaf.example.com). When finished, click **OK**.

Step 5 In the **Network Configuration** menu, select **IP Gateway** and type the IP address of the default gateway of the subnetwork to which the appliance network interface is connected. When finished, click **OK**.

Step 6 Select **Name Servers** and enter the IP addresses of the DNS servers in your network. To specify more than one DNS server, enter their IP addresses separated by a space. The appliance will query the first server in the list, and query others in the order they appear if a name server is unresponsive. When finished, click **OK**.

Step 7 By default, the physical network interfaces on the appliance are disabled. In the **Network Configuration** menu, choose the interface to which you have plugged in the network cable by its identifier, such as **Interface eth0**, and configure the interface as follows:

1. Select the first item on the network interface menu, **enabled**.
2. Type the IP address for the appliance and click **OK**.
3. Enter the network mask (netmask) for the IP address.
4. At the next prompt, choose an Ethernet speed for the interface and click **OK**.
5. In the **Edit Static Routes** screen, choose **Accept settings** and click **OK** or **Cancel** to start over.
6. The settings appear on the commit changes screen. Choose **Yes** to save the new settings and restart the network interfaces.

Step 8 In the **Network Configuration** menu, select **Test Network Settings**. The ACE Web Application Firewall performs a few basic network tests of your settings and reports their results.

Step 9 In the **Network Configuration** menu, choose **Return to Main Menu**.

Step 10 If prompted to configure cluster settings, choose **yes** to configure them at this time. If not prompted, choose the **Gateway Cluster Configuration** item from the **Main Menu**.

Step 11 Choose the **Both Gateway and Manager** option to have the appliance operate in standalone mode. In this mode, the appliance includes the functionality of both the Firewall and Manager (this is the mode normally used for system evaluation).

To have the appliance operate as a dedicated ACE Web Application Firewall, choose **Gateway Cluster Member** and enter the address of the Manager that will administer this appliance.

Step 12 If you chose for the appliance to act as a Manager or standalone machine, click **OK** at the screen that advises you that will need to add or change cluster members from the ACE Web Application Firewall Manager web console. (For more information on adding managed Firewalls to the Manager's control, see the *Cisco ACE Web Application Firewall User Guide*.

Step 13 When prompted, restart the appliance to have your changes take effect.

Congratulations! Your ACE Web Application Firewall is now configured and ready to use. To exit the shell interface choose **Logout** from the Main Menu.

Now that the appliance network settings are configured, you can unplug the monitor and keyboard or serial cable. After completing these steps for the Manager in your system, the Manager web console will be accessible from a browser on the local network. If you need to access the appliance operating system environment again, you can now use a remote SSH client, such as PuTTY or Cygwin.

CHAPTER 4

Accessing the Manager Web Console

The ACE Web Application Firewall Manager web console is the policy development and administration interface for the system. The web console includes tools and wizards that significantly ease the task of securing web applications. After the Firewall and Manager are installed on the network, for most practical purposes, they can be configured and administered exclusively from the Manager web console.

The ACE Web Application Firewall Manager web console works with recent browser versions. It is specifically supported on Mozilla Firefox 1.5.0.*x* and 2.0.0.*x* and on Microsoft Internet Explorer 5.5 and 6. JavaScript must be enabled in the browser for many web console features to work properly.

Logging In to the Manager Web Console

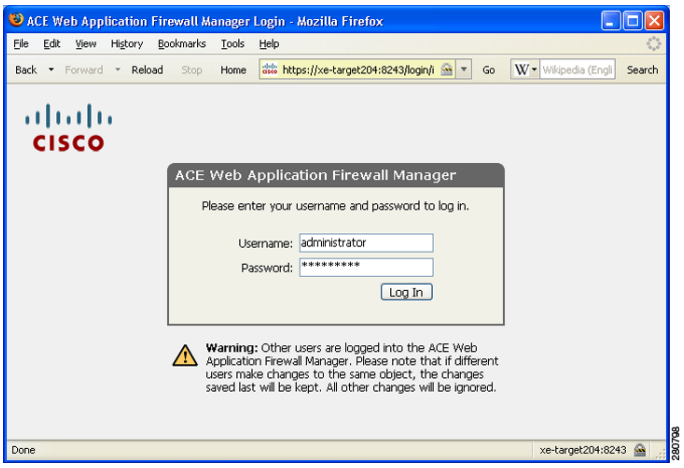
To access the ACE Web Application Firewall Manager web console from a browser:

Step 1 In the address field of the browser, enter the URL for the console: https://<*host*>:8243

Where <*host*> is the IP address or hostname you configured for the ACE Web Application Firewall Manager in the "Performing the Initial Configuration" section on page 3-10. Note that you need to connect to the Manager web console by secure HTTP (HTTPS). Also, as shown, the default port on which the Manager publishes the web console is port 8243.

Step 2 Accept the temporary certificate to view the login dialog box.

Figure 4-1 Manager Login



Step 3 Take the following steps in the login dialog box:

a. If the login box displays a **Cluster** menu, leave it set to the **Default Cluster** setting unless your system administrator tells you to use another setting.

b. In the login fields, enter a valid username and password:

   – For a new installation, enter *administrator* as the username (case-sensitive) with the default password *swordfish*.

   – If the default password has been changed, enter the username and password assigned to you by the system administrator.

   One of the first tasks required for starting a production-level project is to create user accounts in the web console for each person or group participating in the project. For evaluation or trial development, however, you may wish to use the pre-existing *administrator* account.

   Keep in mind that the administrator account in the web console is a different user account (usually with a different password) from the `root` account used to access the appliance Shell interface, as described in Chapter 3, "Performing the Initial Setup."

c. Click the **Log In** button.

If your system has *not* been pre-configured with a valid license key, a message appears notifying you that you need to update the product license. In this event, install a license as described in the following step. If licensed, the **Welcome** page appears.

Step 4 If you need to install a license, take these steps:

a. Locate the license file provided to you and copy its contents to your system clipboard. You should have a license for each appliance in your installation.

b. Click the **License Management** link at the bottom of the license error page.

c. Click the **edit** link next to the Manager and paste the license text into the **License File** field. Click **Save Changes**.

d. Perform the task for each Firewall administered by this Manager.

If you do not have a product license, refer to the *Cisco ACE Web Application Firewall Administration Guide* for information on acquiring a product licenses.

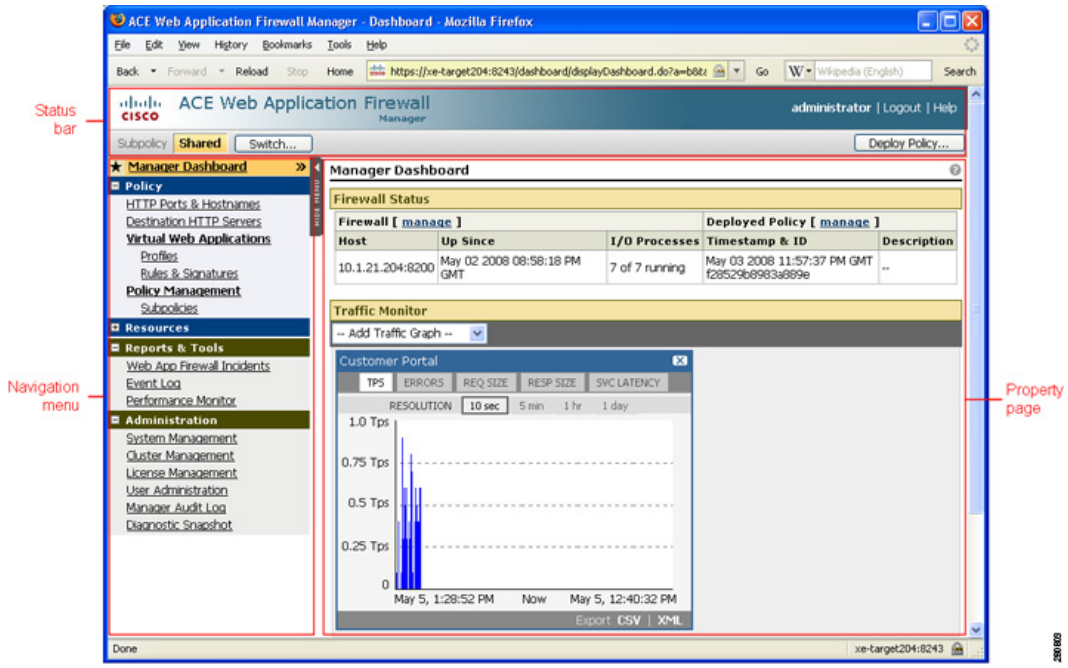
Once the Manager is licensed, you can access all areas of the ACE Web Application Firewall Manager web console and proceed with the policy configuration.

Navigating the Manager Web Console

When you log into the web console with an *empty* policy (that is, a policy in its initial state), a welcome page appears. From the welcome page, you can start configuring the Firewall policy in the console. After the policy contains objects for handling traffic, the default login page is the Dashboard.

Figure 4-2 shows the Dashboard and highlights the main elements of the web console interface.

Figure 4-2 ACE Web Application Firewall Manager web console



The primary components of the interface are:

- The *properties page* shows status information and configuration settings for a particular aspect or behavior of the system. In the case of the Dashboard, the properties page shows statistics for traffic handlers in the policy, as in Figure 4-2. For a new installation, the Dashboard will be empty.
- The *navigation menu* provides access to the various property pages of the console.
- The *status bar* provides access to common operations in the console, such as deploying or switching subpolicies (a policy can be organized into multiple parts, called subpolicies).

