Getting Started Guide, Cisco ACE Application Control Engine Module

Software Version A5(1.0)
September 2011

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-25357-01

CONTENTS

Preface

This preface describes the audience, organization, and conventions of the Cisco Application Control Engine Module Getting Started Guide (italicize name). It also provides information on how to obtain related documentation.

This preface contains the following major sections:

- Audience
- Organization
- Related Documentation
- Conventions
- Obtaining Documentation, Obtaining Support, and Security Guidelines

Audience

This guide is intended for the following trained and qualified service personnel who are responsible for configuring the ACE:

- Web master
- System administrator
- System operator

Organization

This guide is organized as follows:

Chapter	Description
Chapter 1, Overview	Provides an overview of the major functions and features of the ACE.
Chapter 2, Setting Up an ACE	Provides procedures to initially configure the ACE to allow the passing of traffic and remote access.
Chapter 3, Configuring Virtualization	Provides procedures to partition the ACE into virtual contexts for more efficient operation.

Chapter	Description
Chapter 4, Configuring Access Control Lists	Provides procedures to configure an access control list in an ACE to secure your network.
Chapter 5, Configuring Role-Based Access Control	Provides procedures to configure a user with permission to perform limited operations and access a subset of your network.
Chapter 6, Configuring Server Load Balancing	Provides procedures to configure the ACE to allow basic server load balancing.
Chapter 7, Configuring a Load-Balancing Predictor	Provides procedures to select a predefined predictor for server load balancing.
Chapter 8, Configuring Server Persistence Using Stickiness	Provides procedures to configure server persistence for requests from a client using stickiness.
Chapter 9, Configuring SSL Security	Provides procedures to configure SSL security for your network.
Chapter 10, Configuring Health Monitoring Using Health Probes	Provides procedures to configure server health monitoring using health probes.
Chapter 11, Configuring Route Health Injection	Provides procedures to configure route health injection (RHI).
Chapter 12, Configuring Redundant ACE Modules	Provides procedures for configuring fault tolerance in your network.
Chapter 13, Configuring Bridged Mode	Provides procedures for configuring your ACE to operate at Layer 2 with the client-side VLAN and the server-side VLAN in the same IP subnet.
Chapter 14, Configuring One-Arm Mode	Provides procedures for configuring your ACE to operate in a network where the clients and the servers are in the same VLAN.

If you are already familiar with the ACE appliance and would like to quickly set up the device for basic server load balancing, you can follow the configuration procedures in the following chapters:

- Chapter 2, Setting Up an ACE
- Chapter 3, Configuring Virtualization
- Chapter 6, Configuring Server Load Balancing

The remaining chapters allow you to explore additional capabilities of the ACE.

Related Documentation

In addition to this document, the ACE documentation set includes the following documents:

Document Title	Description
Release Note for the Application Control Engine Module	Provides information about operating considerations, caveats, and CLI commands for the ACE.
Installation Note, Cisco ACE Application Control Engine ACE30 Module	Provides information for installing the ACE.

Document Title	Description
Administration Guide, Cisco ACE Application Control Engine	Describes how to perform the following administration tasks on the ACE: • Setting up the ACE • Establishing remote access • Managing software licenses • Configuring class maps and policy maps • Managing the ACE software • Configuring SNMP • Configuring redundancy • Configuring the XML interface • Upgrading the ACE software
Cisco CSM-to-ACE Conversion Tool User Guide	Describes how to use the CSM-to-ACE conversion tool to migrate Cisco Content Switching Module (CSM) running or startup configuration files to the ACE.
Cisco CSS-to-ACE Conversion Tool User Guide	Describes how to use the CSS-to-ACE conversion tool to migrate Cisco Content Services Switches (CSS) running or startup configuration files to the ACE.
Cisco Application Control Engine (ACE) Configuration Examples Wiki	Provides examples of common configurations for load balancing, security, SSL, routing and bridging, virtualization, and so on.
Cisco Application Control Engine (ACE) Troubleshooting Wiki	Describes the procedures and methodology in wiki format to troubleshoot the most common problems that you may encounter during the operation of your ACE.
Command Reference, Cisco ACE Application Control Engine	Provides an alphabetical list and descriptions of all CLI commands by mode, including syntax, options, and related commands.
Routing and Bridging Guide, Cisco ACE Application Control Engine	Describes how to perform the following routing and bridging tasks on the ACE: • VLAN interfaces • IPv6, including transitioning IPv4 networks to IPv6, IPv6 header format, IPv6 addressing, and suported protocols. • Routing • Bridging • Dynamic Host Configuration Protocol (DHCP)

Document Title	Description
Server Load-Balancing Guide, Cisco ACE Application Control Engine	Describes how to configure the following server load-balancing tasks on the ACE: • Real servers and server farms • Class maps and policy maps to load balance traffic to real servers in server farms • Server health monitoring (probes) • Stickiness • Firewall load balancing • TCL scripts
Security Guide, Cisco ACE Application Control Engine	Describes how to perform the following ACE security configuration tasks: • Access control lists (ACLs) • User authentication and accounting using a Terminal Access Controller Access Control System Plus (TACACS+), Remote Authentication Dial-In User Service (RADIUS), or Lightweight Directory Access Protocol (LDAP) server • Application protocol and HTTP deep packet inspection • TCP/IP normalization and termination parameters • Network address translation (NAT)
SSL Guide, Cisco ACE Application Control Engine	Describes how to configure the following SSL tasks on the ACE: • SSL certificates and keys • SSL initiation • SSL termination • End-to-end SSL
System Message Guide, Cisco ACE Application Control Engine	Describes how to configure system message logging on the ACE. This guide also lists and describes the system log (syslog) messages generated by the ACE.
User Guide, Cisco Application Networking Manager	Describes how to use Cisco Application Networking Manager (ANM), a networking management application for monitoring and configuring network devices, including the ACE.
Virtualization Guide, Cisco ACE Application Control Engine	Describes how to operate your ACE in a single context or in multiple contexts and how to configure Role-Based Access Control.

Conventions

This publication uses the following conventions:

Convention	Description
boldface font	Commands, command options, and keywords are in **boldface**. Bold text also indicates a command in a paragraph.
italic font	Arguments for which you supply values are in *italics*.
[ ]	Elements in square brackets are optional.
{ x | y | z }	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	An unquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
`screen` font	Terminal sessions and information the system displays are in `screen` font.
`boldface screen` font	Information you must enter on a command line is in **`boldface screen`** font.
italic screen font	Arguments for which you supply values are in *italic screen* font.
^	The symbol ^ represents the key labeled Control—for example, the key combination ^D in a screen display means hold down the Control key while you press the D key.
< >	Nonprinting characters, such as passwords are in angle brackets.

1. A numbered list indicates that the order of the list items is important.
    a. An alphabetical list indicates that the order of the secondary list items is important.

- A bulleted list indicates that the order of the list topics is unimportant.
    - An indented list indicates that the order of the list subtopics is unimportant.

Notes use the following conventions:

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.

Cautions use the following conventions:

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

Obtaining Documentation, Obtaining Support, and Security Guidelines

For information on obtaining documentation, obtaining support, providing documentation feedback, security guidelines, and also recommended aliases and general Cisco documents, 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

CHAPTER 1

Overview

This guide provides the following information:

- An overview of the major functions and features of the Cisco Application Control Engine (ACE) module
- Instructions on how to initially configure the ACE to allow traffic and basic load balancing
- Instructions on how to configure the ACE to provide various scalability and security capabilities
- References to find the information in the documentation set

Note All configuration examples in this guide are based on IPv4. IPv6 is supported on the ACE module in software releases A5(1.0) and later. For information about configuring and using IPv6 with your ACE module, see the *Routing and Bridging Guide, Cisco ACE Application Control Engine*.

The Cisco Application Control Engine (ACE) module performs server load balancing, network traffic control, service redundancy, resource management, encryption and security, and application acceleration and optimization, all in a single network device.

This chapter contains the following sections:

- ACE Technologies
- Setting Up an ACE
- Creating Virtual Contexts
- Configuring Access Control Lists
- Configuring Role-Based Access Control
- Configuring a Virtual Server
- Configuring a Load-Balancing Predictor
- Configuring Server Persistence Using Stickiness
- Configuring SSL Security
- Configuring Health Monitoring Using Health Probes
- Configuring Route Health Injection
- Configuring Redundancy
- Configuring Bridged Mode
- Configuring One-Armed Mode
- Where to Go Next

ACE Technologies

Server load balancing helps ensure the availability, scalability, and security of applications and services by distributing the work of a single server across multiple servers.

When you configure server load balancing on your ACE, the ACE decides which server should receive a client request such as a web page or a file. The ACE selects a server that can successfully fulfill the client request most effectively, without overloading the selected server or the overall network.

Table 1-1 shows the ACE technologies that provide network availability, scalability, and security at both the device and network services levels.

Table 1-1 ACE Technologies

Level	Availability	Scalability	Security
Device	Device Setup	Virtual Contexts	Access Control Lists
	Redundancy	Role-Based Access Control	
Network Services	Virtual Servers	Load Balancing Predictors	SSL
	Health Probes	Server Persistence Using Stickiness	Access Control Lists
		Role-Based Access Control	

At the device level, the ACE provides high network availability by supporting:

- Device redundancy— High availability, which allows you to set up a peer ACE device in the configuration so that if one ACE becomes inoperative, the other ACE can take its place immediately.
- Scalability—Virtualization, which allows you to partition one ACE device into independent virtual devices, each with its own resource allocation.
- Security—Access control lists, which restrict access from certain clients or to certain network resources.

At the network service level, the ACE provides advance services by supporting:

- High services availability—High-performance server load balancing, which allows you to distribute client requests among physical servers and server farms, and provide health monitoring at the server and server farm levels through implicit and explicit health probes.
- Scalability—Virtualization, which allows you to use advanced load-balancing algorithms (predictors) to distribute client requests among the virtual devices configured in the ACE. Each virtual device includes multiple virtual servers. Each server forwards client requests to one of the server farms. Each server farm can contain multiple physical servers.

  Although the ACE can distribute client requests among hundreds or even thousands of physical servers, it can also maintain server persistence. With some e-commerce applications, all client requests within a session are directed to the same physical server so that all the items in one shopping cart are contained on one server.
- Services-level security—Allows you to establish and maintain a Secure Sockets Layer (SSL) session between the ACE and its peer, which provides secure data transactions between clients and servers.

Setting Up an ACE

To set up an ACE, you first establish a connection to the ACE and perform the initial device setup to prepare the ACE for providing application networking services. For more information, see Chapter 2, Setting Up an ACE.

Creating Virtual Contexts

You partition the ACE device into multiple virtual contexts, each with its own resource allocation. For more information, see Chapter 3, Configuring Virtualization.

Configuring Access Control Lists

You control access to your network resources to guarantee that only desired traffic passes through, and that the appropriate users can access the network resources they need.

You use Access Control Lists (ACLs) to secure your network by permitting or denying traffic to or from a specific IP address or an entire network.

You must configure an ACL for each interface on which you want to permit connections. Otherwise, the ACE will deny all traffic on that interface. An ACL consists of a series of ACL permit-or-deny entries, with criteria for the source IP address, destination IP address, protocol, port, or protocol-specific parameters. Each entry permits or denies inbound or outbound network traffic to the parts of your network specified in the entry.

This guide provides an example of ACL configuration at the device level (see Chapter 4, Configuring Access Control Lists). To learn how to configure ACLs at the network services level, or how to configure more granular access control security, see the *Security Guide, Cisco ACE Application Control Engine*.

Configuring Role-Based Access Control

You can manage the complexity of large-network security administration by defining the commands and resources available to each user through Role-Based Access Control (RBAC). RBAC supports network security at both the device and network services levels by defining physical or virtual resources in a domain that the user can access.

For more information, see Chapter 5, Configuring Role-Based Access Control.

Configuring a Virtual Server

You can configure a virtual server to intercept web traffic to a website and allow multiple real servers (physical servers) to appear as a single server for load-balancing purposes.

Table 1-2 illustrates how the ACE supports scalability through virtual contexts, virtual servers, server farms, and real servers.

Table 1-2 ACE Scalability

Physical Device	Virtual Device	Virtual Server	Server Farm	Real Server
ACE	Virtual Context 1	Virtual Server A	Server Farm A	Real Server A1
				Real Server A2
				……
				Real Server An
			Backup Server Farm a	Real Server a1
				Real Server a2
				……
				Real Server an
		Virtual Server B	Server Farm B	Real Server B1
				Real Server B2
				……
				Real Server Bn
	Virtual Context 2	Virtual Server C	Server Farm C	Real Server C1
				Real Server C2
				……
				Real Server Cn
		Virtual Server D	Server Farm D	Real Server D1
				Real Server D2
				……
				Real Server Dn
	……	……	……	……

You can partition your ACE into multiple virtual contexts, each of which has its own set of policies, interfaces, and resources. A virtual server is bound to physical resources that run on a real server in a server farm.

Real servers relate to the actual, physical servers on your network. They can be configured to provide client services or as backup servers.

Related real servers are grouped into server farms. Servers in the same server farm often contain identical content (referred to as mirrored content) so that if one server becomes inoperative, another server can take over its functions immediately. Mirrored content also allows several servers to share the load during times of increased demand.

For more information, see Chapter 6, Configuring Server Load Balancing.

Configuring a Load-Balancing Predictor

You can distribute incoming client requests among the servers in a server farm by defining load-balancing rules called predictors using IP address and port information.

When a client requests an application service, the ACE performs server load balancing by deciding which server can successfully fulfill the client request in the shortest amount of time without overloading the server or server farm. Some sophisticated predictors take into account factors such as a server's load, response time, or availability, allowing you to adjust load balancing to each application's particular past.

For more information, see Chapter 7, Configuring a Load-Balancing Predictor.

Configuring Server Persistence Using Stickiness

You can configure the ACE to allow the same client to maintain multiple simultaneous or subsequent TCP or IP connections with the same real server for the duration of a session. A session is defined as a series of interactions between a client and a server over some finite period of time (from several minutes to several hours). This server persistence feature is called stickiness.

Many network applications require that customer-specific information be stored persistently across multiple server requests. A common example is a shopping cart used on an e-commerce site. With server load balancing in use, it could potentially be a problem if a back-end server needs information generated at a different server during a previous request.

Depending on how you have configured server load balancing, the ACE sticks a client to an appropriate server after it has determined which load-balancing method to use. If the ACE determines that a client is already stuck to a particular server, then the ACE sends subsequent client requests to that server, regardless of the load-balancing criteria. If the ACE determines that the client is not stuck to a particular server, it applies the normal load-balancing rules to the request.

The combination of the predictor and stickiness enables the application to have scalability, availability, and performance even with persistence for transaction processing.

For more information, see Chapter 8, Configuring Server Persistence Using Stickiness.

Configuring SSL Security

You can provide authentication, encryption, and data integrity in a Public Key Infrastructure (PKI) by using the SSL security protocol in your network.

SSL configuration in an ACE establishes and maintains an SSL session between the ACE and its peer, enabling the ACE to perform its load-balancing tasks on the SSL traffic. These SSL functions include server authentication, private-key and public-key generation, certificate management, and data packet encryption and decryption.

For more information, see Chapter 9, Configuring SSL Security.

Configuring Health Monitoring Using Health Probes

Application services require monitoring to ensure availability and performance. You can configure the ACE to track the health and performance of your servers and server farms by creating health probes. Each health probe that you create can be associated with multiple real servers or server farms.

When you enable ACE health monitoring, the module periodically sends messages to the server to determine the server status. The ACE verifies the server's response to ensure that a client can access that server. The ACE can use the server's response to place the server in or out of service. In addition, the ACE can use the health of servers in a server farm to make reliable load-balancing decisions.

For more information, see Chapter 10, Configuring Health Monitoring Using Health Probes.

Configuring Route Health Injection

Route Health Injection (RHI) allows you to send traffic to a preferred route by instructing the ACE to advertise a VLAN or a VIP throughout the network. By default, the ACE advertises the VLAN of the VIP interface for RHI. You can also configure the ACE to:

- Advertise the VIP of the virtual server
- Advertise a VLAN for RHI that is different from the VIP VLAN when there is an intervening device (for example, a Cisco Firewall Services Module) between the ACE and the supervisor engine

When the ACE advertises a VIP or a VLAN, the associated route is added to the routing table of the devices on the network. For more information, see Chapter 11, Configuring Route Health Injection.

Configuring Redundancy

Redundancy, sometimes referred to as high availability (HA) or fault tolerance (FT), uses a maximum of two ACEs in the same Catalyst 6500 series switch or Cisco 7600 series router or in separate switches or routers to ensure that your network remains operational even if one of the modules becomes unresponsive. Redundancy ensures that your network services and applications are always available.

Note Redundancy is not supported between an ACE module and an ACE appliance operating as peers. Redundancy must be of the same ACE device type and software release.

Redundancy provides seamless switchover of flows in case an ACE becomes unresponsive or a critical host, interface, or a Hot Standby Router Protocol (HSRP) group fails. Redundancy supports the following network applications that require fault tolerance:

- Mission-critical enterprise applications
- Banking and financial services
- E-commerce
- Long-lived flows such as FTP and HTTP file transfers

For more information, see Chapter 12, Configuring Redundant ACE Modules.

Configuring Bridged Mode

When you configure the ACE in bridge or transparent mode, the ACE bridges the traffic between two VLANs. Each pair of VLANs is a subnet. A router is required to route traffic between these bridged subnets. When deployed in redundant pairs, there is no loop because only one device is forwarding at any given time.

In bridge mode, the ACE is said to be transparent because it acts as a bump in the wire instead of a next hop. The ACE operates at Layer 2 and is not dependent on IP addresses to pass traffic as it is in routed mode.

When operating an ACE in transparent mode, it is desirable to pass Bridge Protocol Data Units (BPDUs) to prevent bridge loops in case a redundant pair becomes active. By default BPDUs do not flow through an ACE. You have to explicitly enable this capability.

For more information, see Chapter 13, Configuring Bridged Mode.

Configuring One-Armed Mode

You typically configure the ACE with a client-side VLAN and a server-side VLAN. This configuration is referred to as two-armed mode.

You can also configure the ACE so that the clients, the ACE, and the servers are all in the same VLAN. This configuration is referred to as one-armed mode.

For more information, see Chapter 14, Configuring One-Arm Mode.

Where to Go Next

This chapter has provided you with an overview of the ACE module, its technologies, and its major features. In the next chapter, you will learn how to set up an ACE and to configure it for remote access using a management policy.

CHAPTER 2

Setting Up an ACE

This chapter describes how to set up a Cisco Application Control Engine (ACE) module for remote management.

This chapter contains the following topics:

- Information About Setting up an ACE
- Prerequisites for Setting Up an ACE
- Guidelines and Limitations
- Setting Up an ACE
- Configuration Example for Setting Up an ACE
- Where to Go Next

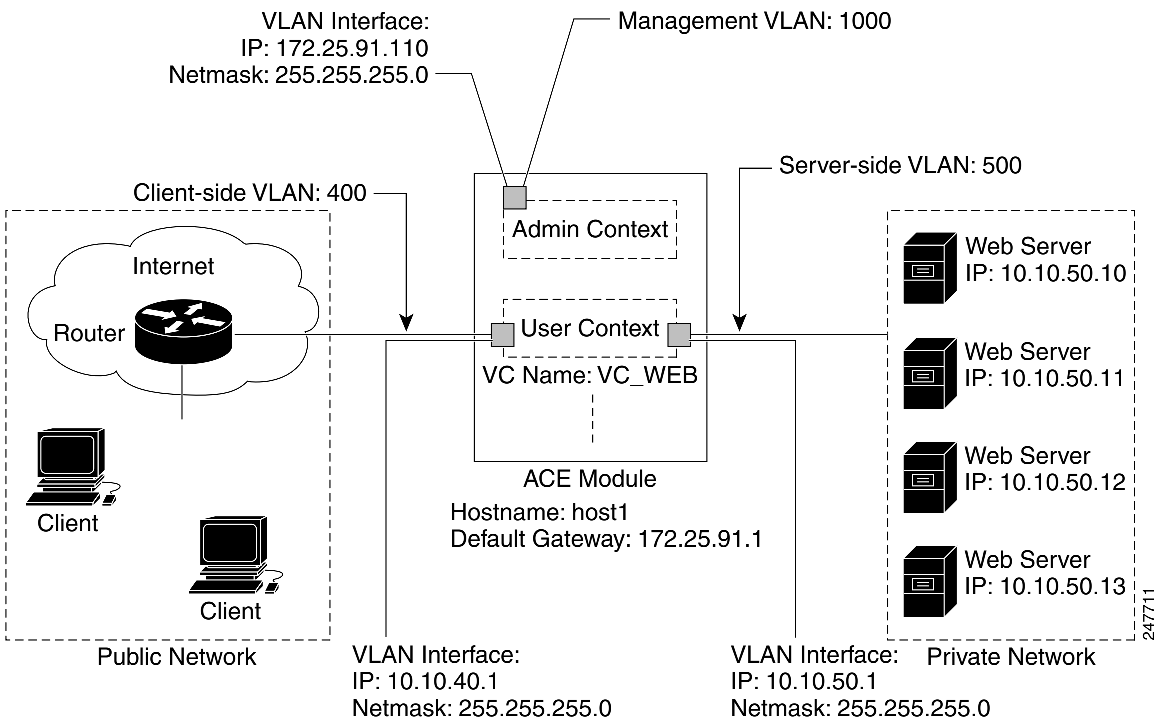
Note All configuration examples in this guide are based on IPv4. IPv6 is supported on the ACE module in software releases A5(1.0) and later. For information about configuring and using IPv6 with your ACE module, see the A5(1.0) *Routing and Bridging Guide, Cisco ACE Application Control Engine*.

Information About Setting up an ACE

After reading this chapter, you should have a basic understanding of how to set up an ACE and configure it for remote access through a management interface.

This chapter describes how to set up an ACE using the example network setup shown in Figure 2-1.

