Advanced Server Load Balancing Deployment Guide

SBA FOR GOVERNMENT

Revision: H2CY10

The Purpose of this Guide

This guide is a concise reference on server load balancing.

This guide introduces the Cisco Application Control Engine (ACE, or ACE4710), the latest server load balancing offering from Cisco.

It explains the requirements that were considered when building the Cisco Smart Business Architecture (SBA) for Government design and introduces each of the products that were selected.

The final section of this guide will present the actual deployment steps that will get the product deployed and working in a specific environment.

Who Should Read This Guide

This guide is intended for the reader who has any or all of the following:

- Multiple application servers
- Hosts their own application servers, either locally or co-located
- IT workers with a CCNA® certification or equivalent experience
- Is looking to deploy server load balancing
- Has read the *Data Center Deployment Guide* and wants more advanced features than are shown in the resilient server module

The reader may be looking for any or all of the following:

- High availability for applications
- Scaling an application across multiple servers
- The assurance of a tested solution

Related Documents

Before reading this guide

DC — Data Center Design Overview

DC — Data Center Deployment Guide

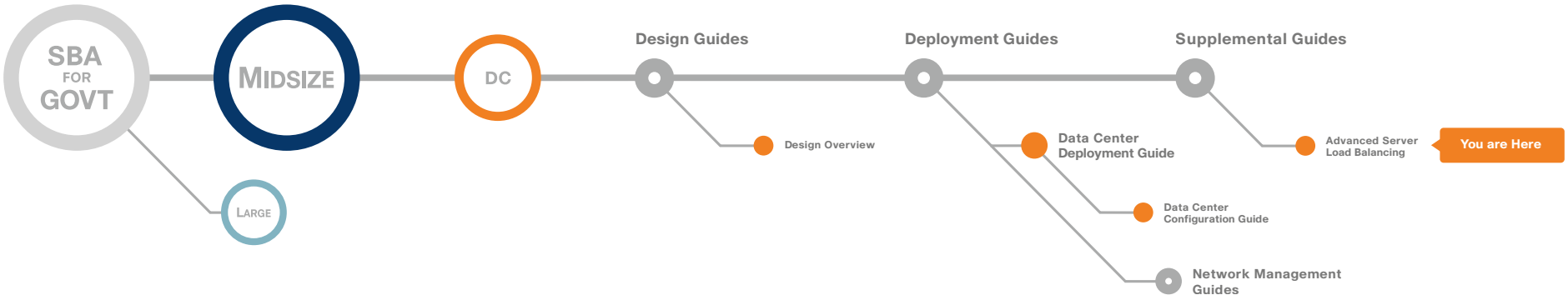


Table of Contents

Introduction

The Cisco® SBA is a comprehensive design for networks with up to 1000 users. This out-of-the-box design is simple, fast, affordable, scalable, and flexible.

The Cisco SBA for Midsize Agencies incorporates LAN, WAN, wireless, security, WAN optimization, and unified communication technologies tested together as a solution. This solution-level approach simplifies the system integration normally associated with multiple technologies, allowing you to select the modules that solve your agency's problems rather than worrying about the technical details.

We have designed the Cisco SBA to be easy to configure, deploy, and manage. This architecture:

- Provides a solid network foundation
- Makes deployment fast and easy
- Accelerates ability to easily deploy additional services
- Avoids the need for re-engineering of the core network

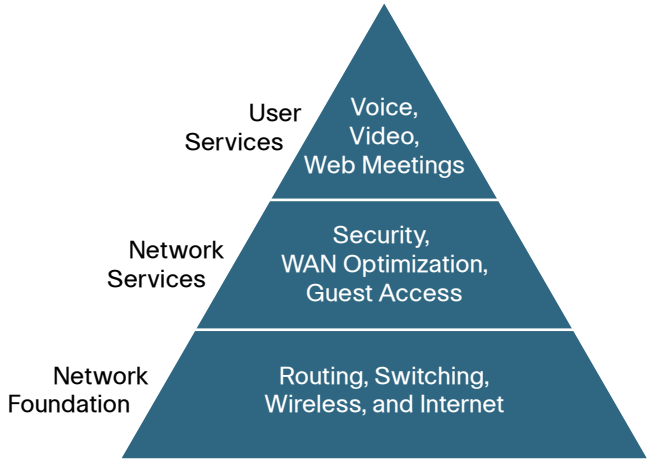
By deploying the Cisco SBA, your agency can gain:

- A standardized design, tested and supported by Cisco
- Optimized architecture for midsize agencies with up to 1000 users and up to 20 branches
- Flexible architecture to help ensure easy migration as the agency grows
- Seamless support for quick deployment of wired and wireless network access for data, voice, teleworker, and wireless guest
- Security and high availability for agency information resources, servers, and Internet-facing applications
- Improved WAN performance and cost reduction through the use of WAN optimization
- Simplified deployment and operation by IT workers with CCNA certification or equivalent experience
- Cisco enterprise-class reliability in products designed for midsize agencies

Guiding Principles

We divided the deployment process into modules according to the following principles:

- **Ease of use:** A top requirement of Cisco SBA was to develop a design that could be deployed with the minimal amount of configuration and day-two management.
- **Cost-effective:** Another critical requirement as we selected products was to meet the budget guidelines for midsize agencies.
- **Flexibility and scalability:** As the agency grows, so too must its infrastructure. Products selected must have the ability to grow or be repurposed within the architecture.
- **Reuse:** We strived, when possible, to reuse the same products throughout the various modules to minimize the number of products required for spares.



The Cisco SBA can be broken down into the following three primary, modular yet interdependent components for the midsize agency.

- **Network Foundation:** A network that supports the architecture
- **Network Services:** Features that operate in the background to improve and enable the user experience without direct user awareness
- **User Services:** Applications with which a user interacts directly

Agency Overview

The network is playing an increasingly important role in the success of an agency. Key applications such as Enterprise Resource Planning (ERP), e-commerce, email, and portals must be available around the clock to provide uninterrupted agency services. However, the availability of these applications is often threatened by network overloads as well as server and application failures. Furthermore, resource utilization is often out of balance, resulting in the low-performance resources being overloaded with requests, while the high-performance resources remain idle. This is evidence that application performance, as well as availability, directly affects employee productivity and the bottom line of an agency. As more users work more hours utilizing key agency applications, it becomes even more important to address application availability and performance issues to ensure achievement of agency processes and objectives.

Many factors make applications difficult to deploy and deliver effectively over the network, including:

- **Inflexible Application Infrastructure:** Application design has historically been done on an application-by-application basis. This means that the infrastructure used for a particular application is often unique to that application. This type of design tightly couples the application to the infrastructure and offers little flexibility. Because the application and infrastructure are tightly coupled, it is difficult to partition resources and levels of control to match changing agency requirements.
- **Server Availability and Load:** The mission-critical nature of applications puts a premium on server availability. Despite the benefits of server virtualization technology, the number of physical servers continues to grow based on new application deployments, raising power, and cooling requirements.
- **Application Security and Compliance:** Many of the new security events are the result of application- and document-embedded attacks that compromise application performance and availability. Such attacks also potentially cause loss of vital application data—even while leaving networks and servers unaffected.

One way to improve application performance and availability is to rewrite the application completely so it is network-optimized. However, this requires application developers to have a much deeper understanding of how different applications respond to things like bandwidth constraints, delay, jitter, and other network variances. In addition, developers need a clearly predictable view of an end user's foreseeable access method. This is simply not feasible for every agency application—particularly legacy applications that took years to write and customize.

Improvements to application performance begin in the data center. The Internet boom ushered in the era of the server load balancers, which balance the load on server banks to improve their response to client requests. Server load balancers have also evolved to take on additional responsibilities such as application proxies and complete Layer 4 through 7 application switching.

Technology Overview

The Application Control Engine (ACE, or ACE 4710) is the latest server load balancing (SLB) offering from Cisco. From its mainstream role in providing Layer 4 through 7 switching, Cisco ACE also provides an array of acceleration and server offload benefits, including:

- Transmission Control Protocol (TCP) processing offload
- Secure Socket Layer (SSL) offload
- Compression
- Various other acceleration technologies

Cisco ACE sits within the data center in front of the Web and application servers and provides services to maximize server and application availability, security, and asymmetric (from server to client browser) application acceleration. As a result, Cisco ACE gives IT departments more control over application and server infrastructure, which enables them to manage and secure application services more easily and improves performance.

There are several ways to integrate ACE within the data center network as shown in Figure SLBB-1. Logically, the ACE is deployed in front of the Web application cluster. Requests to the application cluster are directed to a virtual IP address (VIP) configured on the ACE. The ACE receives connections and Hypertext Transfer Protocol (HTTP) requests and routes them to the appropriate application server based on configured policies as shown in Figure 2.