This guide contains detailed, click-by-click steps for performing various configuration tasks. Be aware that there is usually more than one way to navigate the console to accomplish a task. As you follow these steps, feel free to explore the web console interface and use the navigation path you prefer.

PART II

Virtual Web Apps and Firewall Rules

This part takes you through the steps for defining the policy objects that protect services at the ACE Web Application Firewall. You start by defining a virtual web application and assigning a protection profile to it. Next, you learn how to customize the behavior of the profile by enabling, disabling, or modifying the built-in firewall rules.

This part covers the following topics:

CHAPTER 5

Virtualizing a Web Application

After it is installed, the Manager contains an empty policy to which you can start adding your own rules and settings. By default, the Firewall blocks all traffic directed to it. In order to route traffic through the Firewall, you need to create a *virtual web application* object in the policy. A virtual web application defines traffic handling settings for a particular backend application.

In these steps, you'll create a virtual web application that accepts traffic for an application and passes it through without further processing. Later, you'll add traffic validation and processing steps to the policy.

Creating a Simple Virtual Web Application

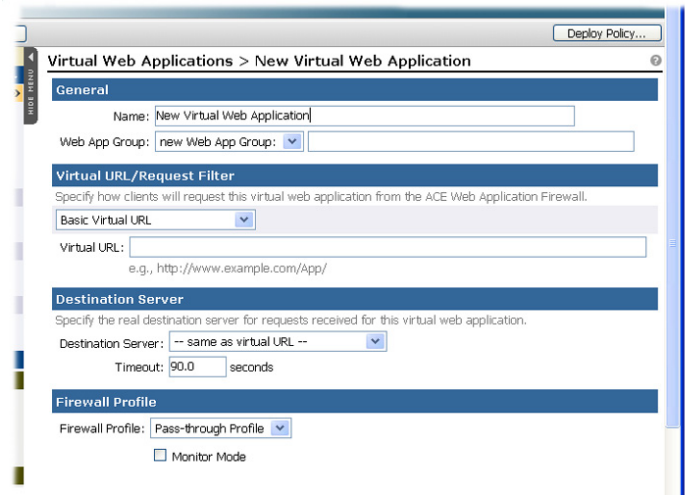
Create a virtual web application for the Poison Oak Insurance web application as follows:

Step 1 If not logged in already, log in to the ACE Web Application Firewall Manager, as described in Chapter 4, "Accessing the Manager Web Console."

Step 2 Click the **Virtual Web Applications** link in the navigation menu, and then click the **New Virtual Web Application** button. (Alternatively, from the Welcome page, click the **Configure Basic Security for a Web Application** link.)

The **New Virtual Web Application** page appears, as shown in Figure 5-1.

Figure 5-1 New Virtual Web Application page



Step 3 In the **Name** field, replace the default suggested name with a descriptive name for the virtual web application, such as **Poison Oak Web Site**.

The name identifies the virtual web application in the policy. When creating your own virtual web applications, giving them a descriptive name helps to describe the virtual web applications to other console users.

Step 4 With **new Web Apps Group** selected in the **Web App Group** menu, type **Poison Oak Apps** in the text field next to the menu.

Step 5 In the **Virtual URL/Request Filter** section, with **Basic Virtual URL** selected in the menu, type the URL that clients use to access the application in the **Virtual URL** field, such as:

http://poisonoak.example.com:8080

The Virtual URL/Request Filter settings comprise the consumer interface for the application. Clients or upstream load balancers will use this address to send requests to the application at the Firewall. In the examples in this book, this is referred to as <*Oak_VirtualURL*>.

Step 6 For the **Destination Server**, choose **new server** from the menu and type the IP address of the backend application in the **Server URL** field, such as:

http://172.16.10.1:8080

In this case, the actual IP address of the backend server serves as the destination host, while the publicly addressed hostname configured in the previous step serves as the virtual host for accessing the application at the Firewall (poisonoak.example.com). If proxying an existing application, this implies the requirement for a DNS reconfiguration, so that client requests are directed to the Firewall. For initial testing or evaluation purposes, you can mimic the effects of a DNS reconfiguration by modifying the hosts file on the computer from which you will be sending client test request to the Firewall. In the host file, map the hostname from the virtual URL you entered to the IP address of the Firewall. The exact procedures for modifying your host file varies by operating system.

Step 7 Keep the default value for the **Timeout** field, 90.0 seconds, and for the **Firewall Profile** selection, **Pass-through Profile**.

A firewall profile defines a collection of processing settings for a particular class of traffic. For the **Pass-through Profile**, all built-in rules are disabled, so that messages are passed through the ACE Web Application Firewall without validation or modification. In subsequent sections, you'll enable rules to apply various types of security measures to application traffic.

Step 8 Click the **Save Changes** button.

The Virtual Web Applications page appears, with the new virtual web application shown in the **Poison Oak Apps** group. Other types of policy objects have been added as a result of this procedure as well, including an HTTP listening port and destination server definition.

Currently, the new virtual web application definition resides on the Manager as part of the working policy only. If you were to send a request to the consumer interface for this virtual web application now, the request would be dropped at the ACE Web Application Firewall, as dictated by the deny-by-default rule. To have your change take effect on network traffic, you must deploy the policy to the Firewall, as described next.

CHAPTER 6

Deploying and Testing the Policy

So far, you've been modifying the *working policy*, the policy under development on the ACE Web Application Firewall Manager. To enforce the policy on network traffic, you need to deploy the policy to the ACE Web Application Firewall. Deployment occurs in several stages:

- The Manager compares the current and new policies and presents the differences for review.
- If you accept the changes, the Manager compiles the policy.
- The Manager transfers the compiled policy to the ACE Web Application Firewalls in its control.

Once transferred, the changed policy is applied to traffic by the ACE Web Application Firewall.

Deploying the Policy

Take the following steps to deploy a policy:

Step 1 Click the **Deploy Policy** button at the top of the ACE Web Application Firewall Manager web console.

The first deployment page shows changes between the new and old policy, in this case, objects added when you created a new virtual web application.

Step 2 Scroll to the bottom of the page and click **Continue to Next Step**.

The **Step 2 of 3, Basic Policy Review** page appears. If there are any errors or unusual conditions detected in the policy by the policy compilation process, they appear on the page.

Step 3 Click the **Continue to next step button.**

The **Compile and Deploy** page appears, which shows the ACE Web Application Firewalls in the administrative control of the Manager. Although you can deploy a policy selectively to specific Firewalls, all ACE Web Application Firewalls in a cluster should run the same policy version.

An **Out of date** status means that the policy on the Firewall differs from the one you just compiled.

Step 4 Type a description of the policy in the **Policy Description** field, such as "Initial version of Poison Oak application security".

While policy descriptions are optional, by default, it is recommended that you provide one. Descriptions help document changes in the policy version for other users. Also, the Manager administrator may configure it to be a requirement for deployment.

Step 5 Click **Deploy to Selected Gateways** to have the compiled policy transferred to all Firewalls in the list that are checked.

After a moment, the **Compile and Deploy** page reappears, this time with a status for the ACE Web Application Firewall of **Up to date**.

The policy you just deployed is automatically archived in the Policy Manager. To access the Policy Manager, click the **Policy Management** link in the **Policy** area of the navigation menu.

Notice that the Policy Manager page allows you to perform many policy administration functions, including:

- *View* or *roll back* a previous version of the policy.
- *Export* the policy to a file.
- *Save to History*, which creates a copy of the current working policy.
- *Compare policies* to view differences between two policy versions
- *Reset policy* to its initial state

The policy history list at the bottom of the page shows several policy versions, including one for the policy you just deployed. The entry shows the deployment date, the user who performed the deployment, and the description entered.

Testing Your Virtual Web Application

Recall that the virtual web application you created simply passes traffic between the back-end service and client, without inspecting or processing it. After you deploy the policy, follow these steps to test traffic routing through the ACE Web Application Firewall:

Step 1 In a web browser, access the Oak Poison home page using its consumer interface at the ACE Web Application Firewall. For example, if you created the virtual web application with a virtual URL of http://poisonoak.example.com:8080 (as in "Creating a Simple Virtual Web Application"), go to this address in the browser.

Note Unless the DNS system in your environment has already been configured to map requests addressed for the virtual URL hostname to the ACE Web Application Firewall, you will need to manually configure this mapping in the hosts file on the computer on which you are working before a request from the browser can be sent to the Firewall.

The web page should appear, demonstrating that the request and response successfully traversed the ACE Web Application Firewall.

When learning the system and sending test requests, whether the request succeeds or not, it is helpful to inspect the event log to view the details on how the ACE Web Application Firewall Manager handled the transaction. The event log provides rich details on the transaction.

Step 2 In the Manager web console, click the **Event Log** link in the navigation menu.

Step 3 If your request succeeded, look for a Notice-level event description similar to: "HTTP GET request for / from 10.1.6.12 matched virtual web app 'Oak Customer Application'; routing to server 'http://<*hostname*>:8080/'".

Failed requests that were seen by the Firewall appear as Warning-level events. If no event appears in the log for a failed request, you might try making a request directly to the Firewall at a nonexistent resource URL. For example: http://<*firewall-IP*>:<*listening-port*>/<*non-configured-path*>. Such a request should register in the event log as a 404 (Notice-level) event. If it does not, make sure that your DNS or host file is correctly configured and that there are no network connectivity issues in the system.

By default, the Firewall generates log events at the Notice-level (and higher). For troubleshooting message transaction errors at the Firewall, it is usually helpful to have the Firewall information logged at a greater level of verbosity, as described in the next section.