Figure 2-1 Example Network Setup



The configuration of the example setup is as follows:

- VLAN 1000 is assigned to the ACE and is used for management traffic for the Admin context.

> **Note** A virtual local area network (VLAN) is a logical division of a computer network within which information can be transmitted for all devices to receive. VLANs enable you to segment a switched network so that devices in one VLAN do not receive information packets from devices in another VLAN.

- VLAN 400 is assigned to the ACE and is used for client-side traffic.
- VLAN 500 is assigned to the ACE and is used for server-side traffic.
- A management VLAN interface is configured for the Admin context with VLAN 1000 and IP address 172.25.91.110.
- A client-side VLAN interface is configured for the user context VC_WEB with VLAN 400 and IP address 10.10.40.1.
- A server-side VLAN interface is configured for the user context VC_WEB with VLAN 500 and IP address 10.10.50.1.
- Four web servers are available to the ACE for load-balancing client requests.

Prerequisites for Setting Up an ACE

Setting up an ACE has the following prerequisites:

- Complete the ACE installation instructions as described in the *Installation Note, Cisco ACE Application Control Engine ACE30 Module*.
- Contact your network administrator to determine which VLANs and addresses are available for use by the ACE.

Guidelines and Limitations

You can assign a maximum of 16 VLAN groups to one ACE.

Setting Up an ACE

This section includes the following topics:

- Task Flow for Setting Up an ACE
- Configuring VLANs for the ACE Using Cisco IOS Software
- Sessioning and Logging in to the ACE from the Supervisor Engine
- Assigning a Name to the ACE
- Configuring a Management VLAN Interface on the ACE
- Configuring a Default Route
- Configuring Remote Management Access to the ACE
- Accessing the ACE through a Telnet Session

Task Flow for Setting Up an ACE

Follow these steps to set up your ACE:

Step 1 Configure VLANs for your ACE using IOS software.

Step 2 Create a session between the Catalyst 6500 series switch or Cisco 7600 series router and the ACE and log in to the ACE from the supervisor engine.

Step 3 Assign a name to your ACE.

Step 4 Configure a management VLAN interface.

Step 5 Configure a default route.

Step 6 Configure remote management access to your ACE.

Step 7 Access your ACE through a Telnet session.

Configuring VLANs for the ACE Using Cisco IOS Software

Before the ACE can receive traffic from the supervisor engine in the Catalyst 6500 series switch or in a Cisco 7600 series router (an ACE20-MOD-K9 module only), you must create VLAN groups on the supervisor engine, and then assign the groups to the ACE. After you configure the VLAN groups on the supervisor engine for the ACE, you can configure the VLAN interfaces on the ACE.

In Cisco IOS software, you can create one or more VLAN groups, and then assign the groups to the ACE. For example, you can assign all the VLANs to one group, or you can create a group for each customer.

You cannot assign the same VLAN to multiple groups; however, you can assign multiple groups to an ACE. VLANs that you want to assign to multiple ACEs, for example, can reside in a separate group from VLANs that are unique to each ACE.

To configure the VLANs for the ACE using the Cisco IOS software, perform the following steps:

Procedure

	Command	Purpose

Step 1	`linux$` **`telnet`** *`ip_address`*
	`User Access Verification`
	
	`Password:` **`cisco`**
	`Router>`
	
	`Example:`
	`linux$ telnet 192.168.12.15`
	`User Access Verification`
	
	`Password: cisco`
	`Router>`
		Connects to the supervisor engine to open a session. Enter the IP address of the supervisor engine in dotted-decimal notation.

Step 2	**`enable`**
	
	`Example:`
	`Router> enable`
	`Password: cisco`
	`Router #`
		Enters Cisco IOS privileged mode.

Step 3	**`config`**
	
	`Example:`
	`Router# config`
	`Router(config)#`
		Enters configuration mode.

Step 4	**`svclc vlan-group`** *`group_number vlan_range`*
	
	`Example:`
	`Router# config`
	`Router(config)# svclc vlan-group 50 40,`
	`41,60,100,400,500,1000`
		Assigns VLANs to a group. VLAN numbers have the range 2 to 1000 and 1025 to 4094.

Step 5	**`svclc module`** *`slot_number`* **`vlan-group`** *`group_number_range`*
	
	`Example:`
	`Router(config)# svclc module 5`
	`vlan-group 50`
		Assigns VLAN group 50 to the ACE. You can assign a maximum of 16 VLAN groups to an ACE.

	Command	Purpose
Step 6	**svclc multiple-vlan-interfaces** **Example:** Router(config)# svclc multiple-vlan-interfaces	(Optional) Configures a switched virtual interface (SVI) on the Multilayer Switch Feature Card (MSFC) and assigns the SVI to the ACE. The **svclc multiple-vlan-interfaces** command allows you to configure multiple SVIs on the MSFC and assign them to the ACE, one for each context in the ACE.
Step 7	**interface vlan** *vlan_id* **Example:** Router(config)# interface vlan 55 ROuter(config-if)#	Enters interface configuration mode for the specified VLAN.
Step 8	**ip address** *ip_address netmask* **Example:** Router(config-if)# ip address 10.1.1.1 255.255.255.0	Assigns an IP address to the VLAN interface.
Step 9	**no shut** **Example:** Router(config-if)# no shut	Enables the interface.
Step 10	**exit** **Example:** Router(config-if)# exit Router(config)# exit Router#	Exits the current configuration mode and returns to the previous CLI mode. You can press Ctrl-G (same as **exit** command) to exit the current config mode. You can also press Ctrl-Z to return to Exec mode from any configuration mode.
Step 11	**show svclc module** *slot_number* **vlan-group** **Example:** Router# show svclc module 5 50	Displays VLAN group numbers for the module in the specified slot.
Step 12	**show svclc vlan-group** group_number **Example:** Router# show svclc vlan-group 50	Displays the group configuration for the ACE and the associated VLANs.
Step 13	**copy running-config startup-config** **Example:** Router# copy running-config startup-config	(Optional) Copies the running-configuration file to the startup-configuration file.

Sessioning and Logging in to the ACE from the Supervisor Engine

You can session and log in to the ACE from the supervisor engine.

Restrictions

For security reasons, you must change the Admin password when you log in to the ACE for the first time. If you do not change the Admin password, the following will occur:

- You will not be able to log in to the ACE remotely using Telnet or SSH

- You will be restricted to using either a console connection or a session through the supervisor engine to access the ACE.

You must also change the password of the www user when you log in to the ACE for the first time. The www user is used internally by the ACE for the XML interface. If you do not change the www user password, the XML interface is inoperable.

Procedure

	Command	Purpose

Step 1	**session slot** *number1* **processor** *number2*
	
	Example:
	`Router# session slot 5 processor 0`
	`switch login:`
	`Password:`
		Establishes a session between the supervisor engine and the ACE.

Step 2	**admin**
	
	Example:
	`switch login: admin`
	`Password: admin`
	`Cisco Application Control Software (ACSW)`
	`TAC support: http://www.cisco.com/tac`
	`Copyright (c) 2002-2009, Cisco Systems, Inc. All rights reserved.`
	`The copyrights to certain works contained herein are owned by`
	`other third parties and are used and distributed under license.`
	`Some parts of this software are covered under the GNU Public`
	`License. A copy of the license is available at`
	`http://www.gnu.org/licenses/gpl.html.`
	`switch/Admin#`
		Logs you in to the ACE. For security reasons, you must change the Admin password when you log in to the ACE for the first time. You should also change the www user password if you intend to use the XML interface to configure the ACE at some point.

Step 3	**username** *name1* [password [0 | 5] {*password*}]
	
	Example:
	`switch/Admin# config`
	`switch/Admin(config)# username Admin password 0 cisco123`
	`switch/Admin(config)# exit`
	`switch/Admin#`
		Changes the Admin password.

Step 4	**username** *name1* [password [0 | 5] {*password*}]
	
	Example:
	`switch/Admin# config`
	`switch/Admin(config)# username www password 0 xmlsecret_801`
	`switch/Admin(config)# exit`
	`switch/Admin#`
		Changes the www user password.

Step 5	**terminal session-timeout** *number* **Example:** `switch/Admin# terminal session-timeout 0`	Prevents the current session from timing out.

	Command	Purpose

Step 6	**`login timeout`** *`number`*
	
	`Example:`
	`switch/Admin# config`
	`switch/Admin(config)# login timeout 0`
		Sets the inactivity timeout in the ACE. A value of 0 disables the inactivity timeout.

Step 7	**`exit`**
	
	`Example:`
	`switch/Admin(config)# exit`
	`switch/Admin#`
		Exits configuration mode.

Step 8	**`show terminal`** **`Example:`** `switch/Admin# show terminal`	Displays the terminal configuration, including the session timeout value.
Step 9	**`show login timeout`** **`Example:`** `switch/Admin# show login timeout`	Displays the login timeout value.
Step 10	**`copy running-config startup-config`** **`Example:`** `switch/Admin# copy running-config startup-config`	(Optional) Copies the running-configuration file to the startup-configuration file.

Assigning a Name to the ACE

The hostname is used for the command-line prompts and default configuration filenames. When you establish sessions to multiple devices, the hostname helps you to keep track of the ACE on which you are entering commands. By default, the hostname for the ACE is "switch."

Procedure

	Command	Purpose

Step 1	**`config`**
	
	`Example:`
	`switch/Admin# config`
	`switch/Admin(config)#`
		Enters configuration mode.

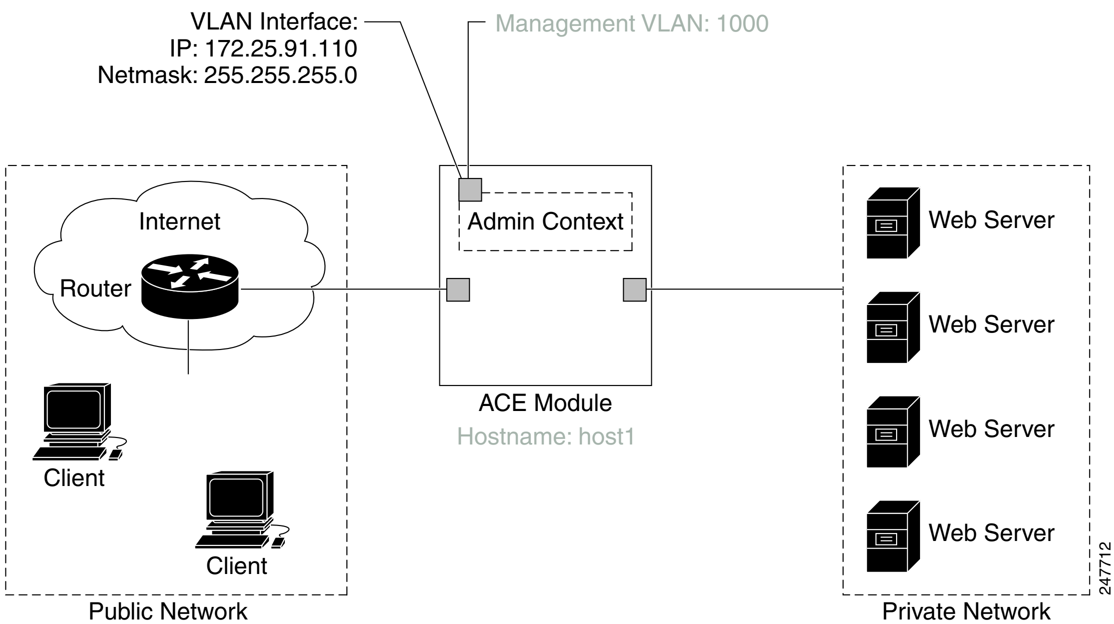
Step 2	**`hostname`** *`name`*
	
	`Example:`
	`switch/Admin(config)# hostname host1`
	`host1/Admin(config)#`
		Changes the hostname from "switch" to "host1."

Step 3	**`do copy running-config startup-config`**
	
	`Example:`
	`host1/Admin(config)# do copy`
	`running-config startup-config`
		(Optional) Copies the running-configuration file to the startup-configuration file. Note that the **do** command allows you to enter Exec mode commands in any configuration mode.

Configuring a Management VLAN Interface on the ACE

You can provide management connectivity to the ACE by assigning an IP address to the VLAN interface on the ACE. For the example configuration, you will assign an IP address 172.25.91.110 and a subnet mask of 255.255.255.0 to VLAN 1000, as shown in Figure 2-2 (previously configured settings are grayed out).

Figure 2-2 Configuring a Management VLAN Interface on the ACE



Procedure

	Command	Purpose

Step 1	**interface vlan** *vlan_id*
	
	Example:
	`host1/Admin(config)# interface vlan 1000`
	`host1/Admin(config-if)#`
		Configures VLAN 1000 on the ACE.

Step 2	**ip address** *ip_address netmask* **Example:** `host1/Admin(config-if)# ip address 172.25.91.110 255.255.255.0`	Assigns an IP address and network mask to the VLAN interface for management connectivity.
Step 3	**description** *string* **Example:** `host1/Admin(config-if)# description Management connectivity on VLAN 1000`	(Optional) Provides a description of the interface.

	Command	Purpose
Step 4	**no shutdown** **Example:** host1/Admin(config-if)# no shutdown	Enables the VLAN interface.
Step 5	**Ctrl-Z** **Example:** host1/Admin(config-if)# Ctrl-Z host1/Admin#	Returns to Exec mode directly from any configuration mode.
Step 6	**show running-config interface** **Example:** host1/Admin# show running-config interface	Displays the VLAN interface configuration.
Step 7	**show interface vlan** *vlan_id* **Example:** host1/Admin# show interface vlan 1000	Displays the status and statistics about the VLAN interface.
Step 8	**ping** *ip_adress* **Example:** host1/Admin(config-if)# ping 172.25.91.110	Verifies the connectivity of a remote host or server by sending ICMP echo messages from the ACE.
Step 9	**copy running-config startup-config** **Example:** host1/Admin# copy running-config startup-config	(Optional) Copies the running-configuration file to the startup-configuration file.

Configuring a Default Route

A default route identifies the IP address where the ACE sends all IP packets for which it does not have a route.

Procedure

	Command	Purpose
Step 1	**config** **Example:** switch/Admin# config switch/Admin(config)#	Enters configuration mode.
Step 2	**ip route** *src_ip_address dest_ip_address default_gateway* **Example:** host1/Admin(config)# ip route 0.0.0.0 0.0.0.0 172.25.91.1	Configures a default IP address where the ACE forwards all IP packets for which it does not have a route.

	Command	Purpose
Step 3	**do show ip route** **Example:** `host1/Admin(config)# do show ip route`	Displays the default route in the routing table.
Step 4	**do copy running-config startup-config** **Example:** `host1/Admin(config)# do copy running-config startup-config`	(Optional) Copies the running-configuration file to the startup-configuration file.

Example

The following example shows how to display the default route in the routing table:

```
host1/Admin(config)# do show ip route

Routing Table for Context Admin (RouteId 0)

   Codes: H - host,   I - interface
          S - static,      N - nat
          A - need arp resolve,      E - ecmp

Destination         Gateway          Interface         Flags
----------------------------------------------------------------------
0.0.0.0             172.25.91.1      vlan1000          S [0xc]
172.25.91.0/24      0.0.0.0          vlan1000          IA [0x30]

Total route entries = 2
```

Configuring Remote Management Access to the ACE

Before remote network access can occur on the ACE, you must create a traffic policy that identifies the network management traffic that can be received by the ACE.

Procedure

	Command	Purpose

Step 1	**class-map type management match-any** *name*
	
	Example:
	`host1/Admin(config)# class-map type management match-any REMOTE_ACCESS`
	`host1/Admin(config-cmap-mgmt)#`
		Creates a management-type class map named REMOTE_ACCESS that matches any traffic.

Step 2	**description** *string* **Example:** `host1/Admin(config-cmap-mgmt)# description Remote access traffic match`	(Optional) Provides a description for the class map.

	Command	Purpose

Step 3	**match protocol** *protocol* **any**
	
	Example:
	`host1/Admin(config-cmap-mgmt)# match protocol ssh any`
	`host1/Admin(config-cmap-mgmt)# match protocol telnet any`
	`host1/Admin(config-cmap-mgmt)# match protocol icmp any`
		Configures the match protocol to permit traffic based on the SSH, Telnet, and ICMP protocols for any source address.

Step 4	**exit**
	
	Example:
	`host1/Admin(config-cmap-mgmt)# exit`
	`host1/Admin(config)#`
		Exits class map management configuration mode.

Step 5	**policy-map type management first-match** *name*
	
	Example:
	`host1/Admin(config)# policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY`
	`host1/Admin(config-pmap-mgmt)#`
		Creates a policy map named REMOTE_MGMT_ALLOW_POLICY for traffic destined to an ACE interface.

Step 6	**class** *name* **Example:** `host1/Admin(config-pmap-mgmt)# class REMOTE_ACCESS`	Applies the previously created REMOTE_ACCESS class map to this policy.
Step 7	**permit** **Example:** `host1/Admin(config-pmap-mgmt-c)# permit`	Allows the ACE to receive the configured class-map management protocols.

Step 8	**exit**
	
	Example:
	`host1/Admin(config-pmap-mgmt-c)# exit`
	`host1/Admin(config-pmap-mgmt)# exit`
	`host1/Admin(config)#`
		Exits policy map class management configuration mode. Exits policy map management configuration mode.

Step 9	**interface vlan** *vlan_id*
	
	Example:
	`host1/Admin(config)# interface vlan 1000`
	`host1/Admin(config-if)#`
		Accesses interface configuration mode for the VLAN to which you want to apply the policy map.

Step 10	**service-policy input** *policy_name* **Example:** `host1/Admin(config-if)# service-policy input REMOTE_MGMT_ALLOW_POLICY`	Applies the REMOTE_MGMT_ALLOW_POLICY policy map to the interface.

Step 11	**Ctrl-Z**
	
	Example:
	`host1/Admin(config-if)# Ctrl-Z`
	`host1/Admin#`
		Returns to Exec mode from any configuration mode.

	Command	Purpose
Step 12	**show running-config class-map** **Example:** host1/Admin# show running-config class-map	Displays the class-map configuration.
Step 13	**show running-config policy-map** **Example:** host1/Admin# show running-config policy-map	Displays the policy-map configuration.
Step 14	**show service-policy** *name* **Example:** host1/Admin# show service-policy REMOTE_MGMT_ALLOW_POLICY	Displays the service-policy that you applied to the interface.
Step 15	**copy running-config startup-config** **Example:** host1/Admin# copy running-config startup-config	(Optional) Copies the running-configuration file to the startup-configuration file.

Accessing the ACE through a Telnet Session

After you have completed the previous configurations, you can use Telnet to access the ACEby using its IP address.

Procedure

	Command	Purpose
Step 1	**telnet** *ip_address* **Example:** remote_host# telnet 172.25.91.110 Trying 172.25.91.110... Open	Initiates a Telnet session from a remote host to the ACE. For example, access the ACE from the VLAN IP address of 172.25.91.110.
Step 2	host1 login: **admin** Password: **xxxxx** **Example:** host login: admin password: cisco123	Logs you in to the ACE. Enter **admin** as the user name and type the new password that you entered in Step 3 of the “Sessioning and Logging in to the ACE from the Supervisor Engine” section.
Step 3	**show telnet** **Example:** host1/Admin# show telnet	Displays the Telnet session.
Step 4	**copy running-config startup-config** **Example:** host1/Admin# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for Setting Up an ACE

The following example configuration shows how to set up an ACE for remote management:

```
host1/Admin# show running-config

Generating configuration....

login timeout 0

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol telnet any
  3 match protocol ssh any
  4 match protocol icmp any

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

interface vlan 1000
  description Management connectivity on VLAN 1000
  ip address 172.25.91.110 255.255.255.0
  service-policy input REMOTE_MGMT_ALLOW_POLICY
  no shutdown

ip route 0.0.0.0 0.0.0.0 172.25.91.1

username admin password 5 $1$JwBOOUEt$jihXQiAjF9igwDay1qAvK.  role Admin domain 
default-domain
username www password 5 $1$xmYMkFnt$n1YUgNOo76hAhg.JqtymF/  role Admin domain 
default-domain
```

Where to Go Next

In this chapter, you have set up your ACE so that you can access it remotely through a management interface. In the next chapter, you will create a user virtual context that you will use later for server load balancing.

CHAPTER 3

Configuring Virtualization

This chapter describes how to configure virtualization for the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

Information About Virtualization

After reading this chapter, you should have a basic understanding of ACE virtualization and be able to partition your ACE into multiple virtual devices or virtual contexts (VCs) for more efficient operation.

Virtualization allows you to create a virtual environment in which a single ACE is partitioned into multiple virtual devices, each functioning as an independent ACE that is configured and managed independently.

You set up virtualization by performing the following configuration steps:

- Configure resource allocation for a virtual context
- Create a virtual context
- Configure access to the virtual context

An example virtual environment will be used throughout this guide, with the user context VC_WEB, for the web traffic through the network. This user context will be associated with the custom resource class RC_WEB.

In this chapter, you will create a virtual context. In subsequent chapters, you will create a virtual server within the virtual context. The virtual server is associated with a server farm and real servers. The example setup is shown in Table 3-1.

Table 3-1 Example Virtual Context

Virtual Context	Virtual Server	Server Farm	Real Servers
VC_WEB	VS_WEB	SF_WEB	RS_WEB1
			RS_WEB2
			RS_WEB3
			RS_WEB4

Before you begin configuring your ACE for virtualization, you should become familiar with a few concepts: virtual context, Admin and user contexts, and resource classes.

With ACE virtualization, you can create a virtual environment, called a virtual context, in which a single ACE appears as multiple virtual devices, each configured and managed independently. A virtual context allows you to closely and efficiently manage system resources, ACE users, and the services that you provide to your customers.

By default, the ACE initially provides you an Admin context, with the ability to define up to five user contexts. With additional licenses, you can define up to a maximum of 251 contexts: one Admin context and 250 user contexts.