There are four key benefits provided by Cisco ACE:

- **Scalability:** ACE scales the performance of a server-based program, such as a Web server, by distributing its client requests across multiple servers, known as a server farm. As traffic increases, additional servers can be added to the farm. With the advent of server virtualization, applications can be staged and added dynamically as capacity requirements change.
- **High Availability (HA):** ACE provides high availability by automatically detecting the failure of a server and repartitioning client traffic among the remaining servers within seconds, while providing users with continuous service.
- **Application Acceleration:** ACE improves application performance and reduces response time by minimizing latency and data transfers for any HTTP-based application, for any internal or external end user.
- **Server Offload:** ACE offloads TCP and SSL processing from the servers, allowing servers to serve more users and handle more requests without increasing the number of servers.

Figure 1. SLB Overview

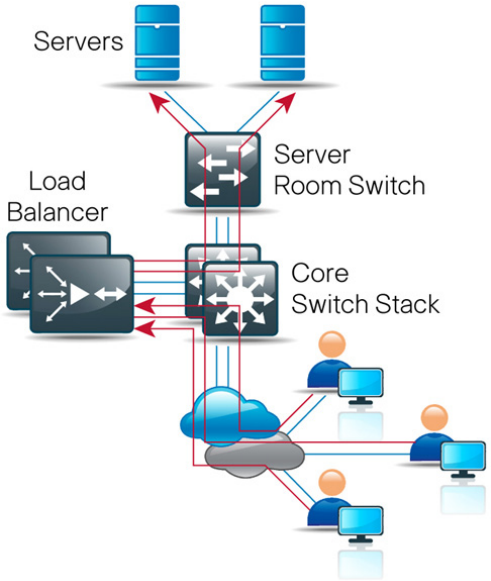
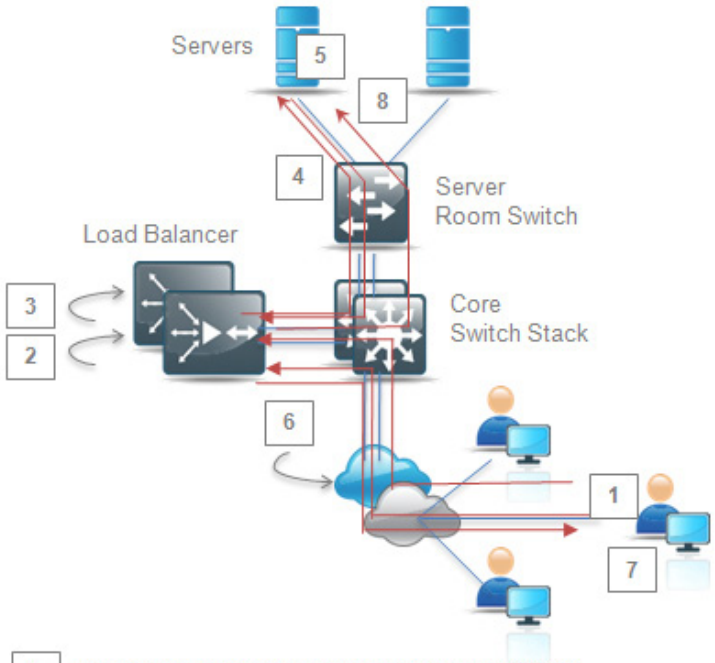


Figure 2. Typical Load Balancing Traffic Flow



1 The client request reaches the load balancer (SYN)

2 The client request is processed and a server farm is selected

3 Within the server farm, a server is selected

4 The load balancer send the packet (SYN) to the server

5 The server responds back to the client (SYN ACK) through the load balancer

6 The load balancer sends the packet toward the client (SYN ACK)

7 The client completes the TCP handshake (ACK)

8 The load balancer establishes the flow and sends the packet (ACK) toward the server.

9 TCP session is established

Physical Topologies

Physically, the network topology can take many forms, including:

- One Armed Mode
- Routed Mode
- Single Virtual Local Area Network (VLAN) One-Armed Mode

One-Armed Mode

One-Armed Mode is the simplest deployment method, where the ACE is connected off to the side of the Layer 2/Layer 3 infrastructure. It is not directly in the path of traffic flow and only receives traffic that is specifically intended for it. Traffic that should be directed to it is controlled by careful design of virtual LANs (VLANs), virtual server addresses, server default gateway selection, or policy routes on the Layer 3 switch or upstream router.

Routed Mode

In Routed Mode, seen in Figure 3, is the most commonly deployed method. The load balancer acts as a Layer 3 device. It routes traffic flows between clients and servers. In this mode, the real server's default gateway is the load balancer.

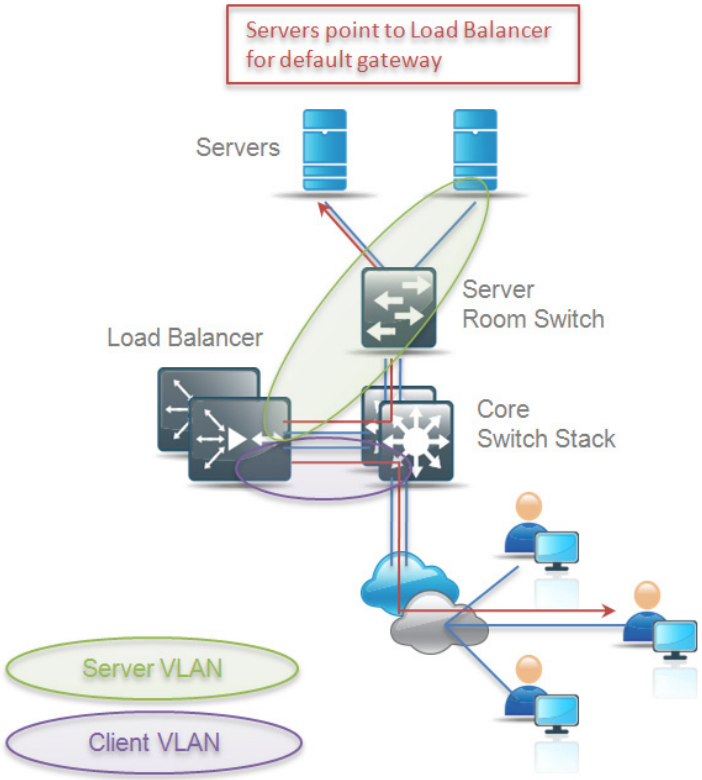
Pros:

- Simple topology, ease of configuration on the ACE.

Cons:

- ACE does not support any dynamic routing protocols. Static routes only, leading to overhead.
- All server traffic must pass through ACE, whether or not load-balancing is required.

Figure 3. Routed Mode



Single-VLAN One-Armed Mode

In Single-VLAN One-Armed Mode, seen in Figure 4, the load balancer resides on the same network as the real servers and clients. In this mode, the real server's default gateway is the upstream router. To ensure the return flow traverses back through the load balancer, the IP address of the client is rewritten to that of the load balancer.

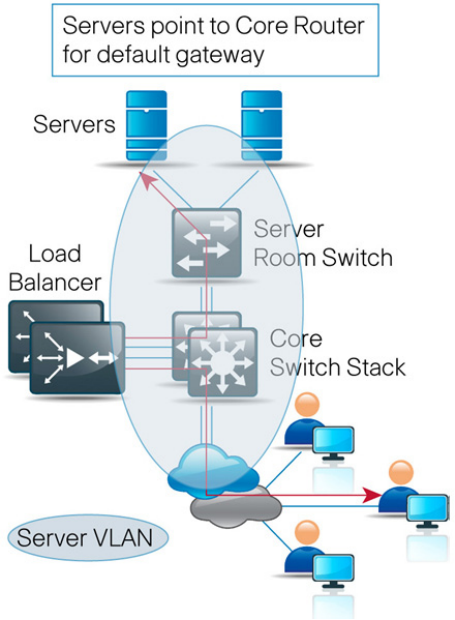
Pros:

- Layer 2 adjacency with the real servers is not required.
- Able to preserve client source IP address.
- Allows direct server traffic to bypass ACE when load-balancing is not required.

Cons:

- Client source IP address is masked by ACE due to Source NAT. All servers see ACE as the client, resulting in loss of visibility of original client IP address.
- Requires HTTP header insert as a workaround to preserve client source IP address.
- Not suitable for non-HTTP-based applications that require source IP address preservation.