Setting the Log Level

As you follow the steps in this guide, it will be helpful to have as much information as possible on the activities of the ACE Web Application Firewall. To have the system produce additional log information, lower the logging severity level as follows:

Step 1 Click the **System Management** link from the navigation menu. The link appears under the **Administration** heading in the navigation menu. (You may need to expand the heading to see the link.)

Step 2 In the page, notice that there are two log-level menus. From the menu under the **ACE Web Application Firewall** heading, choose **alert, error, warning, notice, info, debug**, and click **Set Log Level**. This log-level controls the verbosity of log events generated in the course of traffic processing.

Step 3 Confirm the change when prompted, if it is safe to restart the Firewall processes now.

Step 4 To view the results of the configuration change, send another request to the Gateway and again navigate to the event log. Click on the Message GUID link next to the Notice or Warning event for the request to see all log entries associated with the message.

The Event Log viewer shows all events down to the debug level for that message. This information can be extremely helpful for analyzing and troubleshooting problems in the policy, or even interoperability problems that are external to the ACE Web Application Firewall.

When finished, you can retry your request for the Poison Oak home page through the ACE Web Application Firewall to see the additional information that is logged. After sending the request, be sure to change the default filter criteria in the Event Log viewer to include info and debug items. (That is, from the **for events of type** menu, choose “alert, error, warning, notice, info, debug”, and click **Update**.)

Note For performance reasons, logging at the info or debug severity level is intended for development, testing and troubleshooting scenarios, and not for production deployments.

Requests so far have passed through the ACE Web Application Firewall without inspection. We’ll now modify the policy to validate the message.

CHAPTER 7

Working with Firewall Profiles

So far, the Poison Oak virtual web application you created causes messages to be passed through the ACE Web Application Firewall without validation. In this section, you'll use a *profile* to validate requests.

A profile is a named collection of settings that control how the Firewall rules (active security rules, message inspection, and message rewrite rules) are applied. A profile lets you define these settings apart from the virtual web application, so that they can be applied across handlers that share traffic processing requirements. (Note that application-specific customization to the profile can be achieved using modifiers.)

The Cisco ACE Web Application Firewall provides two built-in profiles:

- The Pass-through profile provides all firewall options in a disabled state; that is, all traffic passes through to the backend service uninspected and unmodified. This is the one you used for your first virtual web application.
- The PCI Compliance profile provides preconfigured settings that help you comply with the *Payment Card Industry Data Security Standard Version 1.1*.

You cannot modify a built-in profile—they are intended to be either applied directly or (as is more likely) used as starting points for your own custom profiles. In the profiles you create, you can choose which rules to enable and set their parameters as desired.

In these steps, you create a profile that can be applied to the Poison Oak virtual web application.

Creating a Profile

Create a profile to apply to the Poison Oak application as follows:

Step 1 Click the **Profiles** link in the navigation menu, which appears under the Virtual Web Applications menu group.

Step 2 In the **Profiles** page, click the **New Profile** button.

Step 3 Type **Poison Oak Traffic Validation** in the **Profile Name** field.

Step 4 Type a description for this profile in the **Description** field, such as "Validates Oak Customer requests". The description helps to document the profile in the policy.

Step 5 Leave the **Copy Settings From** menu set to its default value of **none**.

With **none** as the **Copy Settings From** value, the Manager creates an editable version of the **Pass-through** profile.

Step 6 Click the **Create Profile** button. The Manager creates the profile and the rule settings page for the profile appears. Because you copied settings from the Pass-through profile, all built-in rules are marked "disabled" or "not configured."

After creating the profile, you can assign it to a virtual web application.

Assigning a Firewall Profile to a Virtual Web Application

To apply the rules in a profile to a particular class of traffic at the Firewall, you assign it to a virtual web application. Apply the profile you just created to the Poison Oak virtual web application as follows:

Step 1 Click the **Virtual Web Applications** link in the navigation menu.

Step 2 Click on the name of the virtual web application you created, **Poison Oak Web Site**.

The information page for the **Poison Oak Web Site** virtual web application appears. The **Firewall Profile** section of the page indicates that the **Pass-through** profile is assigned to this virtual web application.

Step 3 Click the **edit** link next to the name of the virtual web application.

Step 4 In the **Firewall Profile** section of the setting page, choose the **Poison Oak Traffic Validation** profile from the **Firewall Profile** menu.

Step 5 Click **Save Changes**.

The **Poison Oak Web Site** information page now shows the **Poison Oak Traffic Validation** profile assignment.

To see the rule enforcement applied by the new profile, click the expand icon (**+**) that appears to the left of the **Firewall Profile** label. Because you used the pass-through profile to create this profile, all built-in rules are noted as disabled or not configured. In the next procedures, you enable a rule in the profile.

Enabling and Configuring a Firewall Rule

One of the easiest ways to compromise an unprotected web application is to send requests of excessive length in order to overflow one or more of its data buffers. The built-in data overflow defense rule can thwart this type of attack by blocking requests that do not meet configurable limits.

In this section, you'll configure the data overflow defense settings of the Poison Oak Traffic Validation profile to block messages based on the number of HTTP headers they present. The default values configured for data overflow defense should allow most non-malicious traffic through the Firewall. To test attack defense, change a value to induce rule execution as follows:

Step 1 Click the **Profiles** link in the navigation menu.

Step 2 In the Profiles page, click on the name of the **Poison Oak Traffic Validation** profile.

The information page for the **Poison Oak Traffic Validation** profile appears. In the **Active Security** pane of the **Firewall Configuration** section of the page, the **Data Overflow Defense** row of the table indicates that the **Data Overflow Defense** rule is disabled.

Step 3 Click the **edit** link that appears in the **Data Overflow Defense** row of the table.

The **Data Overflow Defense** rule editor appears.

Step 4 Check the **Enforce the following data limits on requests** checkbox.

The other controls on the page become enabled.

Step 5 Check the **Maximum Number of HTTP Headers** checkbox.

Step 6 In the **Maximum Number of Headers** field, replace the default value of **20** with the value **1**.

HTTP requests from web browsers typically have multiple headers. Therefore, setting this value to **1** will ensure this rule will be triggered. By default, when triggered, this rule terminates the connection and returns a 400 client error. However, we'll send a custom error message.

Step 7 In the **Actions** section, near the bottom of the page, allow the **Event Log** menu to remain at its default value, **Warning-level event**, and choose **Return a Custom HTTP Error Response** from the **Response** menu.

Step 8 Type **200** into the **Status** field.

Step 9 Type **Poison Oak traffic validation blocked this request** into the **Body** field.

This message will appear in your web browser when you send a request that matches this firewall rule.

Step 10 Click the **Save Changes** button.

The status of the **Data Overflow Defense** rule should now be listed as enabled.

Step 11 Have your changes take effect by deploying the policy, as described in Chapter 6, "Deploying and Testing the Policy."

You can now test the configuration. You may need to clear your web browser's cache before sending the request.

To test the configuration, from a browser, navigate to the web page presented at the virtual URL for the web application you are proxying at the Firewall. Verify that the Firewall rejects the request and returns your custom error message.

Disabling a Firewall Rule

As mentioned, the traffic validation rule you configured in the previous section is unrealistic for most applications. After verifying that the new firewall rule successfully blocks traffic, you should disable the rule as follows:

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

Step 2 In the **Firewall Configuration** section of the page, click the **edit** link next to **Data Overflow Defense**.

Step 3 On the **Data Overflow Defense** rule page, click the **Enforce the following data limits on requests** item to clear its selection.

Step 4 Click the **Save Changes** button.

The **Active Security** section of the **Poison Oak Traffic Validation** page notes that the Data Overflow Defense rule for this profile is disabled in the working policy.

Step 5 Deploy the policy to have your changes take effect.

Clear your web browser's cache and use your web browser to send a request to the public-facing URL of the virtual web application. The Firewall allows the request and the web browser displays the Poison Oak home page.

Next, we'll see how the security incident you just generated appears in the logs.

CHAPTER 8

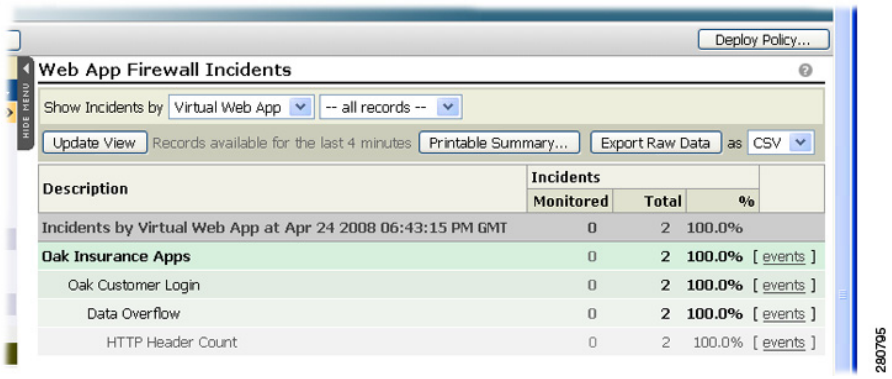
Using the Logs

The event log is one of several tools provided by the ACE Web Application Firewall for monitoring system activity. Another is the incidents report. The incidents report provides information specifically on rule-triggering events.

Inspecting the Incidents Report

If you followed the steps in the previous section closely, you have generated a data overflow defense incident. To view the incident report for it, click the **Web App Firewall Incidents** link in the Manager navigation menu. The Manager displays the **Web App Firewall Incidents** page, as shown in Figure 8-1.