As the system administrator, you have full system administrator access to configure and manage the Admin context and all user contexts. Each context can also have its own administrator and log-in mechanism that provides access only to the specific context. When you log in to the ACE using the console or Telnet, you are authenticated in the Admin context.

Although virtualization allows you to create multiple contexts, in the physical world, you still have a single ACE with finite resources, such as the number of concurrent connections. To address this limitation, the ACE provides resource classes that allow you to manage each virtual context's access to physical ACE resources. A resource class is a definition of what portion of an ACE's overall resources will be assigned, at a minimum or maximum, to any given context. One resource class may be associated with one or more contexts.

The ACE is preconfigured with a default resource class for the Admin context. This default resource class is applied to all virtual contexts that you create. It allows a maximum of 100 percent access to all resources by all virtual contexts. When a resource is being used to its maximum limit, the ACE will deny additional requests for that resource from any other virtual contexts. To avoid oversubscribing resources and to help guarantee that resource availability is shared among multiple virtual contexts, you create custom resource classes and associate them with the virtual contexts that you define.

Licensing Requirements for Virtual Contexts

By default, your ACE provides an Admin context and five user contexts that allow you to use multiple contexts if you choose to configure them. To increase the number of user contexts up to a maximum of 250, you must obtain a separate license from Cisco.

Table 3-2 shows the licensing requirements for increasing the number of virtual contexts in your ACE.

Table 3-2 ACE Virtualization Licensing Options

Feature	License Model	Description
Virtualization	ACE-VIRT-020	20 virtual contexts.
	ACE-VIRT-050	50 virtual contexts.
	ACE-VIRT-100	100 virtual contexts.
	ACE-VIRT-250	250 virtual contexts.
	ACE-VIRT-UP1	Upgrades 20 to 50 contexts.
	ACE-VIRT-UP2	Upgrades 50 to 100 contexts.
	ACE-VIRT-UP3	Upgrades 100 to 250 contexts.

For details about licensing, see the *Administration Guide, Cisco ACE Application Control Engine*.

Configuring a Virtual Context

This section describes how to configure a virtual context and it contains the following topics:

- Task Flow for Configuring a Virtual Context
- Configuring a Resource Class
- Creating a Virtual Context
- Configuring Remote Management Access to the User Contexts
- Configuring the Client-Side VLAN Interface
- Configuring the Server-Side VLAN Interface
- Configuring a Default Route for the Virtual Context

Task Flow for Configuring a Virtual Context

Follow these steps to configure a virtual context:

Step 1 Configure a resource class.

Step 2 Create a virtual context.

Step 3 Assign a management VLAN interface to the context.

Step 4 Configure remote management access to the user context.

Step 5 Configure the client-side VLAN interface.

Step 6 Configure the server-side VLAN interface.

Configuring a Resource Class

Note By default, the Admin context is a member of the default resource class. To ensure that the Admin context resources are not depleted by other virtual contexts and that the context will be guaranteed a minimum amount of resources, we recommend that you create a separate resource class, allocate the resources that you estimate will be required by the Admin context, and make the Admin context the only member.

Procedure

	Command	Purpose

Step 1	**telnet** *ip_address*
	
	Example:
	Telnet 172.25.91.110
	
	`host1 login: admin`
	`Password:`
	`Cisco Application Control Software (ACSW)`
	`TAC support: http://www.cisco.com/tac`
	`Copyright (c) 2002-2009, Cisco Systems, Inc. All rights reserved.`
	`The copyrights to certain works contained herein are owned by other third parties and are used and distributed under license.`
	`Some parts of this software are covered under the GNU Public License. A copy of the license is available at http://www.gnu.org/licenses/gpl.html.`
	`switch/Admin#`
		Logs you in to the ACE as the system administrator from the console. At the prompt, enter **admin** and then the new password you entered in Step 3 in the "Sessioning and Logging in to the ACE from the Supervisor Engine" section in Chapter 2, Setting Up an ACE.

Step 2	**config**
	
	Example:
	`host1/Admin# config`
	`host1/Admin(config)#`
		Enters configuration mode.

Step 3	**resource-class** *name*
	
	Example:
	`host1/Admin(config)# resource-class RC_WEB`
	`host1/Admin(config-resource)#`
		Creates a resource class.

Step 4	**limit-resource all minimum** *number* **maximum unlimited | equal-to-min** **Example:** `host1/Admin(config-resource)# limit-resource all minimum 10 maximum equal-to-min`	Limits the resources of a context to 10 percent of the total resources available on the ACE.

	Command	Purpose

Step 5	**exit**
	
	Example:
	`host1/Admin(config-resource)# exit`
	`host1/Admin(config)#`
		Exits resource configuration mode.

Step 6	**do copy running-config startup-config** **Example:** `host1/Admin(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Creating a Virtual Context

Procedure

	Command	Purpose

Step 1	**context** *context_name*
	
	Example:
	`host1/Admin(config)# context VC_WEB`
	`host1/Admin(config-context)#`
		Creates a new context.

Step 2	**allocate-interface vlan** *vlan_id*
	
	Example:
	`host1/Admin(config-context)# allocate-interface vlan 60`
	`host1/Admin(config-context)# allocate-interface vlan 400`
	`host1/Admin(config-context)# allocate-interface vlan 500`
	`host1/Admin(config-context)# allocate-interface vlan 1000`
		Associates existing VLAN 400 and VLAN 500 with the context so that the context can receive traffic classified for it. VLAN 60 and VLAN 1000 will be used later when you configure redundancy.

Step 3	**member** *class_name* **Example:** `host1/Admin(config-context)# member RC_WEB`	Associates the context with the resource class that you created in the "Configuring a Resource Class" section.
Step 4	**exit** **Example:** `host1/Admin(config-context)# exit`	Exits context configuration mode.
Step 5	**do show running-config context** **Example:** `host1/Admin(config)# do show running-config context`	Displays the virtual context configuration.

	Command	Purpose
Step 6	**`do show running-config resource-class`** **Example:** `host1/Admin(config)# do show running-config resource-class`	Displays the resource class configuration.
Step 7	**`do copy running-config startup-config`** **Example:** `host1/Admin(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring Remote Management Access to the User Contexts

Before remote network access can occur on the user context through Telnet or SSH, you must create a traffic policy that identifies the network management traffic that can be received by the ACE.

Procedure

	Command	Purpose

Step 1	**`changeto`**
	
	`Example:`
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the VC_WEB user context.

Step 2	**`config`**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**`class-map type management match-any`** *`name`*
	
	Example:
	`host1/VC_WEB(config)# class-map type management match-any REMOTE_ACCESS`
	`host1/VC_WEB(config-cmap-mgmt)#`
		Creates a management-type class map named REMOTE_ACCESS that matches any traffic.

Step 4	**`description`** *`string`* **Example:** `host1/VC_WEB(config-cmap-mgmt)# description Remote access traffic match`	(Optional) Provides a description for the class map.

Step 5	**`match protocol`** *`protocol`* **`any`**
	
	Example:
	`host1/VC_WEB(config-cmap-mgmt)# match protocol ssh any`
	`host1/VC_WEB(config-cmap-mgmt)# match protocol telnet any`
	`host1/VC_WEB(config-cmap-mgmt)# match protocol icmp any`
		Configures the match protocol to permit traffic based on the SSH, Telnet, and ICMP protocols for any source address.

	Command	Purpose

Step 6	**exit**
	
	Example:
	`host1/VC_WEB(config-cmap-mgmt)# exit`
	`host1/VC_WEB(config)#`
		Exits class map management configuration mode.

Step 7	**policy-map type management first-match** *name*
	
	Example:
	`host1/VC_WEB(config)# policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY`
	`host1/VC_WEB(config-pmap-mgmt)#`
		Creates a REMOTE_MGMT_ALLOW_POLICY policy map for traffic destined to an ACE interface.

Step 8	**class** *name*
	
	Example:
	`host1/VC_WEB(config-pmap-mgmt)# class REMOTE_ACCESS`
	`host1/VC_WEB(config-pmap-mgmt-c)#`
		Applies the REMOTE_ACCESS class map to this policy.

Step 9	**permit** **Example:** `host1/VC_WEB(config-pmap-mgmt-c)# permit`	Allows the ACE to receive the configured class map management protocols.

Step 10	**exit**
	
	Example:
	`host1/VC_WEB(config-pmap-mgmt-c)# exit`
	`host1/VC_WEB(config-pmap-mgmt)# exit`
	`host1/VC_WEB(config)#`
		Exits policy map management class configuration mode.

Step 11	**service-policy input** *policy_name* **Example:** `host1/VC_WEB(config)# service-policy input REMOTE_MGMT_ALLOW_POLICY`	In configuration mode, applies the REMOTE_MGMT_ALLOW_POLICY policy map globally to all interfaces in the user context. You can also apply a remote management policy to a VLAN. In this topology, you could apply the management policy to either VLAN 400 (client-side VLAN) or VLAN 500 (server-side VLAN).

Step 12	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)#`
		Exits interface configuration mode.

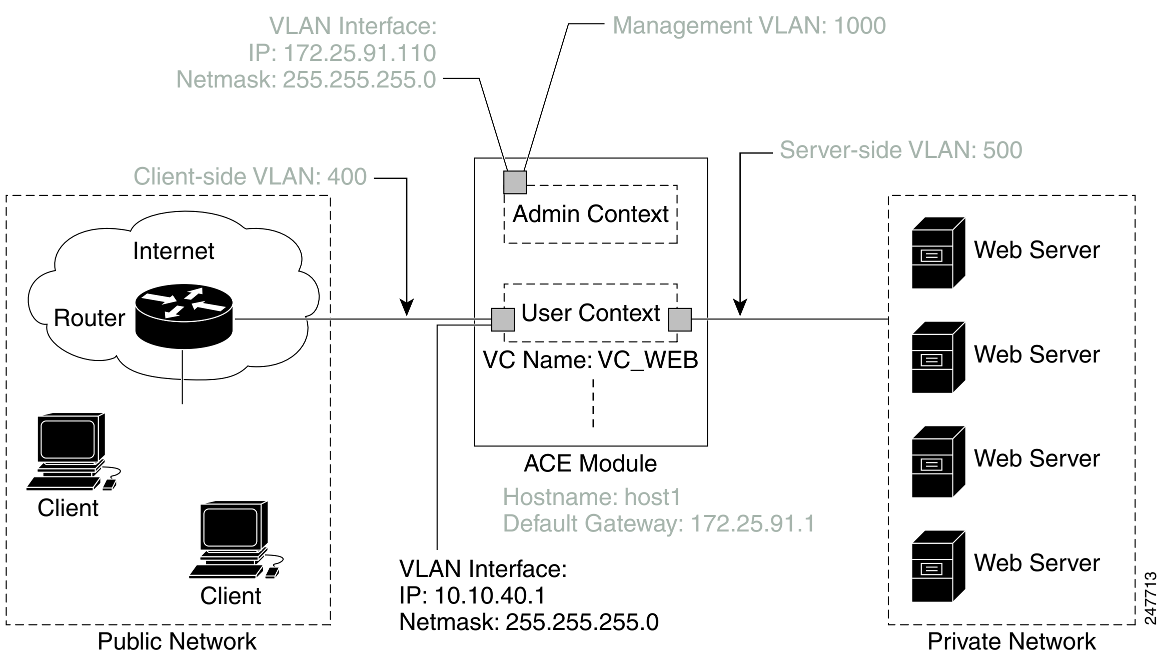
Step 13	**do show service-policy** *policy_name* **Example:** `host1/VC_WEB(config)# do show service-policy REMOTE_MGMT_ALLOW_POLICY`	Displays the REMOTE_MGMT_ALLOW_POLICY policy applied to all interfaces in the context.

	Command	Purpose
Step 14	**`do show running-config`** **Example:** `host1/VC_WEB(config)# do show running-config`	Displays the running configuration.
Step 15	**`do copy running-config startup-config`** **Example:** `host1/VC_WEB(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration in the VC_WEB context.

Configuring the Client-Side VLAN Interface

At this point, you can configure a client-side VLAN interface, the address to which the client traffic is sent, as shown in Figure 3-1.

Figure 3-1 Configuring the Client-Side VLAN Interface



Procedure

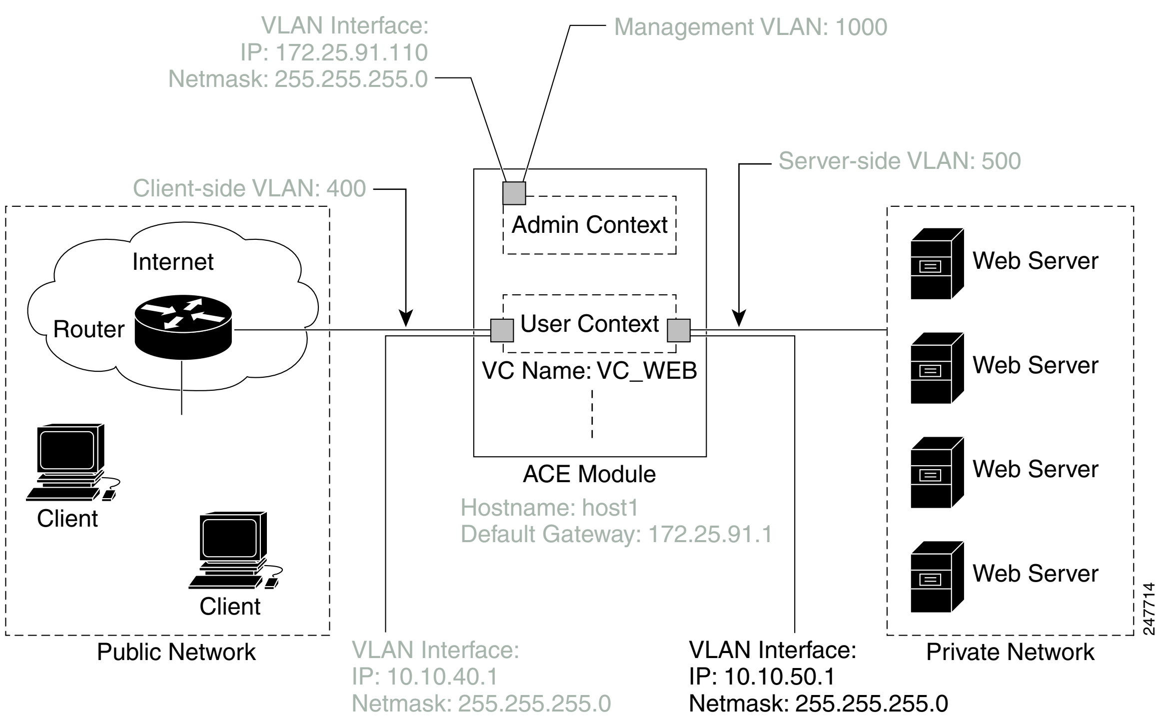
	Command	Purpose
Step 1	**interface vlan** *vlan_id* **Example:** host1/VC_WEB(config)# interface vlan 400 host1/VC_WEB(config -if)#	Accesses interface configuration mode in the VC_WEB context for client-side VLAN 400.
Step 2	**ip address** *ip_address netmask* **Example:** host1/VC_WEB(config-if)# ip address 10.10.40.1 255.255.255.0	Assigns an IP address of 10.10.40.1 and a subnet mask of 255.255.255.0 to the VLAN interface for management connectivity.
Step 3	**description** *string* **Example:** host1/VC_WEB(config-if)# description Client connectivity on VLAN 400	(Optional) Provide a description for the interface.
Step 4	**no shutdown** **Example:** host1/VC_WEB(config-if)# no shutdown	Enables the VLAN interface.
Step 5	**do show interface vlan** *vlan_id* **Example:** host1/VC_WEB(config-if)# do show interface vlan 400	Shows that VLAN 400 is active.
Step 6	**do show arp** **Example:** host1/VC_WEB(config-if)# do show arp	Displays the ARP table. **Note** The Address Resolution Protocol (ARP) allows the ACE to manage and learn the mapping of IP to Media Access Control (MAC) information to forward and transmit packets.
Step 7	**exit** **Example:** host1/VC_WEB(config-if)# exit host1/VC_WEB(config)#	Exits interface configuration mode.
Step 8	**do copy running-config startup-config** **Example:** host1/Admin(config)# do copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuring the Server-Side VLAN Interface

Next, you can configure a server-side VLAN interface, the address to which the server traffic is sent, and a NAT pool as shown in Figure 3-2.

Note Network Address Translation (NAT) is designed to simplify and conserve IP addresses. It allows private IP networks that use unregistered IP addresses to connect to the Internet. You configure a NAT pool for the ACE so that the ACE exposes only one address for the entire network to the outside world. This pool, which hides the entire internal network behind that address, offers both security and address conservation.

Figure 3-2 Configuring the Server-Side VLAN Interface



Procedure

	Command	Purpose

Step 1	**`interface vlan`** *`vlan_id`*
	
	`Example:`
	`host1/VC_WEB(config)# interface vlan 500`
	`host1/VC_WEB(config -if)#`
		Accesses interface configuration mode in the VC_WEB context for server-side VLAN 500.

Step 2	**`ip address`** *`ip_address netmask`* **`Example:`** `host1/VC_WEB(config-if)# ip address 10.10.50.1 255.255.255.0`	Assigns an IP address of 10.10.50.1 and a subnet mask of 255.255.255.0 to the VLAN interface for management connectivity.

	Command	Purpose
Step 3	**description** *string* **Example:** `host1/VC_WEB(config-if)# description Server connectivity on VLAN 500`	(Optional) Provides a description for the interface.
Step 4	**no shutdown** **Example:** `host1/VC_WEB(config-if)# no shutdown`	Enables the VLAN interface.
Step 5	**do show interface vlan** *vlan_id* **Example:** `host1/VC_WEB(config-if)# do show interface vlan 500`	Shows that VLAN 500 is active.
Step 6	**do show arp** **Example:** `host1/VC_WEB(config-if)# do show arp`	Displays the ARP table. **Note** The Address Resolution Protocol (ARP) allows the ACE to manage and learn the mapping of IP to Media Access Control (MAC) information to forward and transmit packets.

Step 7	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits interface configuration mode. Exits configuration mode.

Step 8	**copy running-config startup-config** **Example:** `host1/Admin# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring a Default Route for the Virtual Context

Because the ACE does not share configurations across contexts, you must configure a default route for each virtual context that you create. For details about creating a default route, see the "Configuring a Default Route" section. The default route used for this virtual context is 172.25.91.1. See the configuration example at the end of this chapter.

Configuration Examples for Configuring a Virtual Context

The following examples show how to configure a virtual context. The two examples are for the Admin context and the VC_WEB virtual context, respectively. The commands that you have configured in this chapter are shown in bold text.

Admin Context Configuration Example

The following example shows the running configuration of the Admin context with the commands that you have configured in this chapter in bold text.

```
host1/Admin# show running-config

Generating configuration....

login timeout 0

resource-class RC_WEB
  limit-resource all minimum 10.00 maximum equal-to-min

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol telnet any
  3 match protocol ssh any
  4 match protocol icmp any

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

interface vlan 1000
  description Management connectivity on VLAN 1000
  ip address 172.25.91.110 255.255.255.0
  service-policy input REMOTE_MGMT_ALLOW_POLICY
  no shutdown

ip route 0.0.0.0 0.0.0.0 172.25.91.1

context VC_WEB
  allocate-interface vlan 60
  allocate-interface vlan 400
  allocate-interface vlan 500
  allocate-interface vlan 1000
  member RC_WEB

username admin password 5 $1$JwBOOUEt$jihXQiAjF9igwDay1qAvK.  role Admin domain
default-domain
username www password 5 $1$xmYMkFnt$n1YUgNOo76hAhg.JqtymF/  role Admin domain
default-domain
```

VC_WEB Configuration Example

The following example shows the running configuration of the VC_WEB user context with the commands that you have configured in this chapter in bold text.

```
host1/Admin# changeto VC_WEB
VC_WEB/Admin# show running-config

Generating configuration....

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

ip route 0.0.0.0 0.0.0.0 172.25.91.1
```

Where to Go Next

In this chapter, you have partitioned your ACE into an Admin context and a user context (VC_WEB). Each of the virtual contexts is now associated with a resource class that is appropriate to its intended use. You have also configured a management VLAN interface, as well as the client and server VLAN interfaces in the user context.

In the next chapter, you will configure an access control list (ACL) to secure your network and to permit traffic to enter the ACE.

CHAPTER 4

Configuring Access Control Lists

This chapter describes how to configure security access control lists (ACLs) for the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

Information About ACLs

After reading this chapter, you should have a basic understanding of how to configure an ACL in an ACE to secure your network.

An ACL consists of a series of ACL entries, which are permit or deny entries with criteria for the source IP address, destination IP address, protocol, port, or protocol-specific parameters. Each entry permits or denies inbound or outbound network traffic to the parts of your network specified in the entry.

You can use ACLs with the ACE to permit or deny traffic to or from a specific IP address or an entire network. For example, you can permit all e-mail traffic on a circuit, but block Telnet traffic. You can also use ACLs to allow one client to access a part of the network while preventing other clients from doing so.

The order of the ACL entries is important. When the ACE decides whether to accept or refuse a connection, it tests the packet against each ACL entry in the order in which the entries are listed. After it finds a match, it stops checking entries.

For example, if you create an entry at the beginning of an ACL that explicitly permits all traffic, the ACE skips any other entries in the ACL. An implicit deny all entry exists at the end of every ACL, so you must include entries for every interface on which you want to permit connections. Otherwise, the ACE will deny all traffic on the interface.

Certain applications require special handling of the data portion of a packet as the packets pass through the ACE. The ACE verifies the protocol behavior and identifies unwanted or malicious traffic that attempts to pass through. Based on the specifications of the traffic policy, the ACE performs application protocol inspection to accept or reject the packet to ensure the secure use of applications and services.

For more information on how to configure an ACL to permit or deny specific traffic or resources, see the *Security Guide, Cisco ACE Application Control Engine*.

Guidelines and Restrictions

You must configure an ACL on each interface that you want to permit connections. Otherwise, the ACE will deny all traffic on the interface.