Figure 4. Single-VLAN One-Armed Mode



ACE Overview

ACE hardware is always deployed in pairs for highest availability: One primary and one secondary. If the primary ACE fails, the secondary ACE takes control. Depending on the configuration of session state redundancy, this failover may take place without disrupting the client-to-server connection.

Cisco ACE uses both active and passive techniques to monitor server health. By periodically probing servers, the ACE will rapidly detect server failures and quickly reroute connections to available servers. A variety of health-checking features are supported, including the ability to verify Web servers, SSL servers, application servers, databases, File Transfer Protocol (FTP) servers, streaming media servers, and a host of others.

Cisco ACE can be used to partition components of a single Web application across several application server clusters. For example: The two URLs www.mycompany.com/quotes/getquote.jsp and www.mycompany.com/trades/order.jsp could be located on two different server clusters even though the domain name is the same. This allows the application developer to easily scale the application to several servers without numerous code modifications. Furthermore, it maximizes the cache coherency of the servers by keeping requests for the same pages on the same servers.

Additionally, ACE may be used to push requests for cacheable content such as image files to a set of caches that can serve them more cost-effectively than the application servers.

Running SSL on the Web application servers is a tremendous drain on server resources. By offloading SSL processing, those resources can be applied to traditional Web application functions. In addition, because persistence information used by the content switches is inside the HTTP header, this information is no longer visible when carried inside SSL sessions. By terminating these sessions before applying content switching decisions, all the persistence options previously discussed become available for secure sites.

ACE reduces the amount of data sent from the Web application server to the browser by utilizing hardware compression and patented Delta Encoding. Delta Encoding determines exactly what has changed from page to page, to the level of detail of a single byte, and sends only the content that has changed.

ACE further improves the end-user application experience by reducing latency and the number of roundtrips required for application access. ACE eliminates unnecessary browser cache validation requests and provides automatic embedded object version management at the server, resulting in significantly improved application response times for application users.

Virtualization

Virtual contexts are separate logical partitions that essentially turn an ACE appliance into multiple virtual instances that can be independently configured in terms of topology, resource usage, and functional usage.

Virtual contexts should be created on an as-needed basis, using the following guidelines:

- For Application Teams that require frequent login access to the ACE to configure and/or fine-tune parameters, or to take real servers in and out of service, a separate context should be created on the basis of one context per Application Team to allow administrative segmentation. Each context can be tied to RBAC, with the appropriate users and roles assigned to each.
- In most cases, there will be a dedicated set of application delivery environments—Dev, Stage, Prod. If, however, this physical separation does not exist, then contexts should be created to segregate the different types of environments from each other.
- To support virtual contexts, RBAC needs to be enabled on ACE, with appropriate roles and domains defined and assigned to each user account.
- All Application Owner access should be limited to their specific virtual context. They should not be allowed to view objects other than the ones permitted in their own virtual context. Cisco documentation on how to configure RBAC in ACE can be found online at http://www.cisco.com/en/US/docs/app_ntwk_services/data_center_app_services/ace_appliances/vA3_1_0/configuration/quick/guide/rbac.html

Hardware HA and Virtual Contexts

ACE hardware is always deployed in pairs: One primary appliance and one secondary appliance. It supports high availability using redundant Fault-Tolerant (FT) groups that are configured based on virtual contexts. Multiple FT groups can be configured per ACE pair, but only one FT group can be associated with any virtual context pair. Two instances of the same context form a redundancy group, one is “Active” and the other “Standby”.

A Fault-Tolerant (FT) VLAN is a dedicated VLAN used by a redundant ACE pair to communicate heartbeat and state information. All redundancy-related traffic is sent over this FT VLAN (including Trusted Relay Point (TRP) protocol packets, heartbeats, configuration sync packets, and state replication packets). Heartbeat packets are sent over UDP via the FT VLAN between peer units and are used to monitor the health of the peer device. The FT VLAN also carries state information transmitted between the two ACE peers in order to maintain sessions and stickiness in the event of failover.

HA virtualization is modeled as follows:

- In normal operational state: All contexts are “Active” on the primary ACE appliance, and all contexts are “Standby” on the secondary appliance.
- In a transient failure state: Some contexts are “Active” in one ACE appliance (A) and “Standby” in the other appliance (B), and other contexts are “Active” in (B) and “Standby” in (A).
- Each context can fail over independently. However, at any given time, each context and each VIP is active only on one single physical ACE.
- Preempt with a higher priority should be configured in all virtual contexts on the primary ACE appliance to force mastership after the primary appliance recovers from failure.

Load-Balancing

ACE traffic policies support the following SLB traffic attributes:

- **Layer 3 and Layer 4 connection information:** Source or destination Internet Protocol (IP) address, source or destination port, virtual IP address, and IP protocol. The ACE uses the Layer 3 and Layer 4 traffic classes to perform server load-balancing. For a Layer 3 and Layer 4 traffic classification, the match criteria in a class map include the VIP address, protocol, and port of the ACE.
- **Layer 7 protocol information:** HTTP cookie, HTTP URL, HTTP header, and SSL. The layer 7 SLB will have the same configuration logic of traffic class map and policy map, but its class map contains match criteria that classify specific Layer 7 network traffic. It is based on HTTP cookies, HTTP headers, HTTP URLs, or SSL ciphers.

The Load-Balancer gathers parse results from HTTP until HTTP is done parsing the header. Using the parse results, ACE determines the best policy match for load balancing.

Probe Functions

ACE lets you continually monitor the server's health and availability. It uses probes as one of the available keep-alive methods to verify the availability of a real server. A probe can be attached to a real server, server farm, or a gateway. It can take a real server out of service for the following reasons:

- Probe failure
- ARP timeout
- No inservice command
- Inservice standby command

Figure 5. Probe Functions

Probe Service	Function
Dns	Uses a default domain of "www.cisco.com"
echo	Configure echo probe
finger	Configure finger probe
ftp	Open a FTP connection with server and disconnect
http	Sends a "GET / HTTP 1.1" request
https	Establishes an SSL connection, send HTTP query and tears it down
icmp	Configure icmp probe
imap	Open an imap session and disconnect
ldap	Configure ldap probe
pop3	Open a pop session and disconnect
radius	Open an authentication session and disconnect
scripted	Uses TCL Interpreter to execute user defined TCL scripts and perform health monitoring
smtp	Sends a "hello" followed by a "QUIT" message
tcp	Open a TCP session with server and disconnect with TCP FIN
telnet	Makes a Telnet connection, send a "QUIT" message
udp	Sends a UDP packet, probe is considered successful if no icmp error is received

In addition to the default probe shown in Figure 5, ACE supports the usage of custom probes written in Tool Command Language (TCL) up to 256 script files. If the standard probes do not meet the requirements, application owners can provide a TCL probe for health monitoring.

Session Persistence

Session Persistence (or stickiness) is an ACE feature that allows the same client to maintain multiple simultaneous or subsequent TCP or IP connections with the same real server for the duration of a session. Load balancer accesses the sticky database before making a destination decision in case the connection already exists. 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 specified by the matched policy.

Sticky code supports replication of the Sticky database for HA to other ACE appliances that have been configured in a Fault Tolerant group.

Supported sticky attributes include:

- IP address (source, destination or both)
- HTTP Cookie: A cookie is a small data structure within the HTTP header that is used by a server to deliver data to a Web client and requests that client store the information. ACE can then use the information in the cookie or URL to direct the content request to the appropriate server
- HTTP header
- Configuration synchronization

HA setup in ACE is on a per-context basis; therefore, the ACE configuration synchronization is per-context as well.

There are two types of configuration sync, both of which will be needed:

- Bulk sync, where the entire configuration is transferred from "Active" to "Standby". Bulk sync is required when adding a new ACE peer to form a redundant pair.
- Dynamic sync, where configuration changes are copied to the "Standby' context after it is reachable.

Incremental sync occurs after an ACE pair has been formed in steady state, and the primary ACE appliance proactively synchronizes each new configuration change with the standby ACE.

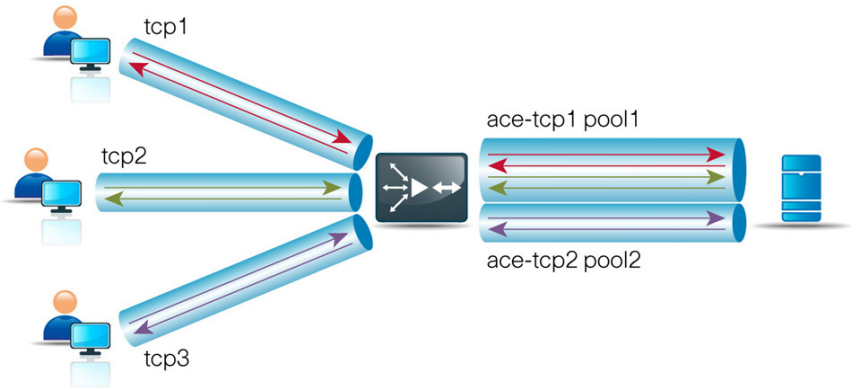
Connection State Replication

Connection replication is supported with ACE while the switchover occurs between the HA pair. However, very short-term or proxy connections cannot be synced across. Persistent HTTP connections and other long-lived TCP connections will be replicated across the HA pair. This state replication traffic runs over the FT VLAN.