Figure 8-1 Web App Firewall Incidents page



The **Incidents** column shows the number of times each rule executed in the specified time frame, while the **%** column represents this value as a percentage of all firewall rules executed. If the recent activity of the data overflow defense does not appear on the page, use the **Show Incidents by** and **Update View** controls to update the report. To sort incidents by virtual path, virtual web application, group, or rule set that reported them, choose the item from the **Show Incidents by** menu and then click **Update**.

If you did not execute any other rules, the % column should show that Data Overflow incidents represent 100 percent of all rules executed.

The **events** link takes you to the event log entry for the event, where you can see details on the event. For more information on the event log, see "Testing Your Virtual Web Application" section on page 6-22.

Using the Performance Monitor

The performance monitor presents performance statistics on traffic processing activities. It presents counts of messages processed, average message sizes, and times required to process messages. You can use these values to measure and track the performance of your network, and to determine whether the current policy is performing as intended.

To view the performance monitor, take the following steps:

Step 1 Click the **Performance Monitor** link in the Manager menu.

Step 2 In the **Performance Monitor** page, click the plus sign (+) next to the **Poison Oak Apps** web app group.

The display expands to show a separate set of statistics for the **Poison Oak Web Site** virtual web application. At this point, it's the only web application in this group; however, the security policy for a production environment typically has multiple virtual web app groups, each of which may contain multiple virtual web applications.

Like the **Event Log**, the **Performance Monitor** provides controls that allow you to filter its display by Firewall and by time period.

Step 3 To view performance statistics by time period:

a. Choose the **for last hour** item from the **Show** menu on the right, top side of the page.

b. Click the **Update View** button.

The **Performance Monitor** statistics reflect the period of time that began one hour prior to when you clicked the **Update** button. If you have completed the preceding steps quickly, the display may not change very much.

Step 4 To view performance statistics for another time period:

a. Choose the **all records** item from the **Show** menu on the right, top side of the page.

b. Click the **Update View** button.

The **Performance Monitor** statistics reflect all records processed since the ACE Web Application Firewall began running. Again, the change to the display may not be very dramatic if you have worked through the preceding steps in this guide relatively quickly. However, you can come back to the **Performance Monitor** page at any time to see updates to its display as you send additional messages through the ACE Web Application Firewall.

For detailed information on the statistics shown in the performance monitor, open the help page from the **Performance Monitor** page.

CHAPTER 9

Creating Request Match Filters

Incoming requests are typically assigned to a virtual web application based on a match between the request URL and the virtual URL of the virtual web application. By default, a virtual URL is matched to a request on a prefix-match basis. This means that the virtual URL can match a request even if the request URL includes additional steps in the request path. That is, a request for http://example.com/resources/images/logo.gif can be matched to a virtual URL of http://example.com/resources/.

If a request can be matched to more than one virtual web application, the one with the most specific consumer interface wins. That is, with a second virtual web application with a virtual URL of http://example.com/resources/images/, the request for logo.gif would be matched to the second virtual web application rather than the first.

In addition to the request URL, a virtual web application can use other message matching criteria as well, such as the HTTP method type of the request (GET, POST, DELETE, and so on), HTTP header values, and HTTP parameter values. The request filter conditions allows you to apply special processing or validation only to the requests that match particular characteristics.

In the following steps, you'll create a filter expression that applies a security rule specifically to the login page. The security rule will check for SQL injection attack.

Demonstrating SQL Injection Vulnerability

A SQL injection attack is one in which SQL statements embedded in user input data are executed by the database layer of the application. Insufficient user input validation can expose web applications to this type of attack. The Poison Oak sample application is vulnerable to SQL injection attack, as you can demonstrate with the following steps:

Step 1 In your browser, go to the Poison Oak home page.

Step 2 In the **Username** field, type any value. The username does not need to be that of a valid user in the system. For instance, try the username admin.

Step 3 Type the following into the **Password** field:

```
' or 'A' = 'A
```

This is the injection string. It modifies the logic of the SQL statement passed to the database with the "or" operator. As a result, instead of just checking the user credential, the database also evaluates the truth of the added SQL fragment. Since the values "A" and "A" are equal, the statement evaluates to true. The web application interprets this result as a valid credential submission.

Step 4 Click the **Login** button.

Log in should succeed. Next, you'll define a request filter that validates requests to the login page to prevent such attacks.

Creating a Filter Expression

At this point, the virtual web application you created treats all requests to the protected application alike. You can specify specialized processing for particular components of the web application using request filters. In this case, you'll impose security measures for the login page to protect it from SQL injection attack:

Step 1 Click the **Virtual Web Applications** link in the navigation menu.

Step 2 Click the **Poison Oak Apps** web app group link. This link appears in the green-shaded banner.

The **Poison Oak Apps** group appears. At the moment, it contains only one virtual web application.

Step 3 Click the **Add Virtual Web Applications** link to the right of the **Virtual Web Applications** banner.

The **New Virtual Web Application** page appears.

Step 4 In the **General** section, type **Poison Oak Login** into the **Name** field

Step 5 Choose **Poison Oak Apps** from the **Web App Groups** menu, if not already selected.

Step 6 In the **Virtual URL/Request Filter** settings section:

a. Choose the **Custom Virtual URL with Filters** item from the menu.

b. In the **Port/Hostname** menu, choose the Poison Oak port previously created.

c. Type **login.do** in the **Path** field. This is the page that is vulnerable to SQL injection attack in the Poison Oak application.

d. For the other settings in the **Virtual URL/Request Filter** section, keep the default selections. That is, **Matching Mode** should be **prefix**, **Methods** should be **basic HTTP Methods (GET/POST/HEAD)**, and HTTP headers and parameters should be **ignore**.

Step 7 For the **Destination Server**, choose the server object previously created.

Step 8 For now, the **Firewall Profile** selection can remain at its default value. You'll create a new profile for this virtual web application later.

Step 9 Click **Save Changes**.

The list of virtual web applications in the **Poison Oak Apps** group should now include the new virtual web application for the login.do page.

If you were to deploy the policy now, the web application would still be vulnerable to SQL injection attacks, since its profile does not check for SQL injection attack signatures. To secure the application, create another profile, as described next.

Enabling SQL Injection Inspection in the Profile

To configure a SQL injection rule in a new profile, follow these steps:

Step 1 Create a new profile named **Check Logins** by duplicating the **Pass-through** profile. For instructions, see the "Creating a Profile" section on page 7-25.

Step 2 In the **Check Logins** settings page that appears when you finish creating it, click the **edit** link next to **SQL Injection** in the **Message Inspection** section.

Step 3 In the **SQL Injection** page, choose **Enabled** from the **Rule Set Mode** menu.

The page displays additional controls and a summary of SQL injection rules.

Step 4 Keep the default **Level** selection, **Strict**.

This severity level controls which SQL injection rules in the Firewall's base configuration are applied in this profile. The rules appear in the list below the menu.

Step 5 In the **Action** section at the bottom of the page, choose **Return a Custom HTTP Error Response** from the **Response** menu.

Step 6 Type **Strict SQL injection checking blocked this request** into the **Body** field

Step 7 Click the **Save Changes** button.

The **Check Logins** profile settings page shows that the profile applies strict SQL Injection message inspection.

Assigning the Profile to the Virtual Web Application

Assign the profile you just created to the new virtual web application as follows:

Step 1 On the **Virtual Web Applications** page, click the **Poison Oak Login** virtual web app.

Step 2 Click the **edit** link that appears on the right side of the page.

Step 3 Choose the **Check Logins** profile from the **Firewall Profile** menu.

Step 4 Click **Save Changes**.

Step 5 Deploy the policy, as described by Chapter 6, "Deploying and Testing the Policy."

You can now test the configuration.

Testing Your Request Filter Conditions

To try out the virtual web application for the login page along with the SQL injection security enabled for the profile, follow these steps:

Step 1 If still logged in to a Poison Oak application session, log out by clicking the **logout** link.

Step 2 Repeat the login steps listed in the "Demonstrating SQL Injection Vulnerability" section on page 9-31.

Step 3 Verify that the Firewall now blocks the SQL injection attack and your web browser displays the custom error message you configured.

Notice that the first virtual web application you created, **Poison Oak Web Site**, handles requests for all pages except the login.do page. If you were to disable the **Poison Oak Web Site** virtual web application, only requests for the login.do page would succeed while all other requests would fail.

Disabling a virtual web application effectively turns off message acceptance at its consumer interface. It should be noted, however, that requests that would have matched a virtual web application that is disabled are not blocked if there are other virtual web applications in the policy that have less specific but matching consumer interface criteria.

CHAPTER 10

Monitor Mode

An important stage of policy development involves identifying potential "false positives" introduced by a policy (that is, legitimate messages that incorrectly trigger a rule). The best way to discover false positives is to observe the policy's effect on live application traffic. However, deploying an untested policy production environment may hinder legitimate access to services.

Monitor mode allows you to direct actual message traffic at a firewall rule without inadvertently blocking legitimate access. A rule that is in monitor mode executes all of its configured behaviors except its terminal action. That is, it inspects the message against each enabled rule and generates log entries, but does not block messages that trigger inspection rules.

Note While messages are not blocked in monitor mode, they are still subject to non-blocking active security features, such as HTTP header processing and cookie security. For more information, see the *ACE Web Application Firewall User Guide*.