Configuring an ACL

Procedure

	Command	Purpose

Step 1	**`changeto`** *`context`*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the VC_WEB context.

Step 2	**`config`**
	
	Example:
	`host1/VC_WEB# Config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**`access-list INBOUND extended permit ip any any`** **Example:** `host1/VC_WEB(config)# access-list INBOUND extended permit ip any any`	Creates an ACL that permits all IP traffic to the ACE.
Step 4	**`interface vlan`** *`vlan_id`* **Example:** `host1/VC_WEB(config)# interface vlan 400`	Enters interface VLAN configuration mode for the client-side VLAN 400.

Step 5	**`access-group input`** *`acl_name`*
	
	Example:
	`host1/VC_WEB(config-if)# access-group input INBOUND`
	`host1/VC_WEB(config-if)# exit`
		Applies the ACL to the interface.

Step 6	**`exit`**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits interface configuration mode. Exits configuration mode.

Step 7	**`show running-config access-list`** **Example:** `host1/VC_WEB# show running-config access-list`	Displays the ACL configuration information.
Step 8	**`copy running-config startup-config`** **Example:** `host1/Admin# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for Configuring an ACL

The following example shows the running configuration of the VC_WEB user context with the commands that you have configured in this chapter in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
no shutdown

ip route 0.0.0.0 0.0.0.0 172.25.91.1
```

Where to Go Next

In this chapter, you have created an ACL entry to permit all traffic to the network. In the next chapter, you will create a user who is allowed to perform a subset of the ACE management functions on part of your network resources.

CHAPTER 5

Configuring Role-Based Access Control

This chapter describes how to configure role-based access control (RBAC) on the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

Information About Role-Based Access Control

After reading this chapter, you should have a basic understanding of how the ACE provides security administration by using role-based access control (RBAC) and how to configure a server maintenance user with permission to access a subset of your network.

One of the most challenging problems in managing large networks is the complexity of security administration. The ACE allows you to determine the commands and resources available to each user through RBAC by associating users with domains and roles.

A domain is a collection of physical and virtual network resources such as real servers and virtual servers.

User roles determine user privileges, such as the commands that the user can enter and the actions the user can perform in a particular context. The ACE provides a number of predefined roles; context administrators can create new roles.

The ACE provides the following predefined roles, which you cannot delete or modify:

- Admin—If created in the Admin context, has complete access to, and control over, all contexts, domains, roles, users, resources, and objects in the entire ACE. If created in a user context, gives a user complete access to and control over all policies, roles, domains, server farms, real servers, and other objects in that context.
- Network Admin—Has complete access to and control over the following features:
  - Interfaces
  - Routing
  - Connection parameters
  - Network Address Translation (NAT)

  - VIPs
  - Copy configurations
  - **changeto** command
- Network-Monitor—Has access to all **show** commands and to the **changeto** command. If you do not explicitly assign a role to a user with the **username** command, this is the default role.
- Security-Admin—Has complete access to and control over the following security-related features within a context:
  - ACLs
  - Application inspection
  - Connection parameters
  - Interfaces
  - Authentication, authorization, and accounting (AAA)
  - NAT
  - Copy configurations
  - **changeto** command
- Server-Appln-Maintenance—Has complete access to and control over the following features:
  - Real servers
  - Server farms
  - Load balancing
  - Copy configurations
  - **changeto** command
- Server-Maintenance—Can perform real server maintenance, monitoring, and debugging for the following features:
  - Real servers—Modify permission
  - Server farms—Debug permission
  - VIPs—Debug permission
  - Probes—Debug permission
  - Load balancing—Debug permission
  - **changeto** command—Create permission
- SLB-Admin—Has complete access to and control over the following ACE features within a context:
  - Real servers
  - Server farms
  - VIPs
  - Probes
  - Load balancing (Layer 3/4 and Layer 7)
  - NAT
  - Interfaces
  - Copy configurations
  - **changeto** command

- SSL-Admin—Can administer all SSL features:
  - SSL—Create permission
  - PKI—Create permission
  - Interfaces—Modify permission
  - Copy configurations—Create permission
  - **changeto** command—Create permission

This chapter describes how to create a domain and a user, and how to associate the user with a predefined role and the new domain. For more information on advanced virtualization configuration, such as restricting user access, predefined roles and how to define a custom role, and creating a domain, see the *Virtualization Guide, Cisco ACE Application Control Engine*.

Configuring RBAC

Procedure

	Command	Purpose

Step 1	`changeto` *`context`*
	
	`Example:`
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the VC_WEB context.

Step 2	**`config`**
	
	`Example:`
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**`domain`** *`name`*
	
	`Example:`
	`host1/VC_WEB(config)# domain DOMAIN1`
	`host1/VC_WEB(config-domain)#`
		Creates a domain for the context.

Step 4	**`add-object all`** **`Example:`** `host1/VC_WEB(config-domain)# add-object all`	Allocates all configuration objects in the VC_WEB context to the domain.

Step 5	**`exit`**
	
	`Example:`
	`host1/VC_WEB(config-domain)# exit`
	`host1/VC_WEB(config)#`
		Exits domain configuration mode.

	Command	Purpose

Step 6	**username** *user* **password 5** *password* **role** *name1* **domain** *name2*
	
	Example:
	`host1/VC_WEB(config)# username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1 role Server-Maintenance domain DOMAIN1`
	`host1/VC_WEB(config)# exit`
		Configures new user USER1, and assigns the predefined role SLB-Admin and the domain DOMAIN1 to USER1
		
		The **5** parameter for the **password** keyword requires that you enter an MD5 hash-encrypted password. You can obtain an MD5 hash password by first entering the **username** command with the **0** parameter and a clear-text password (for example, MYPASSWORD). Next, enter the **show running-config** command and copy the user's encrypted password from the running-configuration file. Enter the username command again using the **5** parameter and the encrypted password.

Step 7	**exit**
	
	Example:
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits configuration mode.

Step 8	**show running-config role**
	show running config domain
	
	Examples:
	`host1/VC_WEB# show running-config role`
	`host1/VC_WEB# show running-config domain`
		Displays the user and domain configurations.

Step 9	**copy running-config startup-config** **Example:** `host1/VC_WEB# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for Configuring RBAC

The following example shows how to configure RBAC. The commands that you have configured in this chapter are shown in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

service-policy input REMOTE_MGMT_ALLOW_POLICY
```

```
interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

ip route 0.0.0.0 0.0.0.0 172.25.91.1

domain DOMAIN1
add-object all
username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role Server-Maintenance
domain DOMAIN1
```

Where to Go Next

In this chapter, you have created a user to perform a limited number of functions on a subset of your network. In the next chapter, you will create a virtual server for server load balancing.

CHAPTER 6

Configuring Server Load Balancing

This chapter describes how to configure server load balancing (SLB) on the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

- Information About Server Load Balancing
- Configuring Server Load Balancing
- Configuration Example for Configuring Server Load Balancing
- Where to Go Next

Information About Server Load Balancing

After reading this chapter, you should have an understanding of the basic SLB capabilities provided by the ACE. You should also be able to configure a virtual server for Layer 7 load-balancing purposes.

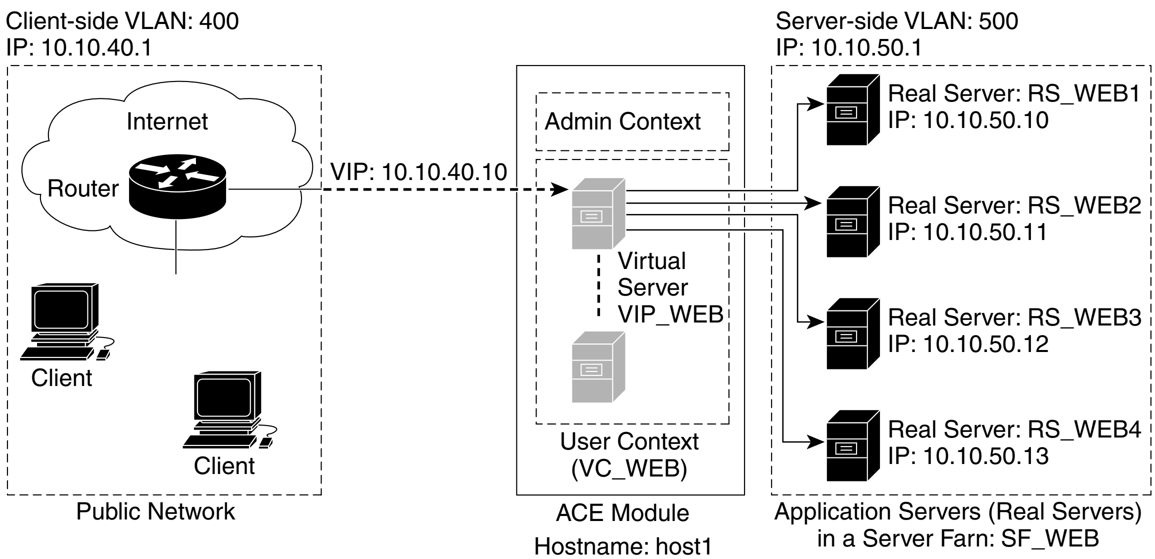
When there is a client request for web services, a load-balancing device decides to which server it should send the request. For example, a client request may consist of an HTTP GET for a web page or an FTP GET to download a file. The ACE, as a server load balancer, selects a server that can successfully fulfill the client request in the shortest amount of time without overloading either the server or the server farm as a whole.

The ACE uses a virtual server to intercept web traffic to a website. A virtual server allows multiple real servers to appear as one for load-balancing purposes. A virtual server, also called a Virtual IP (VIP), is defined by its IP address, the protocol used (for example, UDP or TCP), and the port address.

Multiple servers grouped together in server farms are assigned to each virtual server and the ACE carries out load balancing across them. Real servers are dedicated servers that provide services to clients—for example, delivery of HTTP or XML content. Real servers within a server farm usually contain the same content and typically reside in the same physical location in a data center.

This chapter describes how to configure a virtual serverusing the network example in Figure 6-1.

Figure 6-1 Example Server Load-Balancing Setup



The configuration of the example setup is as follows:

- A virtual server VS_WEB is created with a virtual IP address 10.10.40.10 to forward the client traffic from VLAN 400 to the application servers in VLAN 500.
- There are four real servers grouped into the server farm SF_WEB.
- The virtual server uses a round-robin predictor to forward the client requests to one of the real servers in the server farm.

Configuring Server Load Balancing

This section describes how to configure server load balancing. It contains the following topics:

- Task Flow for Configuring Server Load Balancing
- Configuring Real Servers
- Creating a Server Farm
- Creating a Virtual Server Traffic Policy

Task Flow for Configuring Server Load Balancing

Follow these steps to configure server load balancing:

Step 1 Configure real servers.

Step 2 Create a server farm.

Step 3 Create a virtual server traffic policy.

Configuring Real Servers

Procedure

	Command	Purpose

Step 1	**changeto** *context*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**rserver** *name*
	
	Example:
	`host1/VC_WEB(config)# rserver RS_WEB1`
	`host1/VC_WEB(config-rserver-host)#`
		Creates a real server named RS_WEB1 as type host (the default).

Step 4	**description** *string*
	
	Example:
	`host1/VC_WEB(config-rserver-host)#`
	`description content server web-one`
		Enters a description of the real server.

Step 5	**ip address** *address*
	
	Example:
	`host1/VC_WEB(config-rserver-host)# ip`
	`address 10.10.50.10`
		Assigns the real server with an IP address of 10.10.50.10.

Step 6	**inservice**
	
	Example:
	`host1/VC_WEB(config-rserver-host)#`
	`inservice`
		Places the real server in service.

Step 7	**exit**
	
	Example:
	`host1/VC_WEB(config-rserver-host)# exit`
	`host1/VC_WEB(config)#`
		Exits real server host configuration mode.

	Command	Purpose
Step 8	**rserver** *name* **Example:** host1/VC_WEB(config)# rserver RS_WEB2 host1/VC_WEB(config-rserver-host)#	Add three more real servers by repeating Steps 3 through 7, using the following real server names, descriptions, and IP addresses. For RS_WEB2, enter: • Name: RS_WEB2 • Description: content server web-two • IP Address: 10.10.50.11 For RS_WEB3, enter: • Name: RS_WEB3 • Description: content server web-three • IP Address: 10.10.50.12 For RS_WEB4, enter: • Name: RS_WEB4 • Description: content server web-four IP Address: 10.10.50.13
	description *string* **Example:** host1/VC_WEB(config-rserver-host)# description content server web-two	
	ip address *address* **Example:** host1/VC_WEB(config-rserver-host)# ip address 10.10.50.11	
	inservice **Example:** host1/VC_WEB(config-rserver-host)# inservice	
	exit **Example:** host1/VC_WEB(config-rserver-host)# exit host1/VC_WEB(config)#	
Step 9	**do show running-config rserver** **Example:** host1/VC_WEB(config)# do show running-config rserver	Displays the configuration of the real servers.
Step 10	**do copy running-config startup-config** **Example:** host1/Admin(config)# do copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Creating a Server Farm

After you create and configure the real servers, you can create a server farm and associate the real servers with it.

Procedure

	Command	Purpose

Step 1	**serverfarm** *name*
	
	Example:
	`host1/VC_WEB(config)# serverfarm SF_WEB`
	`host1/VC_WEB(config-sfarm-host)#`
		Creates a server farm of type host (the default) named SF_WEB.

Step 2	**rserver** *name* [*port*]
	
	Example:
	`host1/VC_WEB(config-sfarm-host)#`
	`rserver RS_WEB1 80`
	`host1/VC_WEB(config-sfarm-host-rs)#`
		Associates real server RS_WEB1 with the server farm through port 80. Specifying a port number is optional. If you do not specify a port number, the ACE does not perform PAT and the destination port that was used from the client to the VIP will also be used from the VIP to the real server. When you specify a port number, it is the only destination port that the ACE uses from the VIP to the real server.

Step 3	**inservice**
	
	Example:
	`host1/VC_WEB(config-sfarm-host-rs)#`
	`inservice`
		Places the real server in service within the server farm.
		
		Note Before you can start sending connections to a real server in a server farm, you must place it in service. Otherwise, the ACE considers it out of service and the server farm cannot receive or respond to client requests.

Step 4	**exit**
	
	Example:
	`host1/VC_WEB(config-sfarm-host-rs)#`
	`exit`
	`host1/VC_WEB(config-sfarm-host)#`
		Exits server farm host real server configuration mode.

	Command	Purpose

Step 5	**`rserver`** *`name`* `[`*`port`*`]`
	`inservice`
	
	`Example:`
	`host1/VC_WEB(config-sfarm-host)# rserver RS_WEB2 80`
	`host1/VC_WEB(config-sfarm-host-rs)# inservice`
	`host1/VC_WEB(config-sfarm-host-rs)# exit`
	`host1/VC_WEB(config-sfarm-host)# rserver RS_WEB3 80`
	`host1/VC_WEB(config-sfarm-host-rs)# inservice`
	`host1/VC_WEB(config-sfarm-host-rs)# exit`
	`host1/VC_WEB(config-sfarm-host)# rserver RS_WEB4 80`
	`host1/VC_WEB(config-sfarm-host-rs)# inservice`
	`host1/VC_WEB(config-sfarm-host-rs)# exit`
		Similarly, associates the RS_WEB2, RS_WEB3, and RS_WEB4 real servers with the SF_WEB server farm and places the real server in service. Repeat steps 2 through 4 to configure the remaining real servers in the server farm.

Step 6	**`exit`**
	
	`Example:`
	`host1/VC_WEB(config-sfarm-host)# exit`
	`host1/VC_WEB(config)#`
		Exits server farm host configuration mode.

Step 7	**`do show rserver`** *`name`*
	
	`Example:`
	`host1/VC_WEB(config)# do show rserver RS_WEB1`
	`host1/VC_WEB(config)# do show rserver RS_WEB2`
	`host1/VC_WEB(config)# do show rserver RS_WEB3`
	`host1/VC_WEB(config)# do show rserver RS_WEB4`
		Displays the information for the real servers.
		
		Note The real server status is shown as ARP_FAILED because network connectivity has not been established yet.

Step 8	**`do copy running-config startup-config`** **`Example:`** `host1/VC_WEB(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Creating a Virtual Server Traffic Policy

Procedure

	Command	Purpose

Step 1	**`policy-map type loadbalance first-match`** *`name`*
	
	`Example:`
	`host1/VC_WEB(config)# policy-map type loadbalance first-match PM_LB`
	`host1/VC_WEB(config-pmap-lb)#`
		Creates a Layer 7 server load-balancing policy map named PM_LB to match the class maps in the order in which they occur for load balancing.
		
		Note The ACE uses a class map to specify a series of flow match criteria (traffic classifications). The ACE uses a policy map to define a series of actions (functions) that you want applied to a set of classified inbound traffic.

Step 2	**`class class-default`**
	
	`Example:`
	`host1/VC_WEB(config-pmap-lb)# class class-default`
	`host1/VC_WEB(config-pmap-lb-c)#`
		For a simple load-balancing policy, assigns the ACE default class map which contains an implicit match any statement to match any traffic classification.

Step 3	**`serverfarm`** *`name`* **`Example:`** `host1/VC_WEB(config-pmap-lb-c)# serverfarm SF_WEB`	Adds the server farm SF_WEB to the Layer 7 server load-balancing policy map and exits configuration mode.

Step 4	**`exit`**
	
	`Example:`
	`host1/VC_WEB(config-pmap-c)# exit`
	`host1/VC_WEB(config-pmap)# exit`
	`host1/VC_WEB(config)#`
		Exits policy map class configuration mode. Exits policy map configuration mode.

Step 5	**`class-map`** `{`**`match-all`**`|` **`match-any`** `|` **`type`**`}` *`name`*
	
	`Example:`
	`host1/VC_WEB(config)# class-map VS_WEB`
	`host1/VC_WEB(config-cmap)#`
		Creates a Layer 3 and Layer 4 load-balancing class map VS_WEB. The default is **match-all**.

Step 6	**`match virtual-address`** *`address netmask`* **`tcp eq`** *`port`* **`Example:`** `host1/VC_WEB(config-cmap)# match virtual-address 10.10.40.10 255.255.255.255 tcp eq 80`	Defines a match statement for the IP address 10.10.40.10, the TCP IP protocol, and port 80.

Step 7	**`exit`**
	
	`Example:`
	`host1/VC_WEB(config-cmap)# exit`
	`host1/VC_WEB(config)#`
		Exits class map configuration mode.

	Command	Purpose

Step 8	**policy-map multi-match** *name*
	
	Example:
	`host1/VC_WEB(config)# policy-map multi-match PM_MULTI_MATCH`
	`host1/VC_WEB(config-pmap)#`
		Creates a Layer 3 and Layer 4 multi-match policy map to direct classified incoming requests to the load-balancing policy map.

Step 9	**class** *name*
	
	Example:
	`host1/VC_WEB(config-pmap)# class VS_WEB`
	`host1/VC_WEB(config-pmap-c)#`
		Associates the VS_WEB Layer 3 and Layer 4 class map that you created in Step 4 with the PM_MULTI_MATCH policy map.

Step 10	**loadbalance policy** *name*
	
	Example:
	`host1/VC_WEB(config-pmap-c)# loadbalance policy PM_LB`
	`host1/VC_WEB(config-pmap-lb-c)#`
		Associates the PM_LB Layer 7 load-balancing policy map with the PM_MULTI_MATCH Layer 3 and Layer 4 policy map.

Step 11	**loadbalance vip inservice** **Example:** `host1/VC_WEB(config-pmap-lb-c)# loadbalance vip inservice`	Enables a VIP for load-balancing operations.

Step 12	**exit**
	
	Example:
	`host1/VC_WEB(config-pmap-c)# exit`
	`host1/VC_WEB(config-pmap)# exit`
	`host1/VC_WEB(config)#`
		Exits policy map class configuration mode. Exits policy map configuration mode.

Step 13	**interface vlan** *vlan_id*
	
	Example:
	`host1/VC_WEB(config)# interface vlan 400`
	`host1/VC_WEB(config-if)#`
		Accesses the interface to which you want to apply the multi-match policy map.

Step 14	**service-policy input** *policy_name* **Example:** `host1/VC_WEB(config-if)# service-policy input PM_MULTI_MATCH`	Applies the PM_MULTI_MATCH Layer 3 and Layer 4 policy map.

Step 15	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits interface configuration mode. Exits configuration mode.