TCP Reuse

TCP reuse, also known as TCP multiplexing, is designed to improve application server performance by consolidating incoming TCP/IP requests to reduce the number of times server connections have to be made.

Figure 6. TCP Reuse



Configuration Rollback

ACE supports the creation of configuration snapshots so that the previous configuration can be rolled back in case mishaps happen during configuration changes.

Deep Packet Inspection

ACE performs application inspections of HTTP, File Transfer Protocol (FTP), Domain Name Server (DNS), Internet Control Message Protocol (ICMP), and Real Time Streaming Protocol (RTSP) pro¬tocols as a first step before passing the packets to the destination server.

- **HTTP:** The ACE performs a stateful deep packet inspection of the HTTP protocol. During HTTP deep inspection, the main focus of the application inspection process is on HTTP attributes such as the HTTP header, the URL, and, to a limited extent, the payload. User-defined regular expressions can also be used to detect "signatures" in the payload.
- **FTP:** FTP inspection inspects FTP sessions for address translation in a message, dynamic opening of ports, and stateful tracking of request and response messages. Each specified FTP command must be acknowledged before the ACE allows a new command. Command filtering allows you to restrict specific commands by the ACE. When the ACE denies a command, it closes the connection.

- **DNS:** Domain Name System (DNS) inspection performs the following tasks:
  - Monitors the message exchange to ensure that the ID of the DNS response matches the ID of the DNS query.
    - Allows one DNS response for each DNS query in a User Datagram Protocol (UDP) connection. The ACE removes the DNS session associated with the DNS query as soon as the DNS reply is forwarded.
    - Translates the DNS A-record based on the Network Address Translation (NAT) configuration. Only forward lookups are translated using NAT; the ACE does not handle pointer records (PTR).
    - Performs a number of security checks, including:
      - Verifies that the maximum label length is no greater than 63 bytes.
      - Verifies that the maximum domain name length is no greater than 255 bytes.
      - Checks for the existence of compression loops.
- **ICMP:** Allows ICMP traffic to have a "session" so that it can be inspected similarly to TCP and UDP traffic. If you do not use ICMP inspection, we recommend that you not create an ACL that allows ICMP traffic to pass through the ACE. Without stateful inspection, ICMP can be used to attack your network. ICMP inspection ensures that there is only one response for each request, and that the sequence number is correct.
- **RTSP:** The Real Time Streaming Protocol (RTSP) is used by RealAudio, RealNetworks, Apple QuickTime 4, RealPlayer, and Cisco IP/TV connections. RTSP applications use the well-known port 554 with TCP and UDP as the control channel. The ACE supports TCP only in conformity with RFC 2326.

Application Acceleration

Used to increase the performance of data center-based Web applications, the Cisco ACE 4710 uses asymmetric compression, FlashForward, and other patented techniques to reduce the volume of transferred data and improve response times to end users, thus providing a better overall end-user experience.

- **Delta Optimization:** Allows the ACE to enable the content provider to dynamically calculate the content differences, or deltas, between subsequent content retrievals (on a per-user basis if desired) and send only those deltas for subsequent visits to the dynamic content. A user would retrieve 70 KB on the first visit to the home page but would need to retrieve only the 2 KB content delta on subsequent visits. With delta optimization, only the deltas between the subsequently requested pages are sent to the users. These deltas, which are encoded by using dynamic HTML, enable the ACE to directly update the client's browser cache, much like an origin server updates a traditional edge cache.
- **FlashForward Object Acceleration:** Eliminates the network delays associated with embedded Web objects such as images, style sheets, and JavaScript files. Without the ACE, the user experiences delays when pages with graphic images load because each object requires validation to ensure that the user has the latest version. Object validation can result in 20 KB or more of unnecessary upstream traffic. Each validation involves an HTTP request from the client to the server. FlashForward enforces embedded object version management at the server. All object validity information is carried in the single download of the parent HTML document, which eliminates unnecessary validation requests.

Deploying ACE

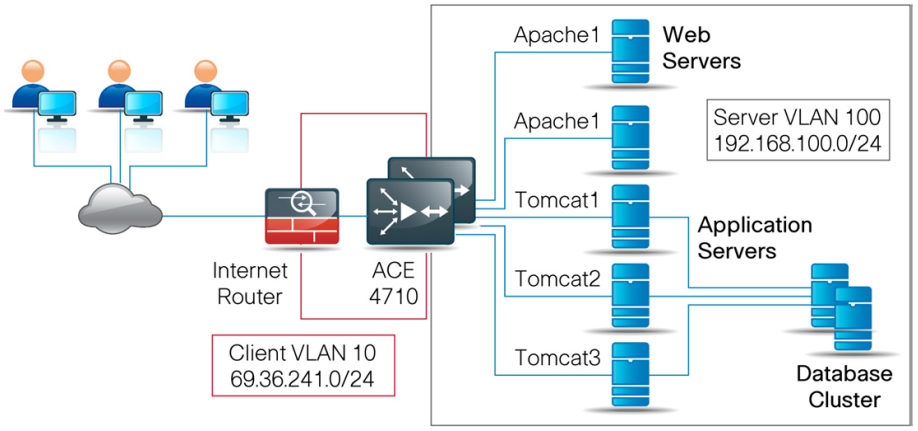
The following case study, which explains the ACME Company's current challenges and setup, will be used to illustrate how to deploy ACE.

ACME Company Overview

ACME Company is deploying a new e-commerce website to allow customers to browse products and place orders online. The underlying Web application utilizes a 3-tier architecture with Apache Web Server serving static content, Tomcat Application Server serving the dynamic content, and MySQL database server for customer and product information storage and retrieval.

To ensure the e-commerce website is available around the clock, a pair of ACE 4710 load balancers will be deployed along with 2 Apache servers, 3 Tomcat servers, and a MySQL cluster as shown in Figure 7. ACME's customers will access the new website through http://www.acme.com, which points to a VIP on a pair of ACE 4710s. Depending on the actual Web page or object being requested, the traffic will be forwarded to one of the two Apache Web Servers or one of the three Tomcat Application Servers. The Tomcat servers communicate directly with the MySQL cluster for information storage and retrieval.

Figure 7. E-Commerce Web Application Network Layout



For maximum network security, the servers will reside on a separate network with private IP addresses. ACE 4710 will be deployed in "routed" mode and serve as the default gateway for the servers. Only the VIP on the ACE will be exposed to the Internet.

The Apache Web Servers have been configured to run on TCP port 80 and serve static files under directories/images, /css, and /js. A static maintenance page also resides on the Apache Web Server under /sry.html. This page will inform users that the site is undergoing maintenance when three of the Tomcat Application Servers are unavailable.

The Tomcat Application Servers have been configured to run on TCP port 8080 and serve dynamically generated files that are not served by the Apache Web Servers. The application servers use HTTP sessions to maintain state information as a user navigates ACME's e-commerce website. This session allows the Web application to "personalize" the user experience by keeping track of the individual users during the course of navigation, including any transactions. Therefore, ACE 4710 will be configured with session persistence to ensure that each user is returned to the Tomcat Application Server where their session resides.

To provide secure shopping sessions, a SSL certificate has been purchased for www.acme.com and will be installed on the ACE 4710 for encryption and decryption of the secure traffic. All requests to /cart need to be SSL-encrypted. Should a user issue nonencrypted requests to /cart, those requests are redirected so that they are encrypted.

A summary of the ACME E-Commerce application environment is shown in Figure 8.