You can apply monitor mode at the following levels: to an individual rule in a profile (this affects all virtual web application that use the profile), to a particular virtual web application, or to the entire ACE Web Application Firewall, which places all web application traffic processing into monitor mode. This "global monitor mode" allows you to collect information on traffic characteristics in a non-intrusive manner. After collecting data for a period of time, you can use the incident report to determine the ratio of legitimate traffic versus attack traffic and the types of attack against a particular web site or resource.

Development Flow

Monitor mode can be used as a integral part of your development process. For instance, monitor mode would be used in a typical project development process as follows:

1. Create an initial firewall profile that is more strict than will ultimately be needed. For example, you might base your profile on a copy of the built-in PCI compliance profile.
2. Create virtual web applications. Set all set to monitor mode. For convenience, the web console allows you to set the entire Firewall to monitor mode (or enabled mode) at once.
3. Deploy the policy and send a good volume of test traffic to your web application.
4. Review the incident report for requests that cause a high number of web application firewall incidents. Determine whether these are false positives.
5. Modify your firewall profile as necessary to disable or exempt unneeded rules or signatures.

Repeat these steps until you arrive at a policy that can reliably screens attacks without producing false positives.

Placing a Rule into Monitor Mode

To see how monitor mode works, reinstate the highly restrictive Data Overflow Defense rule you created earlier and put it in monitor mode, as follows:

Step 1 Click the **Profiles** link in the Manager navigation menu

Step 2 Click on the name of the first profile you created, **Poison Oak Traffic Validation**.

Step 3 In the profile page, click the **edit** link next to **Data Overflow Defense**.

Step 4 In the **Data Overflow Defense** page, check the **Enforce the following data limits on requests** checkbox. Also, select the **Maximum Number of HTTP Headers** option and make sure its value is 1, making it easy to trigger.

Step 5 At the bottom of the **Data Limits** section, check the **Monitor Mode** option.

Step 6 Click **Save Changes**.

The **Active Security** section of the profiles page notes that the **Data Overflow Defense** rule for this profile is in **monitor** mode.

Step 7 Deploy the policy as described in Chapter 6, "Deploying and Testing the Policy."

Step 8 In your web browser, again access the home page of the Poison Oak application through the Firewall. (You may need to clear your web browser's cache to ensure that you see fresh request results.)

The request should succeed even though the **Data Overflow Defense** rule is active.

Step 9 Open the **Web App Firewall Incident** report.

Notice that new events generated for the **Data Overflow Defense** rule are listed in the **Monitored** column.

Step 10 Click the **events** link to see more details on the incident. Notice that events generated by the **Data Overflow Defense** rule have a **Create Exemption** link in their descriptions. You can use this link to fine-tune a policy to accommodate varying security or processing requirements for traffic handled by a virtual web application.

Next, you'll see how to use incident-based modifiers to fine-tune traffic handling at the ACE Web Application Firewall.

CHAPTER 11

Fine-Tuning Firewall Rules

During testing or monitoring of a policy, you may encounter events in which legitimate requests mistakenly trigger a security rule. In order to avoid so-called "false positives," you can define explicit conditions under which the firewall rule accepts traffic that would otherwise be blocked.

Whenever a firewall rule executes (whether in monitor mode or not) it creates entries in the Web App Firewall Incidents page and in the Event Log. Descriptions of rule blocking events in the Event Log have a **Create Exemption** link that you can use to stop subsequent requests of the same type from triggering an incident. In this chapter, you'll use this feature to construct a modifier to a virtual web application. A modifier applies special processing to a subclass of the traffic handled by a virtual web application.

Restricting the Size of an HTTP Argument

Previously, you used the data overflow defense rules to stop requests that have an excessive number of HTTP headers. In these steps, you'll use the same rule to limit the size of request arguments. Requests with HTTP arguments of excessive length can cause a buffer overflow condition in some web applications. The Data Overflow Defense feature can enforce limits on the size of request arguments.

Configure an argument-length restriction as follows:

Step 1 Click the **Profiles** link in the Manager navigation menu.

Step 2 Click on the name of the profile you created, **Poison Oak Traffic Validation**.

Step 3 Click the **edit** link next to **Data Overflow Defense** in the **Active Security** section of the page.

The **Data Overflow Defense** rule editor displays the current configuration of this rule: allow a maximum of one HTTP header in monitor mode.

Step 4 Uncheck the **Maximum Number of HTTP Headers** option.

Step 5 Check the **Maximum Size of Any Argument** checkbox.

Argument validation applies to arguments in the URL of GET requests and to arguments in the URL and body of POST requests.

Step 6 In the **bytes** field, replace the default value with **20**.

Step 7 Uncheck the **Monitor Mode** option.

Step 8 In the **Actions** section, replace the custom error response body with a descriptive message for this configuration, such as "Argument too big."

Step 9 Click **Save Changes**.

Step 10 Deploy the policy.

The Firewall now checks the length of arguments passed in requests to the Poison Oak application.

Testing the Data Overflow Defense Rule

Test the new rule configuration as follows:

Step 1 First make sure that the Firewall accepts requests within the configured limits. In your browser, enter the following address:

http://<*Oak_VirtualURL*>/index.jsp?myparam=thisisabigarg

Where <*Oak_VirtualURL*> is the hostname and port at which the Firewall exposes the Poison Oak web application to client requests.

Note In general, before sending new requests to the Poison Oak application you may need to clear your browser's cache to ensure that you see fresh rather than cached results.

The request should be accepted. Since the Poison Oak application is not programmed to handle the myparam argument, it is simply ignored.

Step 2 Now send a request that violates the argument-length restriction, such as:

http://<*Oak_VirtualURL*>/index.jsp?myparam=thisisanevenbiggerarg

This time, the Firewall blocks the request and returns an error response.

In the next section, you'll create a modifier that exempts the myparam parameter from this rule, while continuing to enforce the restriction on all other arguments.

Creating a Modifier

Recall that event log entries generated by Firewall security incidents provide a way to define an exemption for the violation that triggered the rule. Next, use this feature to create a modifier that exempts the myparam parameter from the 20-byte size limit, as follows:

Step 1 Click the **Event Log** link in the Manager navigation menu.

Step 2 Find an event log entry generated by the Data Overflow Defense rule after you applied the argument-size limit. To quickly find such an event among many log items, follow these steps:

a. Choose **all records** from the **During** menu at the top of the page.

b. In the **description** field, type "argument exceeded 20 bytes".

Note The **description** field is case-sensitive. Type the search string exactly as indicated.

c. Click the **Update** button.

Incidents triggered by the argument-length rule should appear. Notice that the description of the event includes a **Create Exemption** link.

Step 3 Click the **Create Exemption** link in the **Event Log** entry.

The Manager displays an **Edit Firewall Modifier** page with prepopulated values based on the incident.

Step 4 Keep the prepopulated values for the **Path** checkbox (checked) and the **Path** field (/index.jsp). This modifier will only apply to requests for this page.

Step 5 Change the **Matching Mode** to **prefix**.

Step 6 Exclude the argument-length restriction for the `myparam` parameter, as follows:

a. Check the **Parameters** checkbox.

b. Choose the **query string matches regex** item from the **Parameters** menu.

c. In the text field, type `myparam=*`

This regular expression causes the modifier to apply to all values of the `myparam` parameter.

Step 7 With the other settings on the page at their prepopulated values, click **Save Changes**.

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

When finished deploying, test the configuration of the modifier, as described next.

Testing the Modifier

After creating a modifier that exempts the `myparam` parameter from the 20-byte argument-length restriction, send a request to the Poison Oak sample application to test the modifier, as follows:

Step 1 Clear your browser's cache.

Step 2 Send a request that contains the same `myparam` parameter that previously violated the argument-length restriction, such as:

`http://<Oak_VirtualURL>/index.jsp?myparam=thisisanevenbiggerarg`

This time, the modifier to the data overflow defense rule exempts the `myparam` parameter from this rule, so the Poison Oak home page appears in your browser.

Step 3 Change the name of the parameter in the request to make sure that the restriction is still imposed on other parameters to the page and try again:

`http://<Oak_VirtualURL>/index.jsp?bar=thisisanevenbiggerarg`

This time, data overflow defense blocks the request and returns an error message to your browser.

Notice that a modifier applies special processing to a subclass of traffic handled by a virtual web application. For many false positives, the condition that caused the incident is applicable to requests for other resources in the web application, not just to the page on which the incident is triggered. If this is the case for a given incident, you will need to make the change effected by the modifier at a broader level, such as at the profile level.

PART III

Using Basic Security Features

This part describes how to use security features of the ACE Web Application Firewall to protect against common types of attack, such as cookie tampering, command injection, and cross-site scripting attacks.

This part covers the following topics:

CHAPTER 12

Securing HTTP Cookies

Web applications often use HTTP cookies to store information about a particular user or session. The server embeds a cookie in its response message, and the browser returns the cookie unchanged in a subsequent request.

Because cookies are implemented as clear text, they may be compromised easily, allowing attackers to read sensitive data or inject commands by returning a modified version of the cookie to the server. This example describes how to configure the ACE Web Application Firewall to prevent cookie tampering.

About Cookie Security

When cookie security features are not enabled, cookies pass through the ACE Web Application Firewall without special cryptographic checking. To protect cookie data from tampering, the ACE Web Application Firewall can cover the cookie with a digital signature or it can encrypt the cookie payload.

Either approach ensures that the cookie cannot be modified undetected, but encrypting the cookie obscures its content. If your browser needs to be able to read cookie data, you'll need to sign rather than encrypt cookies.

The cookie security settings of a firewall profile specify whether that profile signs or encrypts cookies. You may configure the profile to perform only one cookie-protection operation or the other.