Step 16	**show service-policy** *policy_name* **Example:** `host1/VC_WEB# show service-policy PM_MULTI_MATCH`	Displays the service policy state for the PM_MULTI_MATCH policy map.
Step 17	**copy running-config startup-config** **Example:** `host1/VC_WEB# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for Configuring Server Load Balancing

The following example shows how to configure server load balancing. The commands that you have configured in this chapter appear in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

rserver host RS_WEB1
  description content server web-one
  ip address 10.10.50.10
  inservice
rserver host RS_WEB2
  description content server web-two
  ip address 10.10.50.11
  inservice
rserver host RS_WEB3
  description content server web-three
  ip address 10.10.50.12
  inservice
rserver host RS_WEB4
  description content server web-four
  ip address 10.10.50.13
  inservice

serverfarm host SF_WEB
  rserver RS_WEB1 80
    inservice
  rserver RS_WEB2 80
    inservice
  rserver RS_WEB3 80
    inservice
  rserver RS_WEB4 80
    inservice

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any
class-map match-all VS_WEB
  2 match virtual-address 10.10.40.10 tcp eq www

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit
policy-map type loadbalance first-match PM_LB
  class class-default
    serverfarm SF_WEB
policy-map multi-match PM_MULTI_MATCH
  class VS_WEB
    loadbalance vip inservice
    loadbalance policy PM_LB

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  service-policy input PM_MULTI_MATCH
  no shutdown
```

```
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

ip route 0.0.0.0 0.0.0.0 172.25.91.1

domain DOMAIN1
add-object all

username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role SLB-Admin domain
DOMAIN1
```

Where to Go Next

In this chapter, you have configured real servers, a server farm, and a a virtual server for load-balancing HTTP traffic. In the next chapter, you will configure a load-balancing predictor to forward client requests to the appropriate real servers.

CHAPTER 7

Configuring a Load-Balancing Predictor

This chapter describes how to configure a load-balancing predictor (method) on the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

Information About Load-Balancing Predictors

After reading this chapter, you should have a basic understanding of how the ACE selects a real server for a client request using a predictor and how to configure the round-robin and the least-connections predictors.

When there is a client request for web services, the ACE selects a server that can successfully fulfill the client request in the shortest amount of time without overloading either the individual server or the server farm.

The ACE makes load-balancing choices using a predictor. When you configure a predictor, you define the series of checks and calculations that the ACE will perform to determine which real server can best service a client request.

For each server farm, you can configure one of several predictor types to allow the ACE to select an appropriate server. Two common predictor types include the following:

- Round-robin—Selects a server from the list of real servers based on the weighted server capacity. A weight can be assigned to each real server based on its connection capacity in relation to the other servers in a server farm. Servers with higher weight values receive a proportionally higher number of connections than servers with lower weight values. For example, a server with a weight of 5 would receive five connections for every one connection received by a server with a weight of 1. Also known as weighted round-robin, this type is the default predictor.
- Least connections— Selects the server with the fewest number of connections based on the server weight. This predictor is useful for processing light user requests (for example, browsing simple static web pages). Use the optional slow-start mechanism to avoid sending a high rate of new connections to servers that you have recently put into service. When a new real server enters slow-start mode, the ACE calculates and assigns an artificially high metric weight value to the new server and sends a small number of connections to the new server initially. The remaining

connections go to the existing servers based on their weight and current connections. When the slow-start timer expires or the real server weight reaches zero, the ACE takes the server out of slow-start mode and assigns connections normally.

For a complete list of predictor types that the ACE supports and how to configure them, see the *Server Load-Balancing Guide, Cisco ACE Application Control Engine*.

This chapter describes how to configure a round-robin predictor for the server farm that was created in Chapter 6, Configuring Server Load Balancing as shown in Figure 6-1.

Configuring the Round-Robin Predictor

Procedure

	Command	Purpose

Step 1	**changeto** *context*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**serverfarm** *name*
	
	Example:
	`host1/VC_WEB(config)# serverfarm SF_WEB`
	`host1/VC_WEB(config-sfarm-host)#`
		Enters server farm host configuration mode for SF_WEB.

Step 4	**predictor roundrobin**
	
	Example:
	`host1/VC_WEB(config-sfarm-host)#`
	`predictor roundrobin`
		Configures the round-robin predictor. For weighted round-robin, assign a weight to the real server.

Step 5	**exit**
	
	Example:
	`host1/VC_WEB(config-sfarm-host)# exit`
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits server farm host configuration mode. Exits configuration mode.

Step 6	**show running-config serverfarm**
	
	Example:
	`host1/VC_WEB# show running-config`
	`serverfarm`
		Display the predictor configuration information.

Step 7	**copy running-config startup-config**
	
	Example:
	`host1/VC_WEB# copy running-config`
	`startup-config`
		(Optional) Copies the running configuration to the startup configuration.

Configuration Example for the Round-Robin Predictor

The following example shows how to configure the round-robin predictor. The commands that you have configured in this chapter appear in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

rserver host RS_WEB1
  description content server web-one
  ip address 10.10.50.10
  inservice
rserver host RS_WEB2
  description content server web-two
  ip address 10.10.50.11
  inservice
rserver host RS_WEB3
  description content server web-three
  ip address 10.10.50.12
  inservice
rserver host RS_WEB4
  description content server web-four
  ip address 10.10.50.13
  inservice

serverfarm host SF_WEB
  predictor roundrobin
  rserver RS_WEB1 80
    inservice
  rserver RS_WEB2 80
    inservice
  rserver RS_WEB3 80
    inservice
  rserver RS_WEB4 80
    inservice

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any
class-map match-all VS_WEB
  2 match virtual-address 10.10.40.10 tcp eq www

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit
policy-map type loadbalance first-match PM_LB
  class class-default
    serverfarm SF_WEB
policy-map multi-match PM_MULTI_MATCH
  class VS_WEB
    loadbalance vip inservice
    loadbalance policy PM_LB

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
```

```
  service-policy input PM_MULTI_MATCH
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.2 255.255.255.0
  no shutdown

domain DOMAIN1
add-object all

ip route 0.0.0.0 0.0.0.0 172.25.91.1
username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role SLB-Admin domain
DOMAIN1
```

Where to Go Next

In this chapter, you have configured a round-robin predictor for your server load balancing. In the next chapter, you will configure server persistence by using the stickiness feature.

CHAPTER 8

Configuring Server Persistence Using Stickiness

This chapter describes how to configure server persistence using stickiness on the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

Information About Configuring Stickiness

After reading this chapter, you should have a basic understanding of how the ACE provides server persistence using stickiness, and how to configure HTTP cookie stickiness.

When customers visit an e-commerce site, they usually start by browsing the site. Depending on the application, the site may require that the client remain connected (stuck) to one server as soon as the initial connection is established, or the application may require this action only when the client starts to create a transaction, such as building a shopping cart.

For example, after the client adds items to a shopping cart, it is important that all subsequent client requests are directed to the same real server so that all the items are contained in one shopping cart on one server. An instance of a customer's shopping cart is typically local to a particular server rather than duplicated across multiple servers.

E-commerce applications are not the only types of applications that require a sequence of client requests to be directed to the same real server. Any web applications that maintain client information may require this behavior, such as banking and online trading applications, or FTP and HTTP file transfers.

You can configure the ACE so that the same client can maintain multiple, simultaneous, or subsequent TCP or IP connections with the same real server for the duration of a session. This session persistence capability of the ACE is called stickiness. A session is defined as a series of transactions between a client and a server over some finite period of time (from several minutes to several hours).

Depending on the configured server load-balancing policy, the ACE sticks a client to an appropriate server after the ACE determines which load-balancing method to use. If the ACE determines that a client is already stuck to a particular server, then the ACE sends that client request to that server, regardless of the load-balancing criteria. If the ACE determines that the client is not stuck to a particular server, it applies the normal load-balancing rules to the request.

To determine how a particular client is stuck to a specific web server and how an application distinguishes each client or a group of clients, the ACE supports the following sticky methods:

- Source and/or destination IP address—For stickiness, you can use the source IP address, the destination IP address, or both to uniquely identify individual clients and their requests based on their IP net masks. However, if an enterprise or service provider uses a mega-proxy (a free, anonymous web proxy service that can represent hundreds or thousands of different clients with a single source IP address) to establish client connections to the Internet, the source IP address is not a reliable indicator of the true source of the request. In this case, you can use another sticky method to ensure session persistence.
- Cookie—Client cookies uniquely identify clients to the ACE and to the servers that provide content. A cookie is a small data structure within the HTTP header that a server uses to deliver data to a web client, with the request that the client store the information. The cookie may be inserted into a response packet from the server or the ACE can insert the cookie. This information may include items that users have added to their shopping carts or travel dates that they have chosen. When the ACE examines a request for content and determines that the content is sticky, it examines any cookie or URL present in the content request. The ACE uses the information in the cookie or URL to direct the content request to the appropriate server.
- Hypertext Transfer Protocol (HTTP) header—You can specify a header offset to provide stickiness based on a unique portion of the HTTP header.
- Layer 4 Payload Stickiness—Layer 4 payload stickiness allows you to stick a client to a server based on the data in Layer 4 frames. You can specify a beginning pattern and ending pattern, the number of bytes to parse, and an offset that specifies how many bytes to ignore from the beginning of the data.
- HTTP Content Stickiness—HTTP content stickiness allows you to stick a client to a server based on the content of an HTTP packet. You can specify a beginning pattern and ending pattern, the number of bytes to parse, and an offset that specifies how many bytes to ignore from the beginning of the data.
- RADIUS Attribute Stickiness—The ACE supports stickiness based on RADIUS attributes. The following attributes are supported for RADIUS sticky groups:
  - Framed IP
  - Framed IP and calling station ID
  - Framed IP and username
- RTSP Session Header Stickiness—The ACE supports stickiness based on the RTSP session header field. With RTSP header stickiness, you can specify a header offset to provide stickiness based on a unique portion of the RTSP header.
- SIP Call-ID Header Stickiness—The ACE supports stickiness based on the SIP Call-ID header field. SIP header stickiness requires the entire SIP header, so you cannot specify an offset.
- SSL Session-ID Stickiness—This feature allows the ACE to stick the same client to the same SSL server based on the SSL Session ID. This feature supports SSLv3 only. Because the SSL Session ID is unique across multiple connections from the same client, you can use this feature to stick clients to a particular SSL server when the ACE is configured to load balance SSL traffic, but not terminate it. To use this feature, you must configure a generic protocol-parsing policy for sticky learning. The ACE learns the SSL Session ID from the SSL server or other SSL-termination device.

  Because an SSL server can reuse the same SSL Session ID for new connections from a known client, the SSL handshake time is reduced. This reduction in the handshake time translates directly into lower computational requirements for the server and reduced CPU utilization, and, therefore, increased SSL transactions per second (TPS).

The e-commerce application often dictates which of these methods is appropriate for a particular e-commerce application.

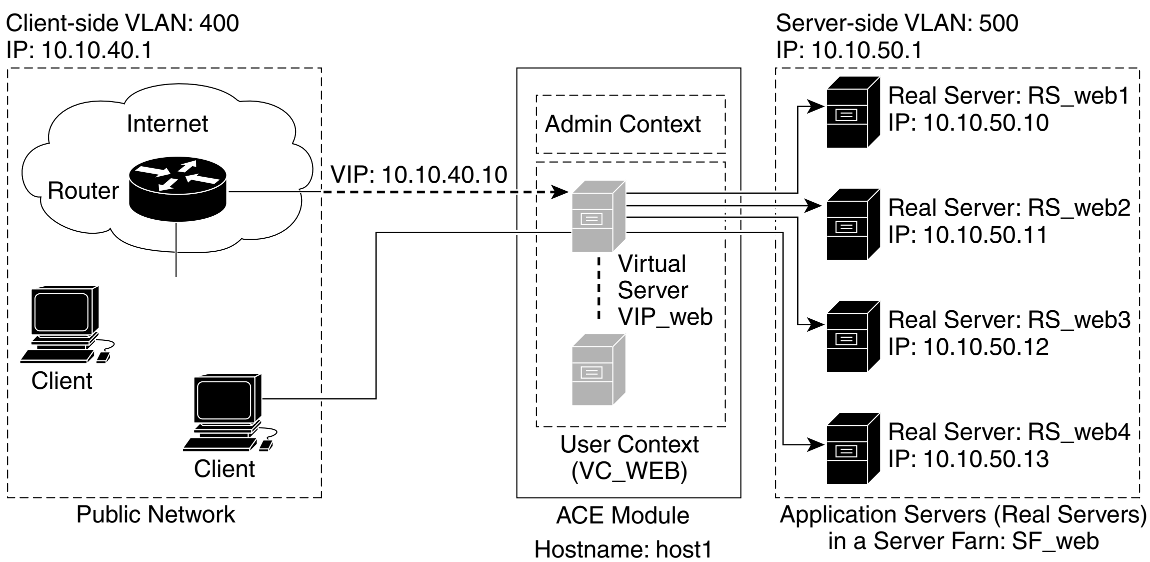
The ACE uses sticky groups for stickiness attributes. These attributes include the sticky method, timeout, replication, and attributes related to a particular sticky method.

To track sticky connections, the ACE uses a sticky table with information about sticky groups, sticky methods, sticky connections, and real servers. The ACE uses a configurable timeout mechanism to age out sticky table entries. When an entry times out, it becomes eligible for reuse. High connection rates may cause the premature aging out of sticky entries. In this case, the ACE reuses the entries that are closest to expiration first.

Entries in the sticky table can be either dynamic (generated by the ACE as needed) or static (configured). When you create a static sticky entry, the ACE places the entry in the sticky table immediately, and it remains in the sticky database until you remove it from the configuration.

Figure 8-1 illustrates that, in a server load-balancing environment, requests from a client are stuck to real server RS_WEB4 in a session.

Figure 8-1 Client Requests Stuck to a Server



This chapter describes how to configure stickiness using the HTTP cookie sticky method. For information on how to configure stickiness using the IP address and HTTP header methods, see the *Server Load-Balancing Guide, Cisco ACE Application Control Engine.*

Configuring HTTP Cookie Stickiness

Procedure

	Command	Purpose

Step 1	**changeto** *context*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**sticky http-cookie** *cookie_name group_name*
	
	Example:
	`host1/VC_WEB(config)# sticky http-cookie COOKIE1 STICKYGROUP1`
	`host1/VC_WEB(config-sticky-cookie)#`
		Creates an HTTP-cookie-type sticky group and enters cookie configuration mode.

Step 4	**cookie insert** [**browser-expire**] **Example:** `host1/VC_WEB(config-sticky-cookie)# cookie insert browser-expire`	Instructs the ACE to insert a cookie into the response packet from the server. The browser-expire option allows the browser to expire the cookie.
Step 5	**timeout** *number* **Example:** `host1/VC_WEB(config-sticky-cookie)# timeout 1440`	Configures a timeout for HTTP cookie stickiness.
Step 6	**serverfarm** *name* **Example:** `host1/VC_WEB(config-sticky-cookie)# serverfarm SF_WEB`	Associates a server farm with the sticky group and exits configuration mode.

Step 7	**exit**
	
	Example:
	`host1/VC_WEB(config-sticky-cookie)# exit`
	`host1/VC_WEB(config)#`
		Exits sticky cookie configuration mode.

Step 8	**policy-map type loadbalance first-match** *policy_name*
	class class-default
	
	Example:
	`host1/VC_WEB(config)# policy-map type loadbalance first-match PM_LB`
	`host1/VC_WEB(config-pmap-lb)# class class-default`
	`host1/VC_WEB(config-pmap-lb-c)#`
		Enters policy map load-balancing class configuration mode for the Layer 7 load-balancing policy.

	Command	Purpose
Step 9	**sticky-serverfarm** *group_name* **Example:** host1/VC_WEB(config-pmap-lb-c)# sticky-serverfarm STICKYGROUP1	Associates the sticky group with the Layer 7 load-balancing policy.
Step 10	**Ctrl-Z** **Example:** host1/VC_WEB(config-pmap-lb-c)# Ctrl-Z host1/VC_WEB#	Returns to Exec mode from any configuration mode.
Step 11	**show running-config sticky** **Example:** host1/VC_WEB# show running-config sticky	Displays the HTTP cookie configuration.
Step 12	**copy running-config startup-config** **Example:** host1/Admin# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for HTTP Cookie Stickiness

The following example shows how to configure HTTP cookie stickiness. The commands that you have configured in this chapter appear in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

rserver host RS_WEB1
  description content server web-one
  ip address 10.10.50.10
  inservice
rserver host RS_WEB2
  description content server web-two
  ip address 10.10.50.11
  inservice
rserver host RS_WEB3
  description content server web-three
  ip address 10.10.50.12
  inservice
rserver host RS_WEB4
  description content server web-four
  ip address 10.10.50.13
  inservice

serverfarm host SF_WEB
  predictor hash header Accept
  rserver RS_WEB1 80
    inservice
  rserver RS_WEB2 80
    inservice
  rserver RS_WEB3 80
    inservice
  rserver RS_WEB4 80
    inservice
```

```
sticky http-cookie Cookie1 StickyGroup1
  cookie insert browser-expire
  timeout 3600
  serverfarm SF_WEB

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any
class-map match-all VS_WEB
  2 match virtual-address 10.10.40.10 tcp eq www

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit
policy-map type loadbalance first-match PM_LB
  class class-default
    sticky-serverfarm STICKYGROUP1
policy-map multi-match PM_MULTI_MATCH
  class VS_WEB
    loadbalance vip inservice
    loadbalance policy PM_LB

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  service-policy input PM_MULTI_MATCH
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