Figure 8. ACME E-Commerce Website Application Summary

Fully Qualified Domain Name		www.acme.com
Virtual IP Address:		69.36.241.10
Protocols		HTTP & HTTPS
Client VLAN		10
Server VLAN		100
ACE default gateway		69.36.241.1
ACE 4710 (Primary) VLAN 10 ip address		69.36.241.4
ACE 4710 (Secondary) VLAN 10 ip address		69.36.241.5
ACE 4710 (Primary) VLAN 100 ip address		192.168.100.2
ACE 4710 (Secondary) VLAN 100 ip address		192.168.100.3
Server default gateway		192.168.100.1
URI Path	**Served By**	**Additional Settings**
/images /css /js /sry.html	Apache Web Servers apache1 (192.168.100.11:80) apache2 (192.168.100.12:80)	
/cart	Tomcat Application Servers	No HTTP traffic allowed, HTTPS only
/* (everything else)	Tomcat Application Servers tomcat1 (192.168.100.21:8080) tomcat2 (192.168.100.22:8080) tomcat3 (192.168.100.23:8080)	Sorry Server Page (http:// www.acme.com/sry.html) Session Persistence

Tech Tip

These instructions do not cover the initial setup and configuration of ACE 4710. For basic setup instructions, please refer to the *Cisco SBA for Midsize Agencies—Data Center Deployment Guide*.

Process

Configuring the Network

1. Define the Virtual Context
2. Add a Health Probe
3. Configure the Real Server
4. Configure the Server Farm
5. Configure Session Persistence (Stickiness)
6. Configure the Virtual Server
7. Configure SSL (HTTPS)

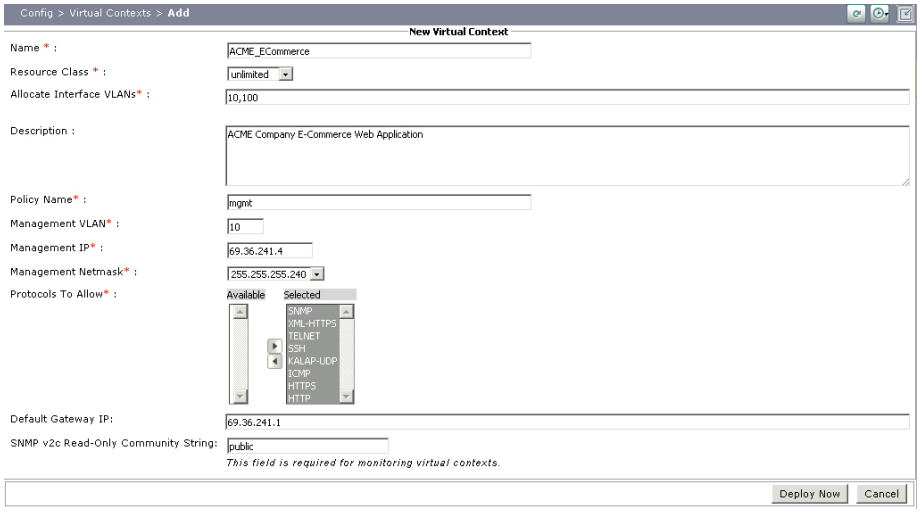
This process will explain how to set up a pair of ACE 4710s in order to provide load balancing for the application environment described in the above scenario.

Procedure 1 Define the Virtual Context

The configuration for the ACME e-commerce website will be contained within a separate virtualized context on the ACE4710.

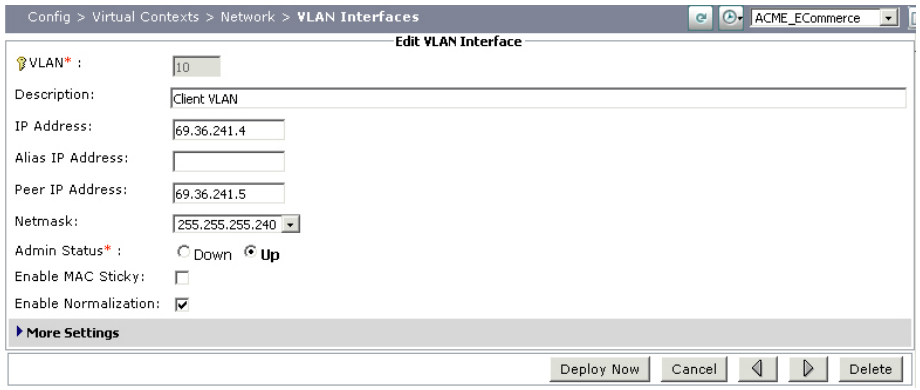
Step 1: Go to **Config > Virtual Contexts**, and click [+] to add a new virtual context. See Figure 9.

Figure 9. New Virtual Context



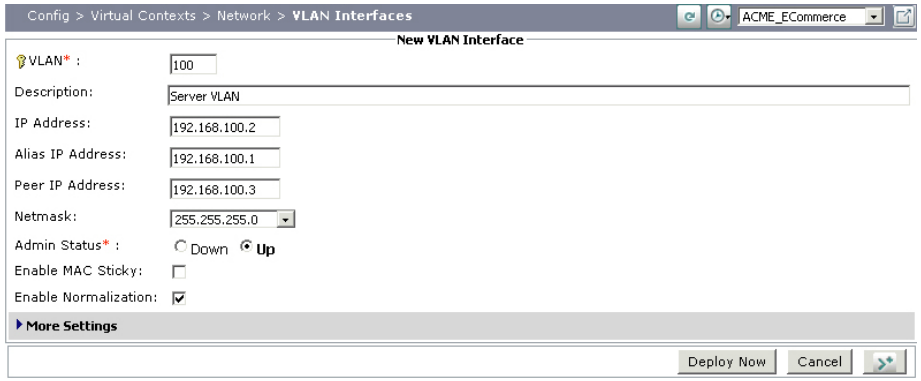
Step 2: Under **Network > VLAN Interfaces**, double click on the interface for Client VLAN (VLAN 10) to edit the VLAN Interface and add the IP address for the secondary ACE 4710 in the Peer IP Address dialog box. See Figure 10.

Figure 10. Edit VLAN Interfaces



Step 3: Next, under **Network > VLAN Interfaces**, click [+] to add a new VLAN Interface for the Server VLAN, as shown in Figure 11.

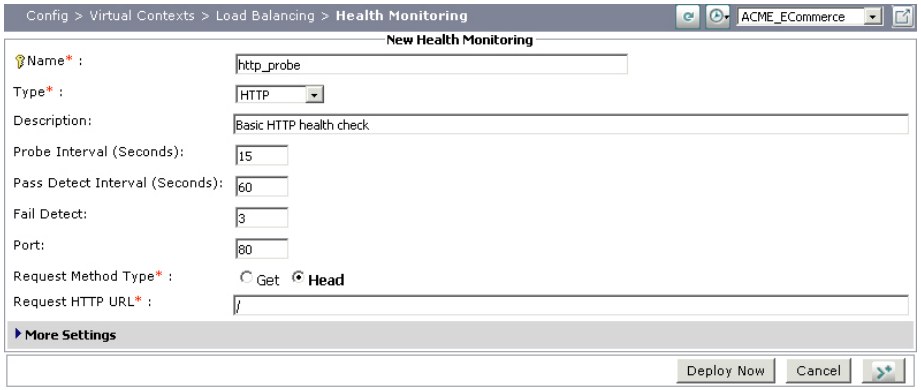
Figure 11. New VLAN Interface



Procedure 2 Add a Health Probe

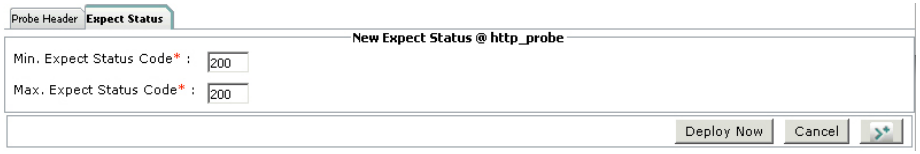
Step 1: Under **Load Balancing > Health Monitoring**, click [+] to add a new health probe to perform a basic HTTP health check. We'll add this health probe when creating Server Farms, as shown in Figure 12.

Figure 12. New Health Monitoring



Step 2: After a probe has been created, select the **Expect Status** tab at the bottom of the page, and enter the value **200** for both Max and Min Expected Status Code, as shown in Figure 13.