How Cookie Security Works

ACE Web Application Firewall cookie security features are transparent to the backend application. When the backend system returns a response message that contains a cookie, the ACE Web Application Firewall processes (signs or encrypts) the cookie before delivering it to the client. Only the processed cookie is transmitted to the client and only the processed cookie resides on the client system. When a subsequent request from the client returns this cookie, the ACE Web Application Firewall decrypts it or tests its signature. If the cookie is intact, the ACE Web Application Firewall passes along the decrypted or unsigned cookie to the backend application.

If the cookie was modified on the client side since sent by the ACE Web Application Firewall, decryption or signature verification fails, causing the ACE Web Application Firewall to remove the cookie before the request is sent to the backend system.

Interaction of Cookie Security and Other Active Security Features

Because cookies are carried in the headers of HTTP messages, they are always subject to constraints imposed by HTTP headers security features, even when cookie security is not enabled. For example, a cookie that exceeds the maximum cookie size specified in data overflow defense triggers this security rule.

Signing a cookie adds at least one header to the processed message and changes the length of the cookie itself. Encrypting a cookie does not necessarily change the number of headers in the message but it may increase the length of the cookie significantly. If your Active Security configuration specifies very stringent constraints on the length or number of HTTP headers, a signed or encrypted cookie may trigger Data Overflow Defense rules that an unsigned cookie would not match, resulting in unwanted rejection of messages. For more information, see the online help for cookie security.

Demonstrating Insecure Cookies

In this section, you'll use commonly available tools to examine and tamper with the cookies used by the Poison Oak sample application. In subsequent sections, you'll enable Cookie Security features and see how they ensure the integrity and security of cookie data.

The procedures in this section use the *Add N Edit Cookies* browser add-on for Firefox; however, any web-development environment or browser plug-in that enables you to view and edit cookies is suitable for completing these tasks.

User Login Data in Example Service Cookies

When you log in to the Poison Oak home page, it returns a cookie that holds the username and password as clear text. To demonstrate insecure cookies, follow these steps:

Step 1 First, disable the data overflow defense settings configured for your virtual web application, as follows:

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

b. Click the **edit** link next to **Data Overflow Defense** in the **Active Security** section.

c. Uncheck the **Enforce the following data limits on requests** checkbox and then **Save Changes**.

Step 2 Also, for simplicity, disable the virtual web application you created for the login page as follows:

a. Click the **Virtual Web Applications** link in the Manager navigation menu and then click the **Poison Oak Login** virtual web application link.

b. Click the **disable** link on the right side of the page.

Disabling a virtual web application in effect turns it off. Requests to the login page will subsequently be handled by the Poison Oak Web Site virtual web application.

c. Click **Save Changes** and deploy the policy.

Step 3 In the Poison Oak home page, log in using these credentials:

- Username: `Perry`
- Password: `Johnson`

Step 4 Click the **Accounting Reports** link.

Your browser displays a message indicating that you cannot view the accounting reports. In order to view these reports, you must log in as the Accounting user. Instead of doing so, however, you'll fake this login by tampering with the cookie.

Step 5 With your cookie editor, examine the **username** cookie that the Poison Oak application returned to your browser. With the Add n Edit Cookies add-on for Firefox, for example:

a. Choose **Tools > Cookie Editor** from the browser menu.

   If the **Cookie Editor** menu item does not appear in the **Tools** menu of Firefox, the Add n Edit Cookies add-on is not installed.

b. To view only cookies returned by the example service, type the virtual URL of the Poison Oak sample application into the field at the top of the **AnEC Cookie Editor** window and then click the **Filter/Refresh** button.

c. In the list of cookies near the top of the editor pane, click the **username** cookie to select it.

   The editor displays the **username** cookie. Note that the username you sent to the example service (Perry) appears as the payload (content) of the cookie.

Step 6 Use your cookie editor to change the content of the **username** cookie from **Perry** to **Accounting**.

Note that this value is case-sensitive. To perform this task in the Add n Edit Cookies editor, take the following steps:

a. In the list of cookies near the top of the editor pane, click the **username** cookie to select it.

   The editor displays the **username** cookie.

b. Click the **Edit** button.

   A window that provides text-editing fields for the cookie data appears.

c. Change **Perry** to **Accounting** in the **Content** field of the cookie.

   Note that you must provide this case-sensitive value exactly as above; that is, **accounting** won't produce the expected results.

d. Click the **Save** button

   The editor rewrites the cookie as you specified.

e. Close the cookie editor.

Step 7 Click the **Home** link on the page.

The page displays the **Welcome Accounting** message instead of the **Welcome Perry** message, indicating that it has accepted and used your edited cookie.

Step 8 Return to the Poison Oak home page and click the **Accounting Reports** link again.

The **Accounting Reports** page appears, confirming that you've gained access to a restricted area of the site by tampering with cookie data.

Next, you'll configure the Cookie Protection features of the ACE Web Application Firewall to protect against this type of attack.

Signing Cookies

To protect the Poison Oak web application from cookie tampering, you'll use the cookie security features in the ACE Web Application Firewall to apply a digital signature to the payload of the cookie. When finished, the ACE Web Application Firewall will reject cookies that have been modified.

To configure cookie security, 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 **Cookie Security** feature is disabled in the profile. The feature appears with the Active Security features.

Step 2 Click the **edit** link next to **Cookie Security**.

The **Cookie Security** rule editor appears, as shown in Figure 12-1.

Figure 12-1 Cookie Security rule editor

Step 3 Check the cookie security checkbox.

The other controls in the **Cookie Security** rule editor become enabled.

Step 4 Allow the pull-down menu to remain at its default setting of **Sign (HMAC-SHA1)**.

Step 5 Type a secret passphrase in the **Passphrase** field.

The ACE Web Application Firewall will use this passphrase to sign cookies and to validate the signed cookies in subsequent client requests. You can use any passphrase you like for this value, but in an actual implementation, you would want to use a passphrase that is at least seven characters in length.

Step 6 Click the **Save Changes** button.

The information page for the profile shows that the **Cookie Security** rule is enabled.

Step 7 Deploy the policy to have the changes take effect.

In the next section, you'll test the new cookie security configuration in the policy.

Testing Cookie Signature

Now that your virtual web application signs cookies, you should not be able to modify the cookie successfully. When the ACE Web Application Firewall detects that a cookie has been altered, it removes the altered cookie and passes the message to the backend service without the cookie. To demonstrate this behavior, follow these steps:

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

Step 2 If you are already logged in to the Poison Oak home page, click the **Logout** link.

Step 3 Again log in using these credentials:

- Username: `Perry`
- Password: `Johnson`

Step 4 As previously described, use your cookie editor to change the content of the **username** cookie from **Perry** to **Accounting**.

Notice that additional cookies appear in the cookie editor. The cookies with "sig" appended to their names contain the cookie signatures generated by the Firewall.

Step 5 Now when you try to access the Accounting reports page, the page should indicate that only a user with the accounting role can access the reports.

Also, on the Poison Oak home page, the **Username** field, **Password** field, and **Login** button appear, since the Firewall removed the altered cookie before passing the request to the backend service. The Poison Oak sample application treats a request that arrives with no username cookie as a new login.

Encrypting Cookies

Next, you'll see how the Firewall performs cookie encryption. Encryption ensures the validity of cookies, and prevents attackers from being able to view the content of cookie payload.

To encrypt cookies and reject altered cookies, take the following 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**.

Step 2 Click the **edit** link next to **Cookie Security** in the **Active Security** section.

Step 3 With the checkbox selected, change the action from **Sign (HMAC-SHA1)** to **Encrypt (AES)** .

Step 4 You can keep the **Passphrase** value you entered previously, or type a new one.

The ACE Web Application Firewall uses the passphrase to encrypt response cookies and to decrypt the cookies in subsequent requests before forwarding them to the application.

Step 5 Click the **Save Changes** button.

Step 6 Deploy the policy to have the changes take effect.

Testing Encrypted Cookies

Now that your virtual web application encrypts cookies, you should not be able to modify the Oak Insurance cookie successfully, nor should you be able to view the cookie payload as clear text. When the ACE Web Application Firewall detects that a cookie has been altered, it removes the altered cookie and passes the message to the backend service without the cookie.

To demonstrate cookie encryption, follow these steps:

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

Step 2 Log in using these credentials:

- Username: `Perry`
- Password: `Johnson`

Previously, you were able to fake a successful login as the Accounting user by tampering with the cookie. This time, the username cookie does not appear as clear text; in fact, even its name is encrypted, so no cookie named username appears at all. If, based on past experience with the Poison Oak sample application, you modify the cookie that you feel is likely to hold the username, it will not decrypt successfully and the ACE Web Application Firewall will reject the altered cookie.

Step 3 Use your cookie editor to change the content of the cookie with the encrypted name to **Accounting**.

Step 4 Click the **Accounting Reports** link on the **Oak Insurance** home page.

Your browser displays a message indicating that you cannot view the accounting reports.

CHAPTER 13

Rewriting Response Content

The ACE Web Application Firewall enables you to inspect and mask content in response messages. Message rewrite rules prevent backend applications from emitting sensitive information, such as credit card numbers. The rule replaces a matched pattern in a response with a replacement character. In addition to the credit card number masking rule demonstrated in this section, you can supplement the built-in rules with those you create.

Demonstrating Insecure Credit Card Data

The Poison Oak sample application returns credit card data in clear text in message responses, which you can see as follows:

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 **Payment** link on the Oak home page.