domain DOMAIN1
add-object all

ip route 0.0.0.0 0.0.0.0 172.25.91.1
username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role SLB-Admin domain
DOMAIN1
```

Where to Go Next

In this chapter, you have configured server persistence with a sticky group that uses the HTTP-cookie method. In the next chapter, you will configure secure sockets layer (SSL) security.

CHAPTER 9

Configuring SSL Security

This chapter describes how to configure Secure Sockets Layer (SSL) on the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

Information About SSL

After reading this chapter, you should have a basic understanding of how the ACE provides SSL security for your network and how to configure SSL termination, in which the ACE operates as an SSL server.

SSL configuration in an ACE establishes and maintains a SSL session between the ACE and another device. It provides for secure data transactions between a client and a server. SSL provides authentication, encryption, and data integrity in a Public Key Infrastructure (PKI), which is a set of policies and procedures that establishes a secure information exchange between devices.

In SSL, data is encrypted using one or more symmetric keys that are known only by the two endpoints in the transaction. In a key exchange, one device generates the symmetric key and then encrypts it using an asymmetric encryption scheme before transmitting the key to the other device.

Asymmetric encryption requires each device to have a unique key pair consisting of a public key and a private key. A private key is an encryption/decryption key known only to the parties exchanging the messages. A public key is a value provided by some designated authority as an encryption key that, combined with a private key derived from the public key, can be used to effectively encrypt messages and digital signatures. The two keys are mathematically related; data that is encrypted using the public key can only be decrypted using the corresponding private key, and vice versa.

SSL facilitates client and server authentication through the use of digital certificates. Digital certificates are a form of digital identification to prove the identity of the server to the client, or optionally, the client to the server. A certificate ensures that the identification information is correct and the public key embedded in it actually belongs to the client or server.

A Certificate Authority (CA) issues digital certificates in the context of a PKI. CAs are trusted authorities that sign certificates to verify their authenticity. As the certificate issuer, the CA uses its private key to sign the certificate. Upon receiving a certificate, a client uses the issuer's public key to decrypt and verify the certificate signature to ensure that the certificate was actually issued and signed by an authorized entity.

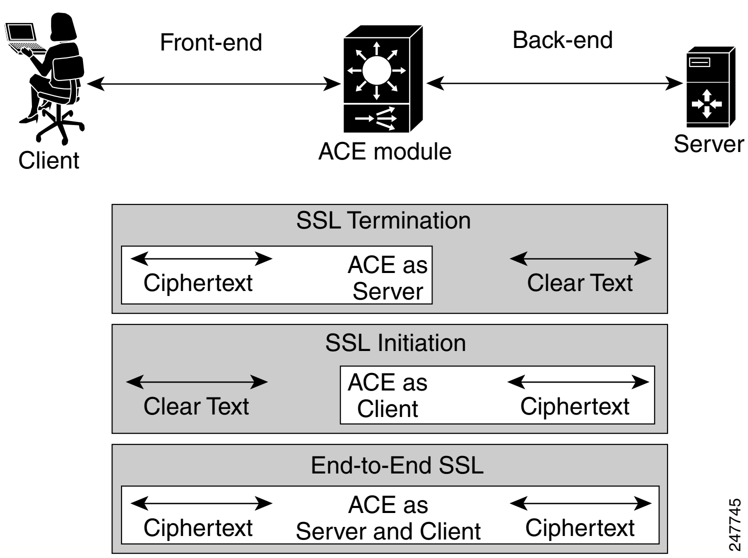
If you do not have a certificate and the corresponding key pair, you can use the ACE to generate a key pair and a certificate signing request (CSR) to apply for a certificate from a CA. The CA signs the CSR and returns the authorized digital certificate to you. The ACE supports import, export, and other management functions to manage the various certificates and key pair files within each context.

The client and server use the SSL handshake protocol to establish an SSL session between the two devices. During the handshake, the client and server negotiate the SSL parameters that they will use during the secure session. During the SSL handshake, the ACE uses an SSL proxy service, which includes the configuration of SSL session parameters, an RSA key pair, and a matching certificate.

The ACE applies SSL session parameters to an SSL proxy service. Creating an SSL parameter map allows you to apply the same SSL session parameters to different proxy services. The SSL session parameters include timeouts, close protocol behavior, and SSL version—SSL 3 and/or Transport Layer Security (TLS) 1. For more information on these parameters, see the *SSL Guide, Cisco ACE Application Control Engine*.

You can configure the ACE to act as a client or a server during an SSL session by defining operational attributes such as SSL session parameters, SSL key pairs and certificates, and traffic characteristics. When the traffic characteristics match the settings specified in the operational attributes, the ACE executes the actions associated with the SSL proxy service. Figure 9-1 shows the three basic SSL configurations in which the ACE is used to encrypt and decrypt data between the client and the server: SSL termination, SSL initiation, and end-to-end SSL.

Figure 9-1 ACE SSL Configurations



In SSL termination, an ACE context is configured for a front-end application in which the ACE operates as an SSL server that communicates with a client. When you define the flow between an ACE and a client, the ACE operates as a virtual SSL server by adding security services between a web browser (the client) and the HTTP connection (the server).

All inbound SSL flows that come from a client terminate at the ACE. After the connection is terminated, the ACE decrypts the ciphertext (encrypted content) from the client and sends the data as clear text (unencrypted content) to an HTTP server. For information about configuring the ACE for SSL termination, see the "Configuring SSL Termination" section.

In SSL initiation, an ACE context is configured for a back-end application in which the ACE operates as a client that communicates with an SSL server. When you define the flow between an ACE and an SSL server, the ACE operates as a client and initiates the SSL session. SSL initiation enables the ACE to receive clear text from a client and then establish an SSL session with an SSL server, joining the client and SSL server connections.

The ACE encrypts the clear text that it receives from the client and sends the data as ciphertext to an SSL server. The SSL server can either be an ACE configured for SSL termination (a virtual SSL server) or a real SSL server (web server). On the outbound flow from the SSL server, the ACE decrypts the ciphertext from the server and sends clear text back to the client. For more information on configuring the ACE for SSL initiation, see the *SSL Guide, Cisco ACE Application Control Engine*.

In end-to-end SSL, an ACE context is configured for both SSL termination and SSL initiation. You configure the ACE for end-to-end SSL when you have an application that requires secure SSL channels between the client and the ACE, and between the ACE and the SSL server.

For example, a transaction between banks requires end-to-end SSL to protect all financial information exchanged. End-to-end SSL also allows the ACE to insert load-balancing and security information into the data. The ACE decrypts the ciphertext that it receives and inserts load-balancing and firewall information into the clear text. The ACE then reencrypts the data and passes the ciphertext to its intended destination. For more information on configuring the ACE for end-to-end SSL initiation, see the *SSL Guide, Cisco ACE Application Control Engine*.

Licensing Requirements for SSL

By default, the ACE module provides 1000 SSL transactions per second (TPS). To increase the SSL transactions per second for the ACE module, you must obtain an optional bundle license from Cisco.

For details about licensing, see the *Administration Guide, Cisco ACE Application Control Engine*.

Prerequisites for Configuring SSL

Before configuring the ACE for an SSL operation, you must first configure it for server load balancing. To configure your ACE for server load balancing, see Chapter 6, Configuring Server Load Balancing

Before you configure SSL termination using the procedure in this chapter, you should have a signed SSL certificate and a key pair residing on an FTP server. For details about obtaining an SSL certificate and key pair, see the *SSL Guide, Cisco ACE Application Control Engine*.

If you do not have your own SSL certificate and key pair, for internal testing only, you can use the following Cisco-provided self-signed sample certificate and key pair files as follows:

- cisco-sample-cert
- cisco-sample-key

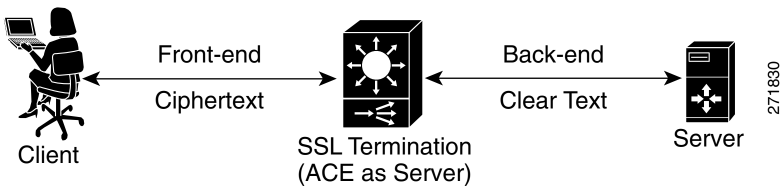
Configuring SSL Termination

SSL termination occurs when the ACE, acting as an SSL proxy server, terminates an SSL connection from a client and then establishes a TCP connection to an HTTP server. When the ACE terminates the SSL connection, it decrypts the ciphertext from the client and transmits the data as clear text to the HTTP server.

Figure 9-2 shows the following network connections in which the ACE terminates the SSL connection with the client:

- Client to ACE—An SSL connection exists between the client and the ACE acting as an SSL proxy server.
- ACE to Server—A TCP connection exists between the ACE and the HTTP server.

Figure 9-2 SSL Termination



SSL termination is a Layer 3 and Layer 4 application because it is based on the destination IP address of the inbound traffic flow from the client. When configuring a policy map for SSL termination, you associate the following elements:

- The SSL proxy service, including SSL session parameters, certificate, and key pair.
- The virtual SSL server IP address that the destination IP address of the inbound traffic must match (a class map). When a match occurs, the ACE negotiates with the client to establish an SSL connection.

Task Flow for Configuring SSL Termination

Follow these steps to configure SSL termination:

Step 1 Import the SSL certificate and key pair files and verify that the certificate matches the key pair.

Step 2 Create an SSL proxy service.

Step 3 Configure a traffic policy for SSL.

Importing the SSL Certificate and Key Pair Files

Procedure

	Command	Purpose
Step 1	**changeto** *context* **Example:** host1/Admin# changeto VC_WEB host1/VC_WEB#	Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.
Step 2	**crypto import ftp** *ip_address username* **/***remote_filename local_filename* **Example:** host1/VC_WEB# crypto import ftp 172.25.91.100 Admin /marketing.pem marketing.pem Password: **** Passive mode on. Hash mark printing on (1024 bytes/hash mark). # Successfully imported file from remote server. host1/VC_WEB#	Imports the key file marketing.pem from an FTP server. Use your own key pair file for this step. If you do not have a key file, see the *SSL Guide, Cisco ACE Application Control Engine* for information about how to obtain one and skip this step. Meanwhile, you can use the sample key file that is provided with your ACE software in a later step for internal testing.
Step 3	**crypto import terminal** *filename* **Example:** host1/VC_WEB# crypto import terminal marketing_cert.pem Enter PEM formatted data ending with a blank line or "quit" on a line by itself. -----------BEGIN CERTIFICATE----------- MIIC1DCCAj2gAwIBAgIDCCQAMA0GCSqGSIb3DQE BAgUAMIHEMQswCQYDVQQGEwJa QTEVMBMGA1UECBMMV2VzdGVybiBDYXBlMRIwEAY DVQQHEwlDYXBlIFRvd24xHTAb BgNVBAoTFFRoYXd0ZSBDb25zdWx0aW5nIGNjMSg wJgYDVQQLEx9DZXJ0aWZpY2F0 aW9uIFNlcnZpY2VzIERpdmlzaW9uMRkwFwYDVQQ DExBUaGF3dGUgU2VydmVyIENB MSYwJAYJKoZIhvcNAQkBFhdzZXJ2ZXItY2VydHN AdGhhd3RlLmNvbTAeFw0wMTA3 -----------END CERTIFICATE-------------	Copies the certificate information from the certificate that you received from the CA, and pastes it into a certificate file called marketing_cert.pem. Use your own certificate file for this step. If you do not have a signed certificate, see the *SSL Guide, Cisco ACE Application Control Engine* for information about how to obtain one and skip this step. Meanwhile, you can use the Cisco-provided sample self-signed certificate in a later step for internal testing.

	Command	Purpose

Step 4	**quit**
	
	Example:
	`quit`
	`host1/VC_WEB#`
		Closes the file.

Step 5	**crypto verify** *key_filename cert_fielname*
	
	Example:
	`host1/VC_WEB# crypto verify cisco-sample-key cisco-sample-cert`
	`keypair in cisco-sample-key matches certificate in cisco-sample-cert`
		Verifies that the certificate matches the key pair. If you have not obtained your own SSL certificate and key pair files yet, you can use the Cisco-provided certificate (cisco-sample-cert) and key (cisco-sample-key) for internal testing.

Creating an SSL Proxy Service

Procedure

Step 1	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 2	**ssl-proxy service** *pservice_name*
	
	Example:
	`host1/Admin(config)# ssl-proxy service PS_SSL_TERMINATION`
	`host1/Admin(config-ssl-proxy)#`
		Creates the PS_SSL_TERMINATION SSL proxy service.

Step 3	**key** *key_filename* **Example:** `host1/VC_WEB(config-ssl-proxy)# key cisco-sample-key`	Specifies the key pair filename in the SSL proxy. Use your own key pair filename or enter the Cisco-provided sample key pair filename (cisco-sample-key) for internal testing.
Step 4	**cert** *cert_filename* **Example:** `host1/VC_WEB(config-ssl-proxy)# cert cisco-sample-cert`	Specifies the certificate filename in the SSL proxy. Use your own key pair filename or enter the Cisco-provided sample certificate filename (cisco-sample-cert) for internal testing.

Step 5	**exit**
	
	Example:
	`host1/VC_WEB(config-ssl-proxy)# exit`
	`host1/VC_WEB(config)#`
		Exits SSL proxy configuration mode.

Configuring a Traffic Policy for SSL Termination

Procedure

	Command	Purpose
Step 1	**class-map** *map_name* **Example:** `host1/VC_WEB(config)# class-map CM_SSL`	Creates a Layer 3 and Layer 4 class map.
Step 2	**match virtual-address** *vip_address* **tcp eq** *port* **Example:** `host1/VC_WEB(config-cmap)# match virtual-address 10.10.40.11 tcp eq https`	Configures the class map with the input traffic match criteria, including the VIP and the TCP protocol.

Step 3	**exit**
	
	Example:
	`host1/VC_WEB(config-cmap)# exit`
	`host1/VC_WEB(config)#`
		Exits class map configuration mode.

Step 4	**policy-map multi-match** *map_name*
	
	Example:
	`host1/VC_WEB(config)# policy-map multi-match PM_MULTI_MATCH`
	`host1/VC_WEB(config-pmap)#`
		Enters policy map configuration mode.

Step 5	**class** *name*
	
	Example:
	`host1/VC_WEB(config-pmap)# class CM_SSL`
	`host1/VC_WEB(config-pmap-c)#`
		Associates the CM_SSL class map with the multimatch policy map.

Step 6	**loadbalance VIP inservice** **Example:** `host1/VC_WEB(config-pmap-c)# loadbalance vip inservice`	Enables a VIP for load balancing operations.
Step 7	**loadbalance policy PM_LB** **Example:** `host1/VC_WEB(config-pmap-c)# loadbalance policy PM_LB`	Associates the PM_LB Layer 7 load-balancing policy map with the PM_MULTI_MATCH Layer 3 and Layer 4 policy map.
Step 8	**ssl-proxy server** *name* **Example:** `host1/VC_WEB(config-pmap-c)# ssl-proxy server PS_SSL_TERMINATION`	Associates the SSL proxy service PS_SSL_TERMINATION with the policy map.

Step 9	**exit**
	
	Example:
	`host1/VC_WEB(config-pmap-c)# exit`
	`host1/VC_WEB(config-pmap)# exit`
	`host1/VC_WEB(config)#`
		Exits policy map class configuration mode. Exits policy map configuration mode.

	Command	Purpose
Step 10	**interface vlan** *vlan_id* **Example:** `host1/VC_WEB(config)# interface vlan 400`	Enters interface configuration mode for the client-side VLAN interface 400.
Step 11	**service-policy input** *map_name* **Example:** `host1/VC_WEB(config-if)# service-policy input PM_SSL`	Applies the policy map to the input traffic of the VLAN 400 interface.

Step 12	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits interface configuration mode. Exits configuration mode.

Step 13	**show running-config** **Example:** `host1/VC_WEB# show running-config`	Displays the running configuration to verify that the information that you just added is configured properly.
Step 14	**copy running-config startup-config** **Example:** `host1/VC_WEB# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for SSL Termination

The following example shows how to configure SSL termination. The commands that you have configured in this chapter appear in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

rserver host RS_WEB1
  description content server web-one
  ip address 10.10.50.10
  inservice
rserver host RS_WEB2
  description content server web-two
  ip address 10.10.50.11
  inservice
rserver host RS_WEB3
  description content server web-three
  ip address 10.10.50.12
  inservice
rserver host RS_WEB4
  description content server web-four
  ip address 10.10.50.13
  inservice

serverfarm host SF_WEB
  predictor hash header Accept
  rserver RS_WEB1 80
    inservice
```

```
  rserver RS_WEB2 80
    inservice
  rserver RS_WEB3 80
    inservice
  rserver RS_WEB4 80
    inservice

sticky http-cookie Cookie1 StickyGroup1
  timeout 3600
  serverfarm SF_WEB

ssl-proxy service PS_SSL_TERMINATION
  key cisco-sample-key
  cert cisco-sample-cert

class-map match-all CM_SSL
  2 match virtual-address 10.10.40.11 tcp eq https
class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any
class-map match-all VS_WEB
  2 match virtual-address 10.10.40.10 tcp eq www

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

policy-map type loadbalance first-match PM_LB
  class class-default
    serverfarm SF_WEB

policy-map multi-match PM_MULTI_MATCH
  class VS_WEB
    loadbalance vip inservice
    loadbalance policy PM_LB
  class CM_SSL
    loadbalance vip inservice
    loadbalance policy PM_LB
    ssl-proxy server PS_SSL_TERMINATION

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  service-policy input PM_MULTI_MATCH
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

domain DOMAIN1
add-object all

ip route 0.0.0.0 0.0.0.0 172.25.91.1
username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role SLB-Admin domain
DOMAIN1
```

Where to Go Next

In this chapter, you have configured an SSL proxy service and a virtual server for SSL termination. In the next chapter, you will configure server health monitoring probes (keepalives).

CHAPTER 10

Configuring Health Monitoring Using Health Probes

This chapter describes how to configure a health probe on the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

- Information About Configuring Health Monitoring
- Prerequisites for Configuring Health Monitoring
- Configuring an HTTP Health Probe
- Configuration Example for an HTTP Health Probe
- Where to Go Next

Information About Configuring Health Monitoring

After reading this chapter, you should have a basic understanding of how the ACE supports server health monitoring using health probes (sometimes referred to as "keepalives"), and how to configure an HTTP health probe.

To detect failures and make reliable load-balancing decisions, you can configure the ACE to track the health of servers and server farms by periodically sending out health probes. By default, the ACE implicitly checks for server failures.

You can configure probes on the ACE to make active connections and explicitly send traffic to servers. The ACE evaluates the server's response to determine the health of that server.

When the ACE determines the health of a server, the result is one of the following:

- Passed—The server returned a valid response.
- Failed—The server failed to provide a valid response to the ACE within a specified number of retries.

When a server fails in response to the probe, the ACE can check for network problems that prevent a client from accessing that server. The ACE can place the server out of service.

A probe can be any of several types, including HTTP, HTTPS, ICMP, TCP, Telnet, and UDP. You can also configure scripted probes using the TCL scripting language.

This chapter describes how to configure an HTTP probe. For information on how to configure other types of probes, see the *Server Load-Balancing Guide, Cisco ACE Application Control Engine.*

Prerequisites for Configuring Health Monitoring

Before you can configure health monitoring, you must configure one or more servers or a server farm. For details, see Chapter 6, Configuring Server Load Balancing.

Configuring an HTTP Health Probe

Procedure

	Command	Purpose

Step 1	**changeto** *context*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**probe http** *name*
	
	Example:
	`host1/VC_WEB(config)# probe http HTTP_PROBE1`
	`host1/VC_WEB(config-probe-http)#`
		Define an HTTP probe named HTPP_probe1 to access its configuration mode.

Step 4	**expect status** *min_number max_number* **Example:** `host1/VC_WEB(config-probe-http)# expect status 200 200`	Configures a single status code or a range of status code responses that the ACE expects from the probe destination. This parameter is required. Without it, all HTTP or HTTPS probes will fail.

Step 5	**exit**
	
	Example:
	`host1/VC_WEB(config-probe-http)# exit`
	`host1/VC_WEB(config)#`
		Exits probe configuration mode.

Step 6	**serverfarm** *name*
	
	Example:
	`host1/VC_WEB(config)# serverfarm SF_WEB`
	`host1/VC_WEB(config-sfarm-host)#`
		Enter server farm host configuration mode for the SF_WEB server farm.

Step 7	**probe** *name* **Example:** `host1/VC_WEB(config-sfarm-host)# probe HTTP_PROBE1`	Associate the probe HTTP_PROBE1 with the server farm SF_WEB.

Step 8	**exit**
	
	Example:
	`host1/VC_WEB(config-sfarm-host)# exit`
	`host1/VC_WEB(config)# exit`
	`host1/VC_WEB#`
		Exits server farm host configuration mode. Exits configuration mode.

	Command	Purpose
Step 9	**show running-config probe** **Example:** host1/VC_WEB# show running-config probe	Displays the running configuration to verify that the information that you just added is configured properly.
Step 10	**copy running-config startup-config** **Example:** host1/VC_WEB# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for an HTTP Health Probe

The following example shows how to configure an HTTP health probe. The commands that you have configured in this chapter appear in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

probe http HTTP_PROBE1
  expect status 200 200

rserver host RS_WEB1
  description content server web-one
  ip address 10.10.50.10
  inservice
rserver host RS_WEB2
  description content server web-two
  ip address 10.10.50.11
  inservice
rserver host RS_WEB3
  description content server web-three
  ip address 10.10.50.12
  inservice
rserver host RS_WEB4
  description content server web-four
  ip address 10.10.50.13
  inservice

serverfarm host SF_WEB
  predictor hash header Accept
  probe HTTP_PROBE1
  rserver RS_WEB1 80
    inservice
  rserver RS_WEB2 80
    inservice
  rserver RS_WEB3 80
    inservice
  rserver RS_WEB4 80
    inservice

sticky http-cookie Cookie1 StickyGroup1
  timeout 3600
  serverfarm SF_WEB
```

```
ssl-proxy service SSL_PSERVICE_SERVER
  key cisco-sample-key
  cert cisco-sample-cert

class-map match-all CM_SSL
  2 match virtual-address 10.10.40.11 tcp eq https
class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any
class-map match-all VS_WEB
  2 match virtual-address 10.10.40.10 tcp eq www

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

policy-map type loadbalance first-match PM_LB
  class class-default
    serverfarm SF_WEB

policy-map multi-match PM_MULTI_MATCH
  class VS_WEB
    loadbalance vip inservice
    loadbalance policy PM_LB
policy-map multi-match PM_SSL
  class CM_SSL
    ssl-proxy server SSL_PSERVICE_SERVER

service-policy input REMOTE_MGMT_ALLOW_POLICY

interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  service-policy input PM_MULTI_MATCH
  service-policy input PM_SSL
  no shutdown
interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

domain DOMAIN1
add-object all

ip route 0.0.0.0 0.0.0.0 172.25.91.1
username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role SLB-Admin domain
DOMAIN1
```

Where to Go Next

In this chapter, you have configured an HTTP health probe. In the next chapter, you will configure route health injection (RHI).

CHAPTER 11

Configuring Route Health Injection

This chapter describes how to configure route health injection (RHI) for the Cisco Application Control Engine (ACE) module.

This chapter contains the following sections:

- Information About RHI
- Configuring Route Health Injection
- Configuration Example for Route Health Injection
- Where to Go Next

Information About RHI

After reading this chapter, you should have a basic understanding of what RHI is, how it works in the ACE, and how to configure it to advertise a VIP.

Route Health Injection (RHI) allows the ACE to advertise the availability of a VIP address throughout the intranet as a host route. The ACE send this RHI information to the MSFC in the Catalyst 6500 series switch or the Cisco 7600 series router, which periodically propagates the VIP availability according to the RHI information it receives. RHI is normally restricted to intranets because the MSFC does not broadcast host-route availability to the Internet.

The ACE uses health probes (configured in Chapter 10, Configuring Health Monitoring Using Health Probes) together with RHI to determine the availability of a VIP before advertising it. When a VIP becomes unavailable, the ACE withdraws the RHI information. The MSFC adds an entry in its routing table for each VIP address it receives from the ACE. The routing protocol running on the MSFC sends routing-table updates, including availability and hop-count routing information for each instance of a VIP address to other routers. The client router uses the routing information to choose a route based on best available path to that VIP address and also where the Cisco application switch is logically closer to the client system.

RHI is aware of virtual routing and forwarding (VRF) allowing ACE virtual devices to inject and remove routes directly from VRF routing tables in the supervisor engine.

By default, the ACE advertises the VLAN of the VIP interface for RHI. To advertise a VLAN for route health injection (RHI) that is different from the VIP interface VLAN, use the **ip route inject vlan** command in interface configuration mode. By default, the ACE advertises the VLAN of the VIP interface for RHI. Use this command when there is no directly shared VLAN between the ACE and the Catalyst 6500 series supervisor engine. This topology can occur when there is an intervening device, for example, a Cisco Firewall Services Module (FWSM), configured between the ACE and the supervisor engine. Be sure to configure this command on the VIP interface of the ACE.

Configuring Route Health Injection

Procedure

	Command	Purpose

Step 1	**changeto** *context*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**policy-map multi-match** *name*
	
	Example:
	`host1/VC_WEB(config)# policy-map multi-match PM_MULTI_MATCH`
	`host1/VC_WEB(config-pmap)#`
		Accesses the PM_MULTI_MATCH Layer 3 and Layer 4 multi-match policy map that you created in Chapter 6, Configuring Server Load Balancing.

Step 4	**class** *name*
	
	Example:
	`host1/VC_WEB(config-pmap)# class VS_WEB`
	`host1/VC_WEB(config-pmap-c)#`
		Accesses the VS_WEB Layer 3 and Layer 4 class map that you created in Chapter 6, Configuring Server Load Balancing.

Step 5	**loadbalance vip advertise** [**active**] | [**metric number**] **Example:** `host1/VC_WEB(config-pmap-c)# loadbalance vip advertise active`	Enables the ACE to advertise the availability of a VIP address throughout the network. Without the **active** option, the ACE always advertises the VIP whether or not there is any active real server associated with this VIP. You must enable the advertising of a VIP using the **loadbalance vip advertise** command before you can enter a distance metric value for the route. Otherwise, the ACE returns an error message.

Step 6	**exit**
	
	Example:
	`host1/VC_WEB(config-pmap-c)# exit`
	`host1/VC_WEB(config-pmap)# exit`
	`host1/VC_WEB(config)#`
		Exits policy map class configuration mode. Exits policy map configuration mode. Alternatively, you can press Ctrl-G to exit one mode.

	Command	Purpose

Step 7	**ip route inject vlan** *vlan_id*
	
	Example:
	`host1/VC_WEB(config)# interface vlan 400`
	`host1/VC_WEB(config-if)# ip route inject vlan 200`
		(Optional) Advertises a VLAN for route health injection (RHI) that is different from the VIP interface VLAN.
		
		The *vlan_id* is the interface shared between the supervisor engine and the intervening device.
		
		Use this command when there is no directly shared VLAN between the ACE and the Catalyst 6500 series supervisor engine. This topology can occur when there is an intervening device, for example, a Cisco Firewall Services Module (FWSM), configured between the ACE and the supervisor engine.
		
		Be sure to configure this command on the VIP interface of the ACE.

Step 8	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config) exit`
	`host1/VC_WEB#`
		Exits interface configuration mode. Exits configuration mode.

Step 9	**show running-config policy-map** *policy_name* **Example:** `host1/VC_WEB# show running-config policy-map PM_MULTI_MATCH`	Displays the policy-map configuration information.
Step 10	**copy running-config startup-config** **Example:** `host1/VC_WEB# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for Route Health Injection

The following example shows how to configure RHI. The commands that you have configured in this chapter appear in bold text.