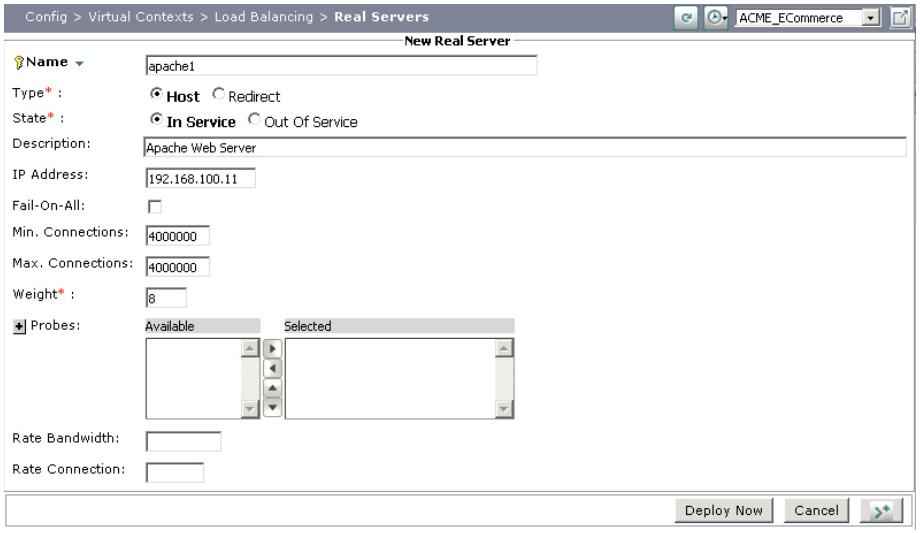
Figure 13. Expect Status Tab



Procedure 3 Configure the Real Server

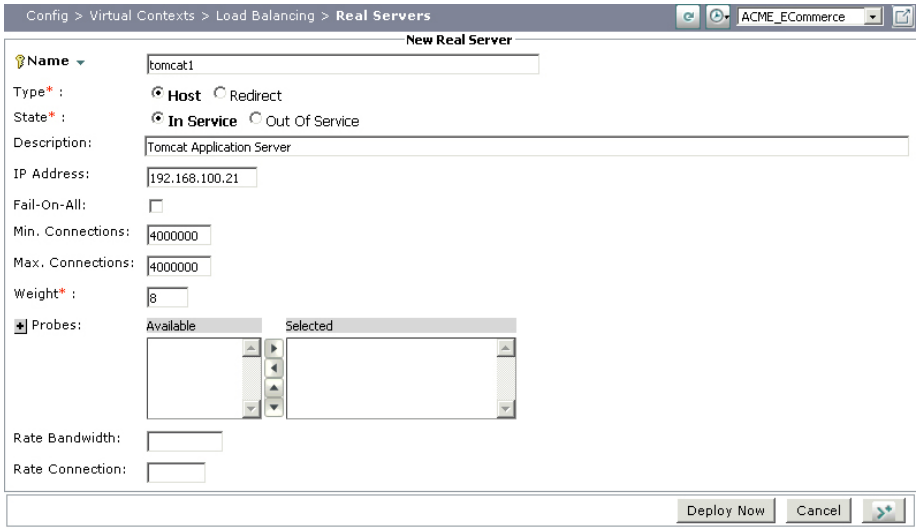
Step 1: Under **Load Balancing > Real Servers**, add 2 real servers for Apache Web Servers, named **apache1** and **apache2**, as shown in Figure 14.

Figure 14. New Real Server



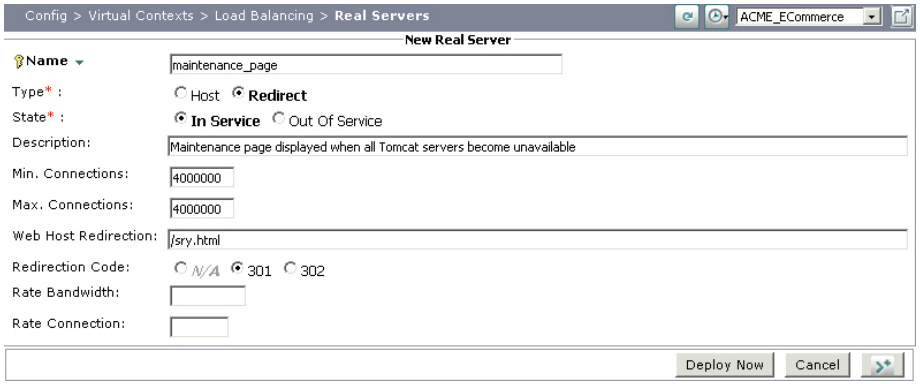
Step 2: Under **Load Balancing > Real Servers**, add 3 real servers for Tomcat Application Servers, named **tomcat1**, **tomcat2**, and **tomcat3**, as shown in Figure 15.

Figure 15. New Real Server



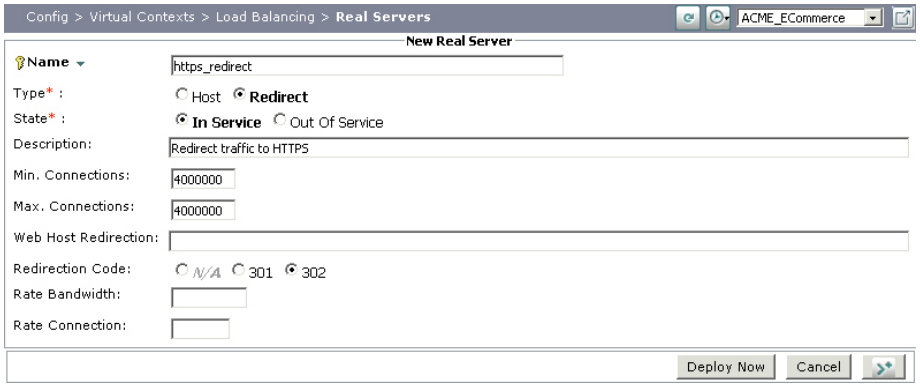
Step 3: Under **Load Balancing > Real Servers**, add a redirect real server called **maintenance_page** for the maintenance page redirection, as shown in Figure 16.

Figure 16. Redirection for Application Failure



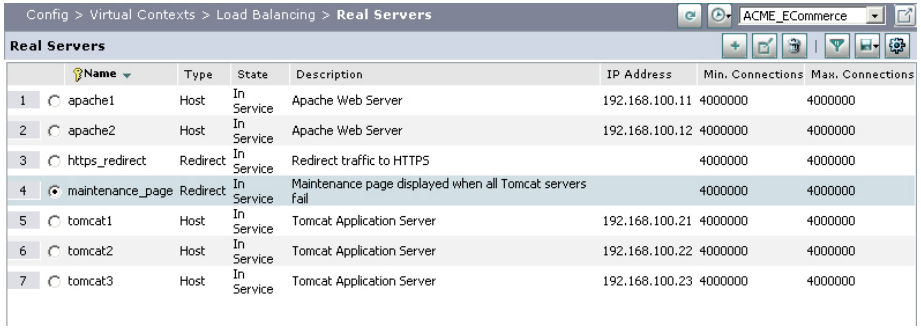
Step 4: Under **Load Balancing > Real Servers**, add a redirect real server called **https_redirect** for redirecting users from cleartext (HTTP) to encrypted sessions (HTTPS) , as shown in Figure 17.

Figure 17. SSL Redirect



A summary of all the real servers created for the ACME e-commerce web-site is shown in Figure 18.

Figure 18. Real Server Summary



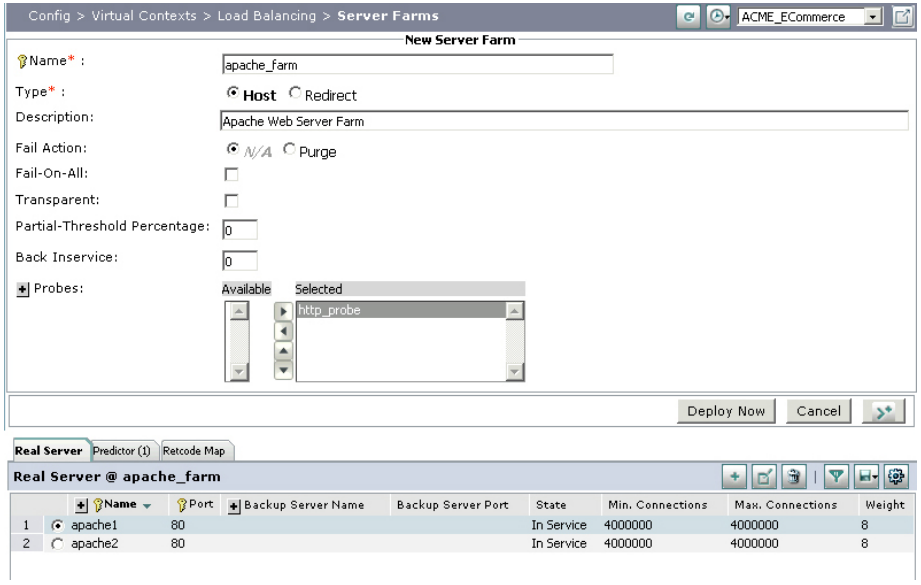
Procedure 4 Configure the Server Farm

Step 1: Under **Load Balancing > Server Farms**, add a server farm called **apache_farm** to group the real servers for the Apache Web Servers, as shown in Figure 19.

Step 2: The **http_probe** help probe created earlier should appear under the Available column. Move the probe to the Selected column by highlighting it and clicking on the right arrow.