The page displays payment data or a **No orders were received** message.

Step 3 If the page displays a **No orders were received** message, click the **Pay anyway** link; otherwise, click the **Submit order** button.

The payment information page displays the credit card number used to submit the order. This number appears as clear text in the source of the page, making it susceptible to theft by an eavesdropper.

In the next section, you'll configure your virtual web application to rewrite response messages from the backend service with "X" characters in place of credit card numbers.

Masking Credit Card Numbers in Response Messages

The **Credit Card Account Number Masking** rule causes credit card numbers in response messages to be masked. To apply credit card number masking, follow these steps:

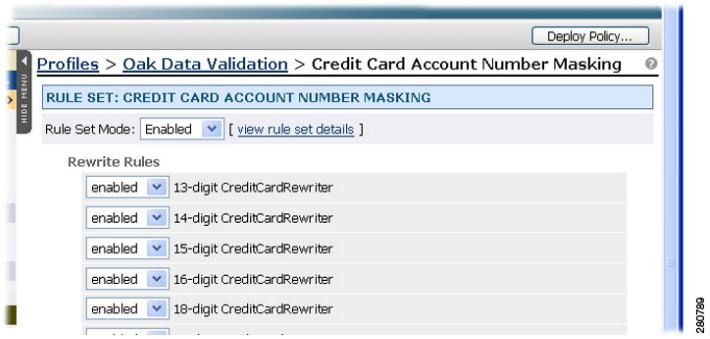
Step 1 Click the **Profiles** link in the Manager navigation menu and then click on the name of the profile you created, **Poison Oak Traffic Validation**.

Notice that the Credit Card Account Number Masking rule is disabled in the profile. The rule appears in the Message Rewrite section of the profile.

Step 2 Click the **edit** link next to **Credit Card Account Number Masking**.

The **Credit Card Account Number Masking** rule page appears.

Figure 13-1 Credit Card Account Number Masking rule editor



Step 3 Choose the enabled item from the **Rule Set Mode** menu.

Step 4 Allow the **Rewrite Rules** menus to remain at the **enabled** default setting.

Step 5 Click the **Save Changes** button.

The **Credit Card Account Number Masking** rule should now appear as enabled in the profile.

Step 6 Deploy the policy to have the changes take effect.

In the next section, you'll test your new **Credit Card Account Number Masking** policy.

Testing Credit Card Number Masking

To test your policy change, follow these steps:

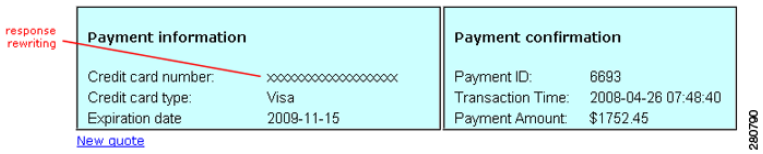
Step 1 In your browser, go to the Poison Oak home page through the ACE Web Application Firewall.

Step 2 Click the **Payment** link on the Oak home page.

Step 3 Click the **Pay anyway** link and then **Submit Payment**.

A payment information page displays "X" characters in place of the credit card number that was used to submit the order.

Figure 13-2 Masked Credit Card data



CHAPTER 14

HTTP Error Response Mapping

Error messages returned by backend applications (stack traces or other detailed errors) can provide sensitive implementation details that an attacker can use to identify ways to exploit the system. For example, the following listing shows a portion of an exception stack trace returned by a web application:

```
HTTP ERROR: 500
INTERNAL_SERVER_ERROR
RequestURI=/register.do;jsessionid=9htwpf0y943o


Caused by:

java.lang.NullPointerException
    at com.oakinsurance.struts.actions.RegisterAction.execute(RegisterAction.java:44)
    at org.apache.struts.action.RequestProcessor.
            processActionPerform(RequestProcessor.java:431)
    at org.apache.struts.action.RequestProcessor.process(RequestProcessor.java:236)
    at org.apache.struts.action.ActionServlet.process(ActionServlet.java:1196)
    at org.apache.struts.action.ActionServlet.doGet(ActionServlet.java:414)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:707)
    at javax.servlet.http.HttpServlet.service(HttpServlet.java:820)

...

    at org.mortbay.io.nio.SelectChannelEndPoint.run(SelectChannelEndPoint.java:395)
    at org.mortbay.thread.BoundedThreadPool$PoolThread.run(BoundedThreadPool.java:450)


Powered by Jetty://
```

Notice that the error response indicates an internal server error with a status code of 500, along with a stack trace of the Java exception that occurred in the application. This type of stack trace can be helpful for testing and debugging an application, but when allowed to be emitted in a production environment, it provides an attacker with valuable information on backend systems.

By default, the ACE Web Application Firewall allows error messages from back-end applications to pass through unaltered. The HTTP Exception Mapping feature enables you to obscure details of the back-end implementation by returning a generic message in place of the error message from the backend application.

Note Instead of concealing information, error mapping can be used to provide additional information to the client on an error, without having to modify the backend system.For instance, you could map a generic “403 Forbidden” response to a 403 response that contains additional information pertinent to the application, such as “Incorrect password. Please try again.”

In these steps, you'll configure HTTP Exception Mapping to replace the error message from the web application with a generic error response. You can use a stack trace response generated by the Oak sample application to test your error mapping configuration.

The Oak sample application can be induced to return a stack trace by going to the payment page directly, without following the order processing procedure. To view a raw stack trace in the browser, go directly to the following address: http://<*Oak_VirtualURL*>/payment.do

Mapping HTTP Exceptions

To map HTTP exception responses to a generic error message, follow these steps:

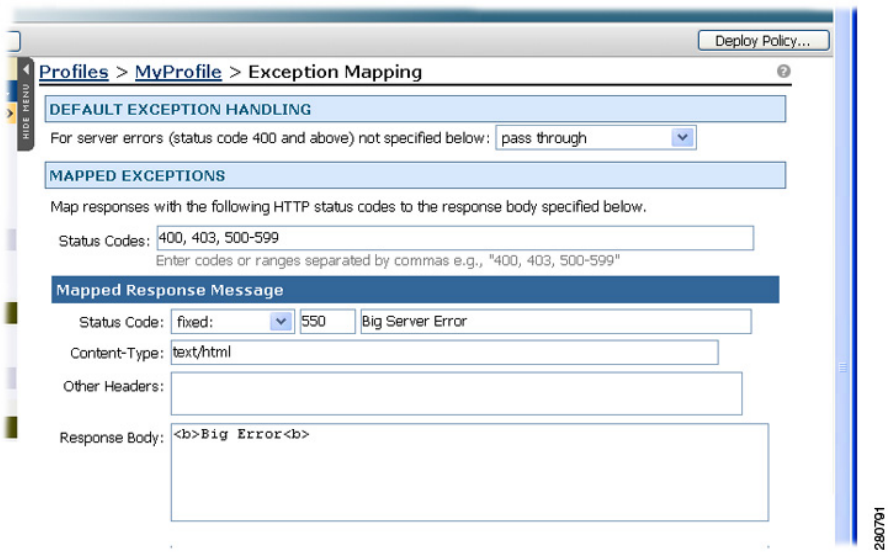
Step 1 Click the **Profiles** link in the Manager navigation menu, and then click on the name of the profile you created, **Poison Oak Traffic Validation**.

Notice that the HTTP Exception Mapping active security feature is currently disabled in this profile.

Step 2 Click the **edit** link next to HTTP Exception Mapping.

The **HTTP Exception Mapping** settings page appears, as shown in Figure 14-1.

Figure 14-1 HTTP Exception Mapping



Step 3 In the **Default Exception Mapping** section, leave the **For service errors ... not specified below** menu set to the **pass through** value that is its default setting.

This setting indicates that the other controls on the page specify a subset of errors to replace with your own message. If you wanted to replace all errors from the backend service with a generic 500 message, you would choose the **replace with generic 500** item from this menu.

Step 4 In the **Mapped Exceptions** section of the page, type **400, 403, 500-599** into the **Status Codes** field.

This value specifies that the ACE Web Application Firewall is to replace any of the HTTP 400, 403, and 500 through 599 errors with a custom message.

Step 5 In the **Mapped Response Message** section, choose the **fixed** value from the **Status Code** menu.

This setting indicates the status code for the response. Other than a numeric value, the code value is not restricted in any way, which provides flexibility in how the response is constructed. For example, you could return a 200 response (indicating a successful response) with the error message in the HTML body of the response.

In this case, we'll keep the default status code number, 500, and title, Server Error.

Step 6 In the **Response Body** field, type a message body such as "Big Error". In an actual implementation, you would type the text that you want to appear in the client's browser.

Step 7 Click **Save Changes**.

The **HTTP Exception Mapping** section of your profile now indicates that exception mapping is enabled and lists the mapped HTTP exceptions.

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

In the next section, you'll test your mapping configuration.

Testing HTTP Exception Mapping

Once you've configured HTTP Exception Mapping and deployed the new policy, you can see your new exception mapping rule at work.

When testing HTTP exception mapping, it is important to note that Microsoft Internet Explorer browser includes a feature that modifies HTTP errors presented in the browser. The feature, called "show friendly HTTP error messages," is automatically enabled in Internet Explorer. (Mozilla FireFox shows the original error message from the backend system.) To test the error mapping with Internet Explorer, you will first need to disable friendly HTTP error messages. Disable it by unchecking the option labelled **Show friendly HTTP error messages** in the **Advanced** tab of the **Internet Options** dialog in Internet Explorer.