```
switch/VC_WEB(config)# do show running config
Generating configuration....

access-list INBOUND line 8 extended permit ip any any

probe http HTTP_PROBE1
  expect status 200 200

rserver host RS_WEB1
  description content server web-one
  ip address 10.10.50.10
  inservice
```

```
rserver host RS_WEB2
  description content server web-two
  ip address 10.10.50.11
  inservice
rserver host RS_WEB3
  description content server web-three
  ip address 10.10.50.12
  inservice
rserver host RS_WEB4
  description content server web-four
  ip address 10.10.50.13
  inservice

serverfarm host SF_WEB
  predictor hash header Accept
  probe HTTP_PROBE1
  rserver RS_WEB1 80
    inservice
  rserver RS_WEB2 80
    inservice
  rserver RS_WEB3 80
    inservice
  rserver RS_WEB4 80
    inservice

sticky http-cookie Cookie1 StickyGroup1
  timeout 3600
  serverfarm SF_WEB

ssl-proxy service SSL_PSERVICE_SERVER
  key cisco-sample-key
  cert cisco-sample-cert

class-map match-all CM_SSL
  2 match virtual-address 10.10.40.11 tcp eq https
class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol ssh any
  3 match protocol telnet any
  4 match protocol icmp any
class-map match-all VS_WEB
  2 match virtual-address 10.10.40.10 tcp eq www

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

policy-map type loadbalance first-match PM_LB
  class class-default
    serverfarm SF_WEB

policy-map multi-match PM_MULTI_MATCH
  class VS_WEB
    loadbalance vip inservice
    loadbalance policy PM_LB
    loadbalance vip advertise active
policy-map multi-match PM_SSL
  class CM_SSL
    ssl-proxy server SSL_PSERVICE_SERVER

service-policy input REMOTE_MGMT_ALLOW_POLICY
```

```
interface vlan 400
  description Client connectivity on VLAN 400
  ip address 10.10.40.1 255.255.255.0
  access-group input INBOUND
  service-policy input PM_MULTI_MATCH
  service-policy input PM_SSL
  no shutdown
  ip route inject vlan 200

interface vlan 500
  description Server connectivity on VLAN 500
  ip address 10.10.50.1 255.255.255.0
  no shutdown

domain DOMAIN1
add-object all

ip route 0.0.0.0 0.0.0.0 172.25.91.1
username USER1 password 5 $1$vAN9gQDI$MmbmjQgJPj45lxbtzXPpB1  role SLB-Admin domain
DOMAIN1
```

Where to Go Next

In this chapter, you have enabled the RHI feature to advertise the availability of a VIP address. In the next chapter, you will learn how to configure redundancy or fault tolerance.

CHAPTER 12

Configuring Redundant ACE Modules

This chapter describes how to configure the Cisco Application Control Engine (ACE) module for redundancy, which provides fault tolerance for the stateful switchover of flows.

This chapter contains the following sections:

Information About Redundancy

After reading this chapter, you should have a basic understanding of ACE redundancy and how to configure it. For detailed information on redundancy, see the *Administration Guide, Cisco ACE Application Control Engine.*

The redundancy (or fault tolerance) feature ensures that your network services and applications are always available. It provides seamless switchover of flows in case an ACE becomes unresponsive or a critical host, interface, or HSRP group fails.

This feature uses a maximum of two ACEs (peers) in the same Catalyst 6500 series switch or in separate switches. Each peer module can contain one or more fault-tolerant (FT) groups. Each FT group consists of two members: one active context and one standby context. For more information about contexts, see the *Virtualization Guide, Cisco ACE Application Control Engine*.

To outside nodes (clients and servers), the active and standby FT group members appear as one node with respect to their IP addresses and associated virtual MAC (VMAC) addresses. The ACE provides active-active redundancy with multiple-contexts only when there are multiple FT groups configured on each module and both modules contain at least one active group member (context). With a single context, the ACE supports active-backup redundancy and each group member is an Admin context.

Each FT group acts as an independent redundancy instance. When a switchover occurs, the active member in the FT group becomes the standby member and the original standby member becomes the active member.

The ACE sends and receives all redundancy-related traffic (protocol packets, configuration data, heartbeats, and state replication packets) on a dedicated FT VLAN that is not used for normal traffic. The active ACE automatically replicates the configuration, including changes made to the configuration, on the standby peer using a process called *configuration synchronization* (config sync). After the ACE synchronizes the redundancy configuration from the active member to the standby peer, it disables configuration mode on the standby.

The two redundant modules constantly communicate over the FT VLAN to determine the operating status of each module. The standby member uses the heartbeat packet to monitor the health of the active member. The active member uses the heartbeat packet to monitor the health of the standby member. The ACE uses the heartbeat to probe the peer ACE, rather than probe each context. When an ACE does not receive a heartbeat from the peer ACE, all the contexts in the standby state become active. The ACE sends heartbeat packets over UDP. You can set the frequency with which the ACE sends heartbeat packets as part of the FT peer configuration.

The ACE replicates flows on the active FT group member to the standby group member per connection for each context. The replicated flows contain all the flow-state information necessary for the standby member to take over the flow if the active member becomes unresponsive. If the active member becomes unresponsive, the replicated flows on the standby member become active when the standby member assumes mastership of the context. The active flows on the former active member transition to a standby state to fully back up the active flows on the new active member.

After a switchover occurs, the same connection information is available on the new active member. Supported end-user applications do not need to reconnect to maintain the same network session.

This chapter describes how to configure each ACE in a redundant configuration.

Guidelines and Limitations

Follow these guidelines and limitations when you configure the redundancy feature:

- You can configure redundancy only in the Admin context.
- Redundancy is not supported between an ACE module and an ACE appliance operating as peers. Redundancy must be of the same ACE device type and software release.
- For redundancy to function properly, both members of an FT group must have identical configurations. Ensure that both ACE modules include the same bandwidth software license (4 Gbps, 8 Gbps, or 16 Gbps) and the same virtual context software license. If there is a mismatch in a software license between the two ACE modules in an FT group, the following operational behavior can occur:
  - If there is a mismatch in the virtual context software license, synchronization between the active ACE and standby ACE may not work properly.
  - If both the active and the standby ACE modules have the same virtual content software license but have a different bandwidth software license, synchronization will work properly but the standby ACE may experience a potential loss of traffic on switchover from, for example, an 8-Gbps ACE module to a 4-Gbps ACE module.
- Redundancy uses a dedicated FT VLAN between redundant ACEs to transmit flow-state information and the redundancy heartbeat. Do not use this dedicated VLAN for any other network traffic, including HSRP and data. You must configure this same VLAN on both peer modules. You also must configure a different IP address within the same subnet on each module for the FT VLAN.
- In bridged mode (Layer 2), two contexts cannot share the same VLAN.
- The IP address and the MAC address of the FT VLAN do not change at switchover.

- For multiple contexts, the FT VLAN resides in the system configuration file. Each FT VLAN on the ACE has one unique MAC address associated with it. The ACE uses these device MAC addresses as the source or destination MACs for sending or receiving redundancy protocol state and configuration replication packets.
- One virtual MAC address (VMAC) is associated with each FT group. The format of the VMAC is 00-0b-fc-fe-1b–*groupID*. Because a VMAC does not change upon switchover, the client and server ARP tables do not require updating. The ACE selects a VMAC from a pool of virtual MACs available to it. For more information about VMACs, see the *Routing and Bridging Guide, Cisco ACE Application Control Engine*.
- In a user context, the ACE allows a switchover only of the FT group that belongs to that context. In the Admin context, the ACE allows a switchover of all FT groups in all configured contexts in the module.
- To achieve active-active redundancy, a minimum of two contexts and two FT groups are required on each ACE.
- When you configure redundancy, the ACE keeps all interfaces that do not have an IP address in the Down state. The IP address and the peer IP address that you assign to a VLAN interface should be in the same subnet, but different IP addresses. For more information about configuring VLAN interfaces, see the *Routing and Bridging Guide, Cisco ACE Application Control Engine*.
- By default, connection replication is enabled in the ACE and is not configurable.
- The ACE does not replicate SSL and other terminated (proxied) connections from the active context to the standby context.
- You must manually copy the SSL certificates and keys to the standby ACE. You can use the **crypto import** command.
- You must manually copy scripts to the standby ACE.

Configuring Redundancy

This section describes how to configure redundancy. You must configure *each* ACE in the fault-tolerant (FT) group. It contains the following topics:

- Task Flow for Configuring Redundancy
- Configuring an FT VLAN
- Configuring an FT Peer
- Configuring an Alias IP Address
- Configuring an FT Group

Task Flow for Configuring Redundancy

Follow these steps to configure redundancy:

Step 1 Configure a dedicated FT VLAN.

Step 2 Configure an FT peer, including a query VLAN.

Step 3 Configure an alias IP address as the shared gateway for the two ACEs.

Step 4 Configure an FT group.

Configuring an FT VLAN

Procedure

	Command	Purpose

Step 1	**`changeto`** *`context`*
	
	`Example:`
	`host1/VC_WEB# changeto Admin`
	`host1/Admin#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**`config`**
	
	`Example:`
	`host1/Admin# config`
	`host1/Admin(config)#`
		Enters configuration mode.

Step 3	**`ft interface vlan`** *`number`*
	
	`Example:`
	`host1/Admin(config)# ft interface vlan 60`
	`host1/Admin(config-ft-intf)#`
		Configures a dedicated FT VLAN for communication between the members of the FT group. This FT VLAN is global and is shared by all contexts.

Step 4	**`ip address`** *`address netmask`* **`Example:`** `host1/Admin(config-ft-intf)# ip address 10.10.60.10 255.255.255.0`	Specifies the IP address and netmask of the FT VLAN.
Step 5	**`peer ip address`** *`address netmask`* **`Example:`** `host1/Admin(config-ft-intf)# peer ip address 10.10.60.11 255.255.255.0`	Specifies the IP address and netmask of the remote peer.

Step 6	**`exit`**
	
	`Example:`
	`host1/Admin(config-ft-intf)# exit`
	`host1/Admin(config)#`
		Exits FT interface configuration mode.

	Command	Purpose
Step 7	**`do show running-config ft`** **Example:** `host1/Admin(config)# do show running-config ft`	Verifies the redundancy configuration.
Step 8	**`do copy running-config startup-config`** **Example:** `host1/Admin(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring an FT Peer

Procedure

	Command	Purpose

Step 1	**`ft peer`** *`number`*
	
	Example:
	`host1/Admin(config)# ft peer 1`
	`host1/Admin(config-ft-peer)#`
		Configures the local redundancy peer.

Step 2	**`ft-interface vlan`** *`number`* **Example:** `host1/Admin(config-ft-peer)# ft-interface vlan 60`	Associates the FT VLAN with the peer.
Step 3	**`heartbeat count`** *`number`* **Example:** `host1/Admin(config-ft-peer)# heartbeat count 20`	Configures the heartbeat count.
Step 4	**`heartbeat interval`** *`seconds`* **Example:** `host1/Admin(config-ft-peer)# heartbeat interval 300`	Configures the heartbeat interval in milliseconds.
Step 5	**`query-interface vlan`** *`vlan_id`* **Example:** `host1/Admin(config-ft-peer)# query-interface vlan 1000`	Configures a query interface to allow the standby member to determine whether the active member is down or if there is a connectivity problem with the FT VLAN. A query interface helps prevent two redundant contexts from becoming active at the same time for the same FT group. Before triggering a switchover, the ACE pings the active member to make sure that it is down. Configuring a query interface allows you to assess the health of the active member, but it increases the switchover time. The *vlan_id* argument specifies the identifier of an existing VLAN. Enter an integer from 2 to 4094. In this example, use VLAN 1000.

	Command	Purpose

Step 6	**exit**
	
	Example:
	`host1/Admin(config-ft-peer)# exit`
	`host1/Admin(config)#`
		Exits FT peer configuration mode.

Step 7	**do show running-config ft** **Example:** `host1/Admin(config)# do show running-config ft`	Verifies the redundancy configuration.
Step 8	**do copy running-config startup-config** **Example:** `host1/Admin(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring an Alias IP Address

An alias IP address serves as the shared gateway for the two ACEs. If you want to configure only one ACE for redundancy initially (for example, your second ACE will arrive a week or two after the first one), you must complete the redundancy configuration as described in this chapter to use the alias IP address. Otherwise, the alias IP address will be inoperable.

Note The alias IP address is the IP address that the real servers will use as their default gateway. If you do not configure an alias IP address on the VLAN, the ACE will fail over, however, the servers will not be able to route because the primary address will no longer exist in a failure.

Procedure

	Command	Purpose
Step 1	**interface vlan 1000** **Example:** `host1/Admin(config)# interface vlan 1000`	Enters interface VLAN configuration mode for VLAN 1000.
Step 2	**alias ip address** *ip_address netmask* **Example:** `host1/Admin(config-intf-config)# alias ip address 172.25.91.112 255.255.255.0`	Configures an alias IP address that floats between the active and the standby ACEs.

Step 3	**exit**
	
	Example:
	`host1/Admin(config-intf-config)# exit`
	`host1/Admin(config)#`
		Exits interface configuration mode.

	Command	Purpose
Step 4	**do show running-config ft** **Example:** `host1/Admin(config)# do show running-config ft`	Verifies the redundancy configuration.
Step 5	**do copy running-config startup-config** **Example:** `host1/Admin(config)# do copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring an FT Group

Procedure

	Command	Purpose

Step 1	**ft group** *number*
	
	Example:
	`host1/Admin(config)# ft group 1`
	`host1/Admin(config-ft-group)#`
		Creates an FT group. Create at least one FT group on each ACE.

Step 2	**associate-context** *name* **Example:** `host1/Admin(config-ft-group)# associate-context VC_WEB`	Associates a context with each FT group. You must associate the local context and the corresponding peer context with the same FT group.
Step 3	**peer** *number* **Example:** `host1/Admin(config-ft-group)# peer 1`	Associates the peer context with the FT group.
Step 4	**inservice** **Example:** `host1/Admin(config-ft-group)# inservice`	Places the FT group in service.

Step 5	**exit**
	
	Example:
	`host1/Admin(config-ft-group)# exit`
	`host1/Admin(config)#`
		Exits FT group configuration mode.

Step 6	**ft auto-sync running-config** | **startup-config**
	
	Example:
	`host1/Admin(config)# ft auto-sync running-config`
	`host1/Admin(config)# ft auto-sync startup-config`
		(Optional) Enables autosynchronization of the running-configuration and/or startup-configuration file from the active to the standby context. Both commands are enabled by default.

	Command	Purpose
Step 7	**do show running-config ft | interface** **Example:** host1/Admin(config)# do show running-config ft host1/Admin(config)# do show running-config interface	Verifies the redundancy configuration.
Step 8	**do copy running-config startup-config** **Example:** host1/Admin(config)# do copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Now that you have configured redundancy on one ACE, configure the other ACE in the FT group in a similar manner.

Configuration Example for Redundancy

The following example shows how to configure redundancy in the Admin context. The commands that you have configured in this chapter appear in bold text.

```
switch/Admin(config)# do show run
Generating configuration....

login timeout 0

resource-class RC_WEB
  limit-resource all minimum 10.00 maximum equal-to-min

class-map type management match-any REMOTE_ACCESS
  description Remote access traffic match
  2 match protocol telnet any
  3 match protocol ssh any
  4 match protocol icmp any

policy-map type management first-match REMOTE_MGMT_ALLOW_POLICY
  class REMOTE_ACCESS
    permit

interface vlan 1000
  description Management connectivity on VLAN 1000 and query interface VLAN
  ip address 172.25.91.110 255.255.255.0
  peer ip address 172.25.91.111 255.255.255.0
  alias ip address 172.25.91.112 255.255.255.0
  service-policy input REMOTE_MGMT_ALLOW_POLICY
  no shutdown

ft interface vlan 60
  ip address 10.10.60.10 255.255.255.0
  peer ip address 10.10.60.11 255.255.255.0

ft peer 1
  heartbeat interval 300
  heartbeat count 20
  ft-interface vlan 60
  query-interface vlan 1000
```

```
domain DOMAIN1
add-object all

ip route 0.0.0.0 0.0.0.0 172.25.91.1

context VC_WEB
  allocate-interface vlan 400
  allocate-interface vlan 500
  allocate-interface vlan 1000
  member RC_WEB