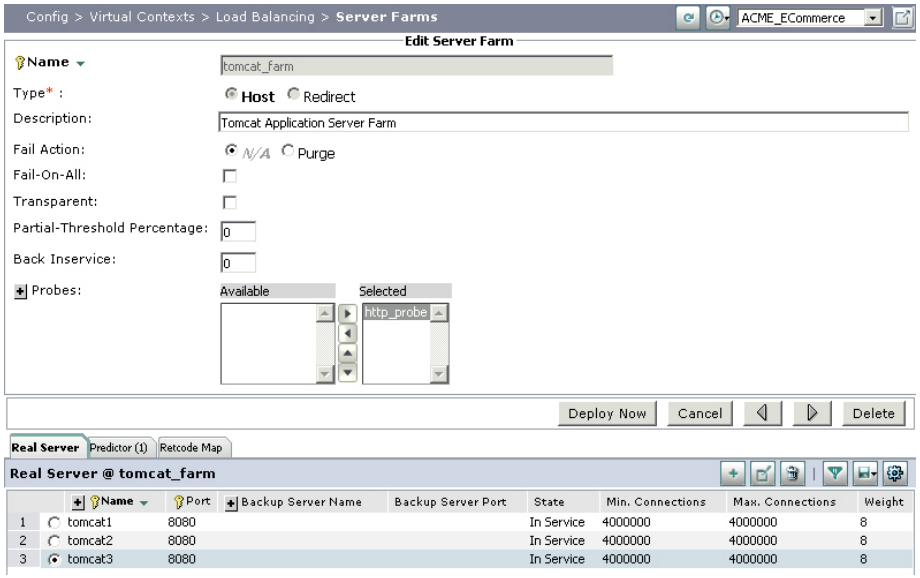
Step 3: Next, add **apache1** and **apache2** to the **apache_farm** server farm.

Figure 19. New Server Farm



Step 4: Under **Load Balancing > Server Farms**, add a server farm called **tomcat_farm** to group the real servers for the Tomcat Application Servers, as shown in Figure 20. Next, add **tomcat1**, **tomcat2**, and **tomcat3** to the **tomcat_farm** server farm.

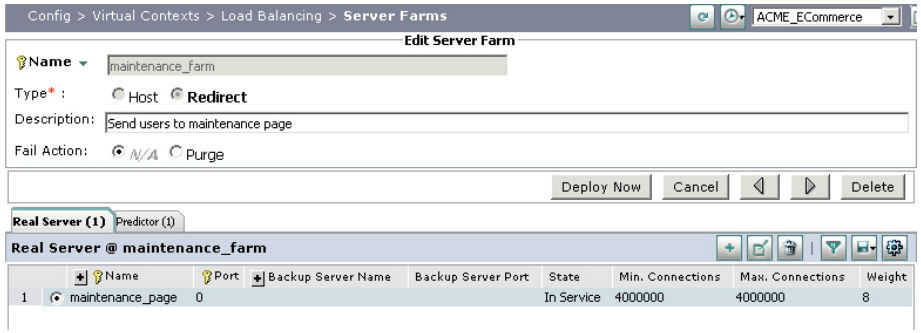
Figure 20. Edit Server Farm



Step 5: Under **Load Balancing > Server Farms**, add a redirect server farm called **maintenance_farm** to group the real server for the maintenance page.

Step 6: Add the **maintenance_page** real server to the **maintenance_farm** server farm, as shown in Figure 21.

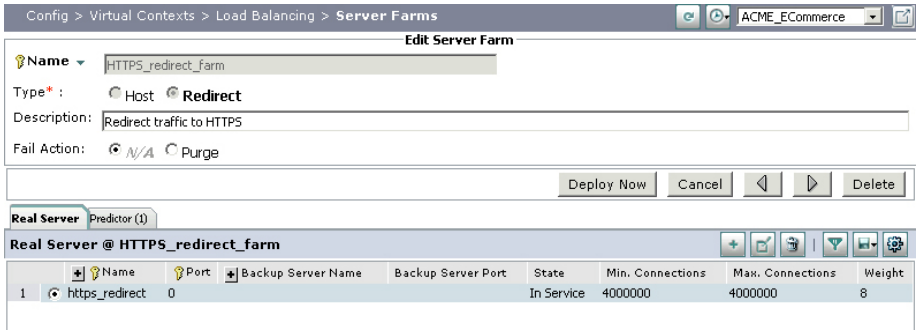
Figure 21. Edit Server Farm



Step 7: Under **Load Balancing > Server Farms**, add a redirect server farm called **HTTPS_redirect_farm** to group the real server for the HTTP to HTTPS redirection.

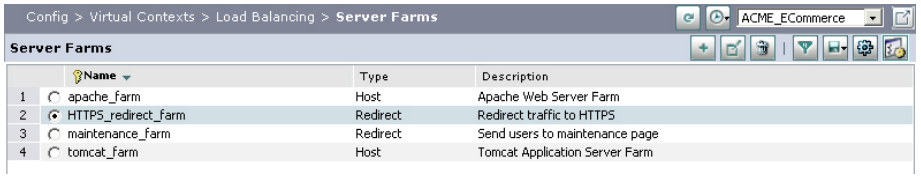
Step 8: Then add the **https_redirect** real server to the **HTTPS_redirect_farm** server farm, as shown in Figure 22.

Figure 22. Edit Server Farm



A summary of all the server farms created for the ACME e-commerce website is shown in Figure 23.

Figure 23. Summary of Server Farms



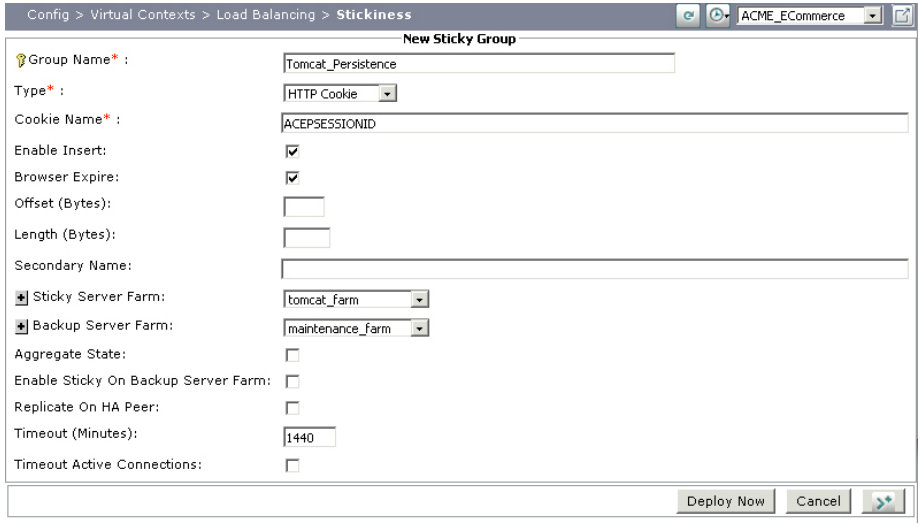
Procedure 6 Configure Session Persistence (Stickiness)

Step 1: Under **Load Balancing > Stickiness**, click to add a new Sticky Group called **Tomcat_Persistence** with type **HTTP_Cookie**. We'll use the cookie name **ACEPSESSIONID**, as shown in Figure 24.

Step 2: Check the checkboxes for **Enable Insert** and **Browser Expire**.

Step 3: Select **tomcat_farm** as the Sticky Server Farm and **maintenance_farm** as the Backup Server Farm.

Figure 24. New Sticky Group

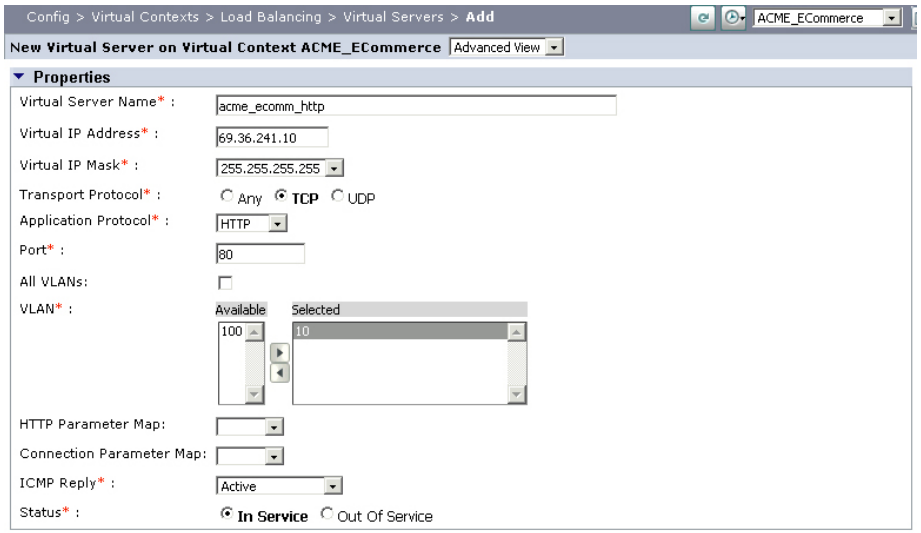


Procedure 7 Configure the Virtual Server

Step 1: Under **Load Balancing > Virtual Servers**, click [+] to add a new virtual server to respond to the HTTP requests destined for the VIP for the ACME e-commerce website.

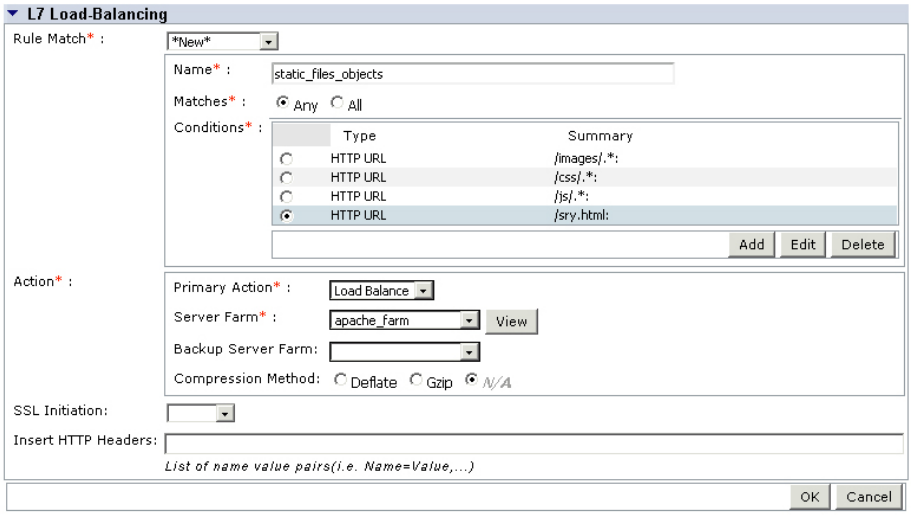
Step 2: Select **Advanced View** to expand the configuration options available, and complete the form fields for **Properties** as shown in Figure 25.

Figure 25. Add Virtual Server



Step 3: Under **L7 Load-Balancing**, create a new rule called **static_files_objects** to match requests for static files that are served from the Apache Web Servers. Set the Action to load balance traffic to the **apache_farm** server farm, as shown in Figure 27.

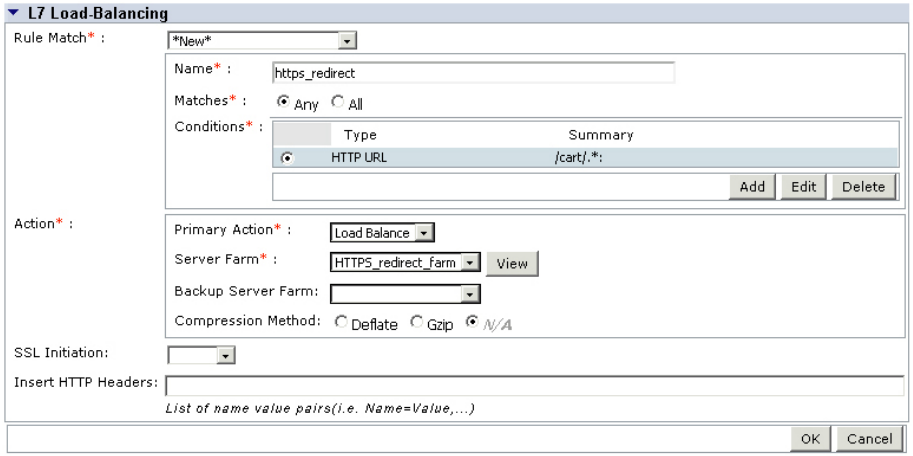
Figure 26. L7 Load Balancing



Step 4: Under **L7 Load-Balancing**, create a new rule called **https_redirect** to match requests for the secure page that should always be encrypted (HTTPS) , as shown in Figure 27.

Step 5: Set the **Action** to load balance traffic to the **HTTPS_redirect_farm** server farm.

Figure 27. L7 Load Balancing

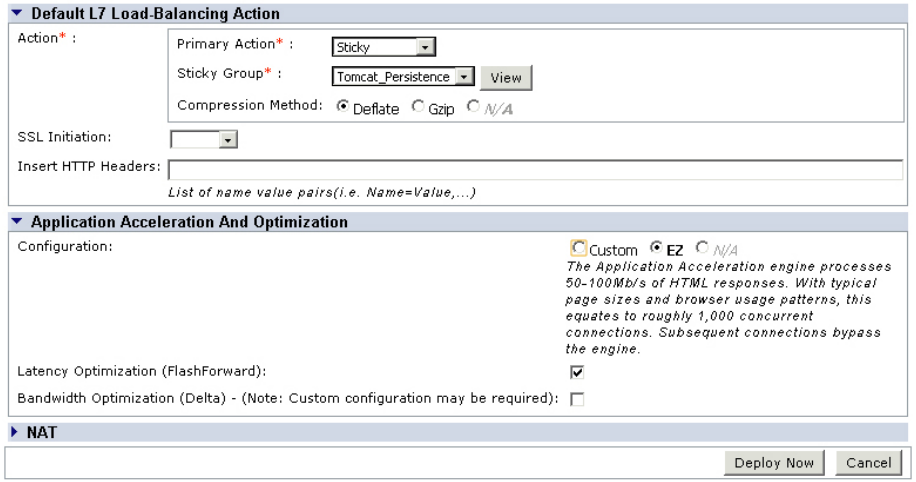


Step 6: Under **Default L7 Load-Balancing Action**, set the Action to use **Sticky** with the **Tomcat_Persistence** Sticky Group, as shown in Figure 28.

Step 7: Select **Deflate** as the Compression Method.

Step 8: Under **Application Acceleration and Optimization**, select **EZ**, and check **Latency Optimization (FlashForward)**.

Figure 28. Default L7 Load Balancing Actions

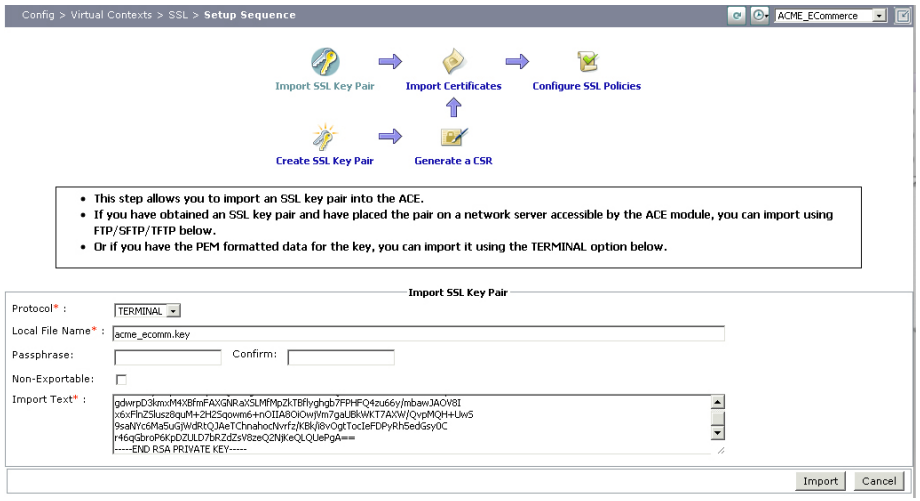


Procedure 8 Configure SSL (HTTPS)

This section guides you through the configuration of SSL termination with the existing SSL private key and certificate.

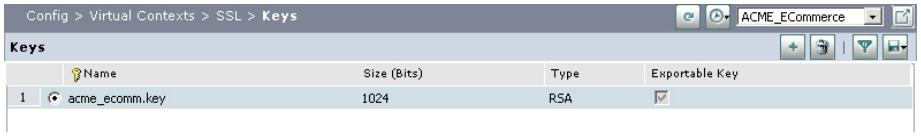
Step 1: Under **SSL > Setup Sequence**, import the SSL private key by copying and pasting the PEM-encoded key into the **Import Text** form field under TERMINAL Protocol, as shown in Figure 29.

Figure 29. Startup Sequence