To test error mapping:

Step 1 In your web browser, access the payment.do page of the Poison Oak sample application by requesting it directly in the browser:

http://<*Oak_VirtualURL*>/payment.do

Step 2 Ensure that instead of displaying a detailed stack trace, your browser displays the generic error message you configured.

CHAPTER 15

Server Header Masking

An attacker does not need to intercept a stack trace to identify implementation details that may facilitate a successful attack. Headers returned in a successful response often reveal clues to the architecture of the backend system. Distinctive formatting styles, error messages, header names or values, and other details may allow an attacker to identify a backend system.

For example, the following example shows message headers similar to that generated by widely used infrastructure. Boldface items may provide useful clues to an attacker:

```
HTTP/1.1 200 ok
Date: Wed, 24 Oct 2007 01:20:18 GMT
Server: Server Name 4.1 SP2 Fri Dec 5 15:01:51 PST 2003 316284
Content-encoding: gzip
Content-length: 6971
Content-type: text/html;       charset=utf-8
Expires: Wed, 24 Oct 2007 01:40:18 GMT
Last-Modified: Wed, 24 Oct 2007 01:20:18 GMT
Set-Cookie: PS_TOKENEXPIRE=24_Oct_2007_01:20:18_GMT; path=/
```

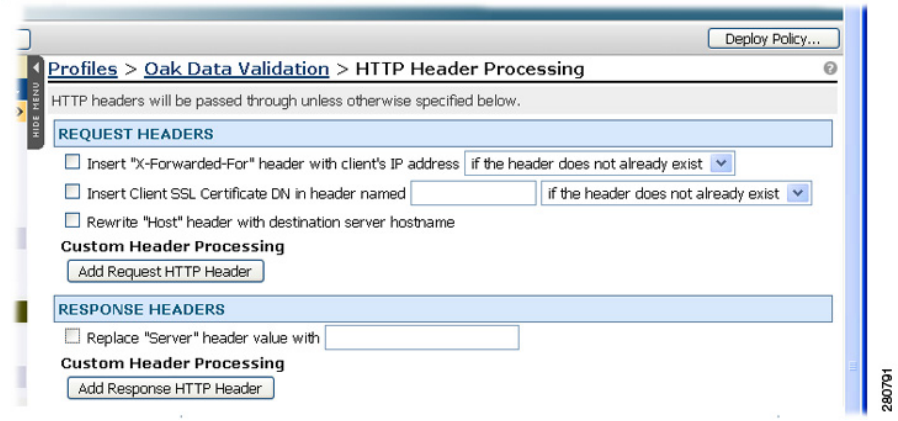
As shown, the Server header provides very specific details (although generalized for this example) that an attacker can use as a starting point for identifying potential vulnerabilities. Extra whitespace and unusual capitalization in the headers suggest a particular backend infrastructure. Non-standard capitalization of several headers is a hint that custom code is running. Often, a custom code module provides a viable entry point for an attacker, either because it utilizes a framework whose weaknesses are known to the attacker or because the custom module's implementation may not follow particularly secure coding practices.

By default, valid HTTP headers pass through the ACE Web Application Firewall without modification. To avoid revealing details to attackers, the **HTTP Header Processing** feature of the Cisco ACE Web Application Firewall may be used to add, remove, or modify message headers as you specify. You can use this feature to sanitize and format headers, such as in the previous example, which would be processed as follows:

```
HTTP/1.1 200 OK
Date: Wed, 24 Oct 2007 01:20:18 GMT
Server: Webserver
Content-Encoding: gzip
Content-Length: 6971
Content-Type: text/html; charset=utf-8
Expires: Wed, 24 Oct 2007 01:40:18 GMT
Last-Modified: Wed, 24 Oct 2007 01:20:18 GMT
Set-Cookie: PS_TOKENEXPIRE=24_Oct_2007_01:20:18_GMT; path=/
```

Figure 15-1 shows the HTTP Header Processing page, which enables you to manipulate the headers that would commonly need to be modified by a reverse proxy, such as the `Server` and `X-Forwarded-For` header. Additionally, you can use this page to configure special handling for any named HTTP header.

Figure 15-1 HTTP Header Processing page



Viewing HTTP Headers

This section describes how to use the "Live HTTP Headers" add-on for Firefox to view HTTP headers. However, you are not required to use this particular tool; any tool that enables you to view HTTP headers is suitable for this activity.

To view HTTP headers with Live HTTP Headers:

Step 1 If you have not done so already, install the **Live HTTP Headers** add-on in your Firefox browser and restart Firefox. After the tool is installed, a **Live HTTP headers** item should appear in the **Tools** menu of the Firefox browser.

Step 2 Configure the **Live HTTP Headers** tool to restrict its display of headers to only those sent by the ACE Web Application Firewall, as follows:

a. Choose the **Live HTTP Headers** menu item from the **Tools** menu of the **Firefox** browser.

b. In the **Live HTTP Headers** window, click the **Config** tab.

The tab shows controls that you can use to restrict the display of headers.

c. Click the **Filter URLs with regexp** checkbox, placing a checkmark in it.

d. Type the following expression into the **Filter URLs with regexp** field:

<*virtualHostname*>$|<*exampleSvcHostname*>$

In this expression, replace *virtualSvcHostname* with the hostname and port of the virtual URL at which the virtual web application publishes the example service. Replace *exampleSvcHostname* with the hostname and port at which the application is served on your trusted network. This expression allows you to view headers returned directly from the example service or from the virtual web application.

e. Allow all other settings in the **Config** tab to remain at their default values.

f. Click the **Headers** tab.

g. In the **HTTP Headers** pane, click the **Clear** button, if enabled. The **HTTP Headers** pane erases its current contents.

Step 3 Clear your web browser's cache.

Step 4 Use your web browser to visit the virtual URL of the Oak Insurance home page.

The **HTTP Headers** pane of the **Live HTTP Headers** tool displays the GET request that your browser sent, as well as the response from the virtual web application. If the pane does not display any headers, click your browser's reload button to resend the request. If headers still do not appear, you may need to modify the **Config > Filter URLs with regexp** value. If needed, clear the field to capture headers for all requests from the browser.

Step 5 Examine the response for a **Server** header. Notice that the **Server** header indicates that Jetty(6.1.7) served the page.

In the next section, you'll configure the HTTP Header Processing rule of the Poison Oak Traffic Validation profile to replace this descriptive Server header with one that does not reveal the implementation of the web server.

Modifying HTTP Headers

To configure the HTTP Header Processing rule of the Poison Oak Traffic Validation profile to replace the Server header returned by the example service with a more generic one:

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 **HTTP Header Processing** active security feature is disabled.

Step 2 Click the **edit** link next to **HTTP Header Processing**.

Step 3 In the **Response Headers** section, click the **Replace Server header value** with checkbox.

Step 4 Type **ACE WAF** into the text field.

Step 5 Click **Save Changes**.

The information page for the specified profile shows that the HTTP Header Processing rule is enabled in the working policy.

Step 6 Click the **Deploy Policy** button and complete the deployment as described in Chapter 6, "Deploying and Testing the Policy".

You can now test the HTTP Header Processing settings, as described in the following section.

Demonstrating HTTP Header Processing

To see how the **HTTP Header Processing** rule modifies the Server header content, follow these steps:

Step 1 Clear your browser's cache.

Step 2 In your browser, access the Poison Oak home page through the ACE Web Application Firewall.

Step 3 From the browser's **Tools** menu, choose the **Live HTTP Headers** menu item.

The **Live HTTP Headers** window displays the server response to your browser's GET request for the Oak Insurance home page.

Instead of identifying the server application that served the page, the **Server** header provides "ACE WAF" as its content.

Note If you see no content or too much content in the **HTTP Live Headers** window, try clicking its **Clear** button and then forcing a refresh of your browser window.

The ACE Web Application Firewall has rewritten the server header value as specified by your configuration. In the next section, you'll try other security features of the profile.

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**.

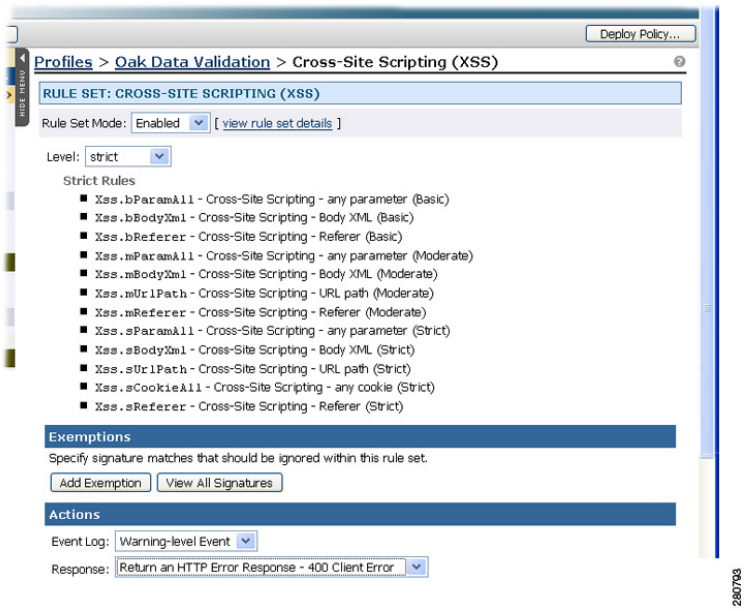
Notice that the Cross-Site Scripting (XSS) rule set is currently disabled in this profile. The Cross-Site Scripting (XSS) rule set is listed with the Message Inspection rules under the Firewall Configuration settings.

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.