ft group 1
  peer 1
  associate-context VC_WEB
  inservice

username admin password 5 $1$JwBOOUEt$jihXQiAjF9igwDay1qAvK.  role Admin domain
default-domain
username www password 5 $1$xmYMkFnt$n1YUgNOo76hAhg.JqtymF/  role Admin domain
default-domain
```

Where to Go Next

In this chapter, you have configured redundancy on the ACE. In the next chapter, you will learn how to configure bridged mode.

CHAPTER 13

Configuring Bridged Mode

This chapter describes how to configure the Cisco Application Control Engine (ACE) module to bridge traffic on a single IP subnet.

This chapter includes the following topics:

- Information About Configuring Bridged Mode
- Guidelines and Limitations
- Task Flow for Configuring Bridged Mode
- Configuring Bridged Mode on the ACE
- Configuration Example for Bridged Mode
- Where to Go Next

Information About Configuring Bridged Mode

After reading this chapter, you should have a basic understanding of bridged mode, how it works in the ACE, and how to configure it.

Up to this point in this guide, you have been configuring the ACE in routed mode. Routed mode treats the ACE as a next hop in the network, typically with a client-side VLAN and a server-side VLAN in different IP subnets or even in different IP networks. The VLAN interfaces rely on IP addresses to route packets from one subnet or network to another.

In bridged mode, the ACE bridges traffic between two VLANs in the same IP subnet. The VLAN facing the WAN is the client-side VLAN. The VLAN facing the data center is the server-side VLAN. A bridge group virtual interface (BVI) joins the two VLANs into one bridge group.

As traffic passes through the client-side VLAN, the ACE evaluates the traffic with the configured service policy. Traffic that matches a policy is redirected to a server that has a dedicated VLAN interface configured on the ACE. Traffic leaving the server goes to the ACE, where it is directed out of the server side VLAN to the origin server. Traffic is routed by means of static routing. No dynamic routing protocols are required.

Prerequisites

Bridged mode on an ACE has the following prerequisites:

- Contact your network administrator to determine which VLANs and addresses are available for use by the ACE. Then, configure VLANs for the ACE using the Cisco IOS Software (see the "Configuring VLANs for the ACE Using Cisco IOS Software" section).
- Configure a default route on the ACE to identify an IP address for the ACE to send all IP packets for which it does not have a route (see the "Configuring a Default Route" section).
- Configure an access list to allow traffic (see the "Configuring an ACL" section).

Guidelines and Limitations

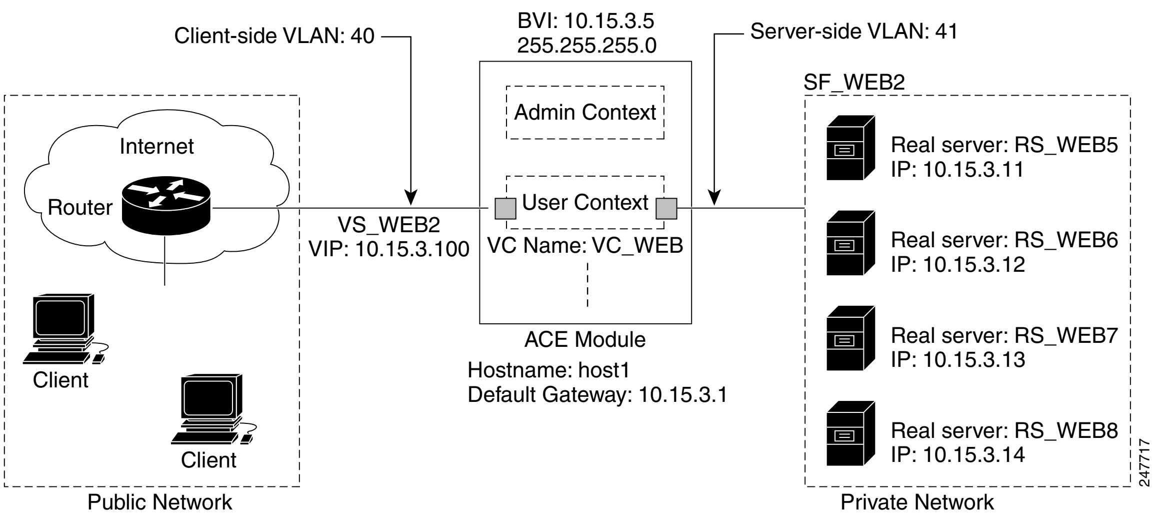
Bridged mode on the ACE has the following configuration guidelines and limitations:

- The ACE supports 4,094 BVIs per system.
- The ACE supports a maximum of 8,192 interfaces per system that include VLANs, shared VLANs, and BVI interfaces.
- When you configure a bridge group on an interface VLAN, the ACE automatically makes it a bridged interface.
- The ACE supports a maximum of two Layer 2 interface VLANs per bridge group.
- The ACE does not allow shared VLAN configurations on Layer 2 interfaces.
- Because Layer 2 VLANs are not associated with an IP address, they require extended access control lists (ACLs) for controlling IP traffic. You can also optionally configure EtherType ACLs to pass non-IP traffic.
- The ACE does not perform MAC address learning on a bridged interface. Instead, learning is performed by ARP. Bridge lookup is based on the bridge-group identifier and destination MAC address. A bridged interface automatically sends multicast and broadcast bridged traffic to the other interface of the bridge group.
- ARP packets are always passed through an Layer 2 interface after their verification and inspection. Multicast and broadcast packets from the incoming interface are flooded to the other L2 interface in the bridge group.
- The server default gateway is the upstream router.
- By default, the ACE performs a route lookup to select the next hop to reach the client. We recommend using the mac-sticky feature, rather than the static default route, to send return traffic back in response to the client connection.

Configuring Bridged Mode on the ACE

This section describes how to configure bridged mode using the example shown in Figure 13-1.

Figure 13-1 Example of Bridged Mode



The configuration of the example setup is as follows:

- A virtual server VS_WEB2 is created with a virtual IP address 10.15.3.100 to forward the client traffic from VLAN 40 to the servers in VLAN 41.
- There are four real servers grouped into the server farm SF_WEB2.
- VLAN 40 is assigned to the ACE and is used for client-side traffic. VLAN 41 is assigned to the ACE and is used for server-side traffic.
- A BVI with the IP address 10.15.3.5 configures the two VLANs into one bridge group.

This section contains the following topics:

- Prerequisites
- Configuring Server Load Balancing
- Configuring the VLANs and a BVI

Task Flow for Configuring Bridged Mode

Follow these steps to configure bridged mode on the ACE:

Step 1 Configure the real servers and server farm.

Step 2 Configure a TCP probe and associate it with the server farm.

Step 3 Configure the VIP address where clients are to send requests.

Step 4 Create the policy for load-balancing traffic.

Step 5 Create a service policy.

Step 6 Create the client and server VLANs and associate them with a BVI.

Step 7 Configure the mac-sticky feature on the client VLAN interface.

Step 8 Apply the access group and service policy to the interface.

Configuring Server Load Balancing

Procedure

Step 1 Add the four real servers (see the "Configuring Real Servers" section in Chapter 6, Configuring Server Load Balancing), using the following real server names, descriptions, and IP addresses and place each server in service:

- Name: RS_WEB5, Description: content server web-five, IP Address: 10.15.3.11
- Name: RS_WEB6, Description: content server web-six, IP Address: 10.15.3.12
- Name: RS_WEB7, Description: content server web-seven, IP Address: 10.15.3.13
- Name: RS_WEB8, Description: content server web-eight, IP Address: 10.15.3.14

Step 2 Group these real servers into a server farm (see the "Creating a Server Farm" section in Chapter 6, Configuring Server Load Balancing) and place each server in service. In this example, name the server farm SF_WEB2.

Step 3 Configure a TCP probe to check the health of all the real servers in the server farm and associate the probe with the server farm. See the "Configuration Example for Bridged Mode" section.

Step 4 Create a virtual server traffic policy (see "Creating a Virtual Server Traffic Policy" section, in Chapter 6, Configuring Server Load Balancing, Steps 1 through 12). For this example, do the following:

- Create a Layer 7 policy map for the action when the client request arrives and is sent to the server farm, name the load-balancing policy HTTP_LB, configure a default class map, and associate the server farm SF_WEB2.
- Create a Layer 3 and Layer 4 class map to define the VIP where the clients will send their requests, and name the class map VS_WEB2 with a match virtual address of 10.15.3.100 with a match on any port.
- Create a Layer 3 and Layer 4 multi-match policy map to direct classified incoming requests to the load-balancing policy map. In this example, name the policy HTTP_MULTI_MATCH, associate the VS_WEB2 class map and the HTTP_LB policy map. and then enable the VIP for load-balancing operations by placing it in service.

Configuring the VLANs and a BVI

You can configure bridged mode by creating the client-side a nd the server side VLANs on the ACE and associating them with a BVI.

Procedure

	Command	Purpose
Step 1	**changeto** *context* **Example:** host1/Admin# changeto VC_WEB host1/VC_WEB#	Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.
Step 2	**config** **Example:** host1/VC_WEB# config host1/VC_WEB(config)#	Enters configuration mode.
Step 3	**interface vlan** *vlan_id* **Example:** host1/VC_WEB(config)# interface vlan 40 host1/VC_WEB(config-if)#	Accesses the interface for the client-side VLAN.
Step 4	**description** *string* **Example:** host1/VC_WEB(config-if)# description Client_side	Enters a description of the VLAN.
Step 5	**bridge-group** *number* **Example:** host1/VC_WEB(config-if)# bridge-group 1	Assigns the VLAN to the BVI.
Step 6	**mac-sticky** enable **Example**: host1/VC_WEB(config-if)# mac-sticky enable	Enables the mac-sticky feature for a VLAN interface.
Step 7	**access-group input** *acl_name* **Example:** host1/VC_WEB(config-if)# access-group input INBOUND	Applies the ACL to the VLAN.
Step 8	**service-policy input** *policy_name* **Example:** host1/VC_WEB(config-if)# service-policy input HTTP_MULTI_MATCH	Applies the multi-match policy map to the VLAN.
Step 9	**no shutdown** **Example:** host1/VC_WEB(config-if)# no shutdown	Places the VLAN in service.

	Command	Purpose

Step 10	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)#`
		Exits interface configuration mode.

Step 11	**interface vlan** *vlan_id*
	
	Example:
	`host1/VC_WEB(config)# interface vlan 41`
	`host1/VC_WEB(config-if)#`
		Accesses the interface for the server-side VLAN.

Step 12	**description** *string* **Example:** `host1/VC_WEB(config-if)# description Server_side`	Enters a description of the VLAN.
Step 13	**bridge-group** *number* **Example:** `host1/VC_WEB(config-if)# bridge-group 1`	Assigns the VLAN to the BVI.
Step 14	**no shutdown** **Example:** `host1/VC_WEB(config-if)# no shutdown`	Places the VLAN in service.

Step 15	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)#`
		Exits interface configuration mode.

Step 16	**interface bvi** *number*
	
	Example:
	`host1/VC_WEB(config)# interface bvi 1`
	`host1/VC_WEB(config-if)#`
		Creates the BVI.

Step 17	**description** *string* **Example:** `host1/VC_WEB(config-if)# description Client and server bridge group 1`	Enters a description of the BVI.
Step 18	**ip address** *ip_address netmask* **Example:** `host1/VC_WEB(config-if)# ip address 10.15.3.5 255.255.255.0`	Assigns an IP address and network mask to the BVI interface.
Step 19	**no shutdown** **Example:** `host1/VC_WEB(config-if)# no shutdown`	Places the BVI in service.

Step 20	**Ctrl-Z**
	
	Example:
	`host1/Admin(config-if)# Ctrl-Z`
	`host1/Admin#`
		Returns to Exec mode directly from any configuration mode.

	Command	Purpose
Step 21	**show running-config interface** **Example:** host1/Admin# show running-config interface	Displays the interface configuration.
Step 22	**show interface bvi** *number* **Example:** host1/Admin# show interface bvi 1	Displays the status and statistics for the BVI interface.
Step 23	**copy running-config startup-config** **Example:** host1/Admin# copy running-config startup-config	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for Bridged Mode

The following running configuration example shows a basic bridged mode configuration. The commands that you have configured in this chapter appear in bold text.

```
access-list INBOUND extended permit ip any

probe tcp TCP_PROBE1

rserver host RS_WEB5
    description content server web-five
    ip address 10.15.3.11
    inservice
rserver host RS_WEB6
    description content server web-six
    ip address 10.15.3.12
    inservice
rserver host RS_WEB7
    description content server web-seven
    ip address 10.15.3.13
    inservice
rserver host RS_WEB8
    description content server web-eight
    ip address 10.15.3.14
    inservice
serverfarm SF_WEB2
    probe TCP_PROBE1
    rserver RS_WEB5 80
        inservice
    rserver RS_WEB6 80
        inservice
    rserver RS_WEB7 80
        inservice
    rserver RS_WEB8 80
        inservice

policy-map type loadbalance first-match HTTP_LB
    class class-default
    serverfarm SF_WEB2

class-map VS_WEB2
    match virtual-address 10.15.3.100 any

policy-map multi-match HTTP_MULTI_MATCH
```

```
    class VS_WEB2
        loadbalance policy HTTP_LB
        loadbalance vip inservice

interface bvi 1
    description Client and server bridge group 1
    ip address 10.15.3.5 255.255.255.0
    no shutdown

interface vlan 40
    description Client_side
    bridge-group 1
    mac-sticky enable
    access-group input INBOUND
    service-policy input HTTP_MULTI_MATCH
    no shutdown

interface vlan 41
    description Server-side
    bridge-group 1
    no shutdown

context VC_WEB
    allocate-interface vlan 40
    allocate-interface vlan 41
    member RC_WEB

ip route 0.0.0.0 0.0.0.0 10.15.3.1
```

Where to Go Next

In this chapter, you have learned how to configure bridged mode on your ACE. For more detailed information about both bridged mode and routed mode, see the *Routing and Bridging Guide, Cisco ACE Application Control Engine*.

In the next chapter, you will learn how to configure your ACE for “one-arm” mode.

CHAPTER 14

Configuring One-Arm Mode

This chapter describes how to configure the Cisco Application Control Engine (ACE) module to receive requests from clients and send them to servers on the same VLAN.

This chapter includes the following sections:

- Information About One-Arm Mode
- Guidelines and Limitations
- Task Flow for Configuring One-Arm Mode
- Configuring One-Arm Mode on the ACE
- Configuration Example for One-Arm Mode
- Where to Go Next

Information About One-Arm Mode

After reading this chapter, you should have a basic understanding of one-arm mode, how it works in the ACE, and how to configure it.

In one-arm mode, you configure the ACE with a single VLAN that handles both client requests and server responses. For one-arm mode, you must configure the ACE with client-source network address translation (NAT) or policy-based routing (PBR) to send requests through the same VLAN to the server. For the remainder of this document, NAT is used for the traffic flows through the ACE.

The ACE is not inline with the traffic and receives and sends requests through the Multilayer Switching Feature card (MSFC) that acts as a default gateway to the servers. The MSFC routes requests to a VIP that is configured on the ACE. When the ACE selects the server for the request based on the configured policy, it rewrites the source IP address with an address in the NAT pool. Then the ACE forwards the request to the server on the same VLAN through the default gateway on the MSFC.

The server sends a response to the default server gateway on the MSFC. The server response contains its source IP address and the NAT address of the ACE as the destination IP address. The MSFC forwards the response to the ACE. The ACE receives the response, changes the source IP address to the VIP, and sends it to the MFSC. Then the MFSC forwards the response to the client.

Guidelines and Limitations

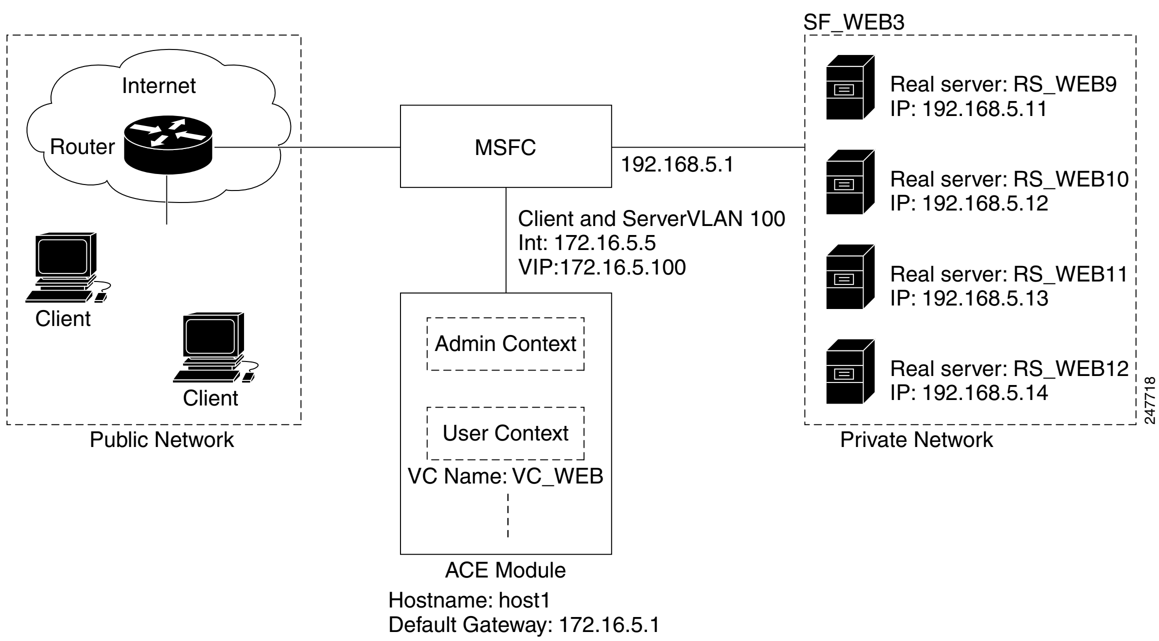
One-arm mode on the ACE has the following configuration guidelines and limitations:

- Layer 2 rewrite is not supported.
- One-arm mode requires policy-based routing or source NAT.

Configuring One-Arm Mode on the ACE

This section describes how to configure one-arm mode using the example shown in Figure 14-1.

Figure 14-1 Example Network Setup



The configuration of the example is as follows:

- A client and server VLAN interface is configured for the user context VC_WEB with VLAN 100.
- A virtual server VS_WEB3 is created with a virtual IP (VIP) address 172.16.5.100 where the clients send requests.
- There are four real servers grouped into the server farm SF_WEB3.
- The IP address 192.168.5.1 is the gateway for the real servers.

This section contains the following topics:

- Prerequisites for One-Arm Mode on the ACE
- Configuring Server Load Balancing and Source NAT
- Configuring the One-Arm VLAN

Prerequisites for One-Arm Mode on the ACE

One-arm mode on an ACE has the following prerequisites:

- An available VLAN for both clients and servers. Find out what VLANs and addresses are available for use by the ACE.
- A default route on the ACE (see the "Configuring a Default Route" section in Chapter 2, Setting Up an ACE).
- An access list to allow traffic to the ACE (see the "Configuring an ACL" section in Chapter 4, Configuring Access Control Lists).

Task Flow for Configuring One-Arm Mode

Follow these steps to configure one-arm mode on the ACE:

Step 1 Configure the real servers and a server farm.

Step 2 Configure a TCP probe and associate it with the server farm.

Step 3 Create a virtual server policy to load balance client requests.

Step 4 Configure the client and server VLAN.

Step 5 Apply the access group for the ACL, the virtual server policy, and the NAT pool to the VLAN.

Configuring Server Load Balancing and Source NAT

Procedure

Step 1 Add the four real servers (see the "Configuring Real Servers" section in Chapter 6, Configuring Server Load Balancing), using the following real server names, descriptions, and IP addresses and place each server in service for use:

- Name: RS_WEB9, Description: content server web-nine, IP Address: 192.168.5.11
- Name: RS_WEB10, Description: content server web-ten, IP Address: 192.168.5.12
- Name: RS_WEB11, Description: content server web-eleven, IP Address: 192.168.5.13
- Name: RS_WEB12, Description: content server web-twelve, IP Address: 192.168.5.14

Step 2 Group these real servers into a server farm (see the "Creating a Server Farm" section in Chapter 6, Configuring Server Load Balancing) and place each server in service. In this example, name the server farm SF_WEB3.

Step 3 Configure a TCP probe and associate it with the server farm. See the "Configuration Example for One-Arm Mode" section.

Step 4 Create a virtual server traffic policy (see Steps 1 through 12 in the "Creating a Virtual Server Traffic Policy" section, in Chapter 6, Configuring Server Load Balancing). For this example, you create the following configuration objects:

- The policy map for the action when the client request arrives and is sent to the server farm. In this example, name the load-balancing policy PM_ONE_ARM_LB, configure a default class map, and associate the server farm SF_WEB3.
- The class map to define the VIP where the clients will send their requests. In this example, name the class map VS_WEB3 with a match virtual address of 172.16.5.100 with a match on any port.
- A multi-match service policy map to direct classified incoming requests to the load-balancing policy map. In this example, you do the following:
  - Name the policy PM_ONE_ARM_MULTI_MATCH.
  - Associate the VS_WEB3 class map and the PM_ONE_ARM_LB policy map.
  - Configure the **nat dynamic 5 vlan 100** command to allow the ACE to source NAT all client requests. The 5 indicates the NAT pool ID as configured in VLAN 100 (see "Configuring the One-Arm VLAN" section.
  - Enable the VIP for load-balancing operations by placing it in service.

Configuring the One-Arm VLAN

You can configure the one-arm mode VLAN on the ACE with a NAT pool.

Procedure

	Command	Purpose

Step 1	**changeto** *context*
	
	Example:
	`host1/Admin# changeto VC_WEB`
	`host1/VC_WEB#`
		Changes to the correct context if necessary. Check the CLI prompt to verify that you are operating in the desired context.

Step 2	**config**
	
	Example:
	`host1/VC_WEB# config`
	`host1/VC_WEB(config)#`
		Enters configuration mode.

Step 3	**interface vlan** *vlan_id*
	
	Example:
	`host1/VC_WEB(config)# interface vlan 100`
	`host1/VC_WEB(config-if)#`
		Accesses the interface for the client-side VLAN.

Step 4	**description** *string* **Example:** `host1/VC_WEB(config-if)# description Client and server VLAN`	Enters a description of the VLAN.

	Command	Purpose
Step 5	**ip address** *address subnet_mask* **Example:** `host1/VC_WEB(config-if)# ip address 172.16.5.5 255.255.255.0`	Assigns the IP address to the VLAN.
Step 6	**access-group input** *acl_name* **Example:** `host1/VC_WEB(config-if)# access-group input INBOUND`	Applies the ACL to the interface.
Step 7	**service-policy input** *policy_name* **Example:** `host1/VC_WEB(config-if)# service-policy input PM_ONE_ARM_MULTI_MATCH`	Applies the multi-match policy map to the VLAN.

	Command	Purpose
Step 8	**nat-pool** *pool_id ip_address1 ip_address2* **netmask** *mask* [**pat**] **Example:** `host1/VC_WEB(config-if)# nat-pool 5 172.16.5.200 172.5.16.209 netmask 255.255.255.0 pat`	Creates a pool of IP addresses for dynamic NAT: • *pool_id*—Identifier of the NAT pool of global IP addresses. Enter an integer from 1 to 2147483647. **Note** If you configure more than one NAT pool with the same ID, the ACE uses the last-configured NAT pool first and then the other NAT pools. • *ip_address1*—Single IP address, or if also using the *ip_address2* argument, the first IP address in a range of global addresses used for NAT. Enter an IP address in dotted-decimal notation (for example, 172.27.5.200). • *ip_address2*—Highest IP address in a range of global IP addresses used for NAT. Enter an IP address in dotted-decimal notation (for example, 172.27.5.209). You can configure a maximum of 65,535 addresses in a NAT pool. **Note** You cannot configure an IP address range across subnets. For example, the following command is not allowed and will generate an Invalid IP address error: nat-pool 2 10.0.6.1 10.0.7.20 netmask 255.255.255.0. • **netmask** *mask*—Specifies the subnet mask for the IP address pool. Enter a mask in dotted-decimal notation (for example, 255.255.255.255). A network mask of 255.255.255.255 instructs the ACE to use all the IP addresses in the specified range. • **pat**—Enables port address translation. The **pat** option instructs the ACE to translate port numbers and IP addresses. If you omit the **pat** option, the ACE will be limited to the number of IP addresses in the pool for the number of concurrent NAT connections
Step 9	**no shutdown** **Example:** `host1/VC_WEB(config-if)# no shutdown`	Places the VLAN in service.

	Command	Purpose

Step 10	**exit**
	
	Example:
	`host1/VC_WEB(config-if)# exit`
	`host1/VC_WEB(config)#`
		Exits interface configuration mode.

Step 11	**Ctrl+Z**
	
	Example:
	`host1/Admin(config-if)# Ctrl+Z`
	`host1/Admin#`
		Returns to Exec mode directly from any configuration mode.

Step 12	**show running-config interface** **Example:** `host1/Admin# show running-config interface`	Displays the interface configuration.
Step 13	**show interface vlan** *number* **Example:** `host1/Admin# show interface vlan 100`	Displays the status and statistics about the VLAN interface.
Step 14	**copy running-config startup-config** **Example:** `host1/Admin# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuration Example for One-Arm Mode

The following example shows how to configure one-arm mode.

```
access-list INBOUND extended permit ip any any

probe tcp TCP_PROBE2

rserver host RS_WEB9
  description content server web-nine
  ip address 192.168.5.11
  inservice
rserver host RS_WEB10
  description content server web-ten
  ip address 192.168.5.12
  inservice
rserver host RS_WEB11
  description content server web-eleven
  ip address 192.168.5.13
  inservice
rserver host RS_WEB12
  description content server web-twelve
  ip address 192.168.5.14
  inservice

serverfarm SF_WEB3
  probe TCP_PROBE2
  rserver RS_WEB9 80
    inservice
  rserver RS_WEB10 80
    inservice
  rserver RS_WEB11 80
    inservice
```

```
  rserver RS_WEB12 80
    inservice

policy-map type loadbalance first-match PM_ONE_ARM_LB
  class class-default
  serverfarm SF_WEB3

class-map VS_WEB3
  match virtual-address 172.16.5.100 any

policy-map multi-match PM_ONE_ARM_MULTI_MATCH
  class VS_WEB3
    loadbalance policy PM_ONE_ARM_LB
    nat dynamic 5 vlan 100
  loadbalance vip inservice

interface vlan 100
  description Client_server
  ip address 172.16.5.5 255.255.255.0
  access-group input INBOUND
  service-policy input PM_ONE_ARM_MULTI_MATCH
  nat-pool 5 172.16.5.200 172.16.5.209 netmask 255.255.255.0 pat
  no shutdown

context VC_WEB
  allocate-interface vlan 100
  member RC_WEB

ip route 0.0.0.0 0.0.0.0 172.16.5.1
```

Where to Go Next

In this chapter, you have learned how to configure one-arm mode.

This chapter concludes the ACE quick start guide. In this guide, you have learned how to configure the basics of many ACE features.

- For more advanced ACE features and functionality, see the configuration guides in the ACE documentation set at the following URL:

  http://www.cisco.com/en/US/products/ps6906/tsd_products_support_model_home.html

- For ease in locating features and topics of interest, see the master index in the configuration guide list.
- For command-specific information, see the *Command Reference, Cisco ACE Application Control Engine*.
- For troubleshooting information, see the ACE Troubleshooting Wiki at the following URL:

  http://docwiki.cisco.com/wiki/Cisco_Application_Control_Engine_%28ACE%29_Troubleshooting_Guide

- For configuration examples, see the ACE Configuration Examples Wiki at the following URL:

  http://docwiki.cisco.com/wiki/Cisco_Application_Control_Engine_%28ACE%29_Configuration_Examples

INDEX

T

U

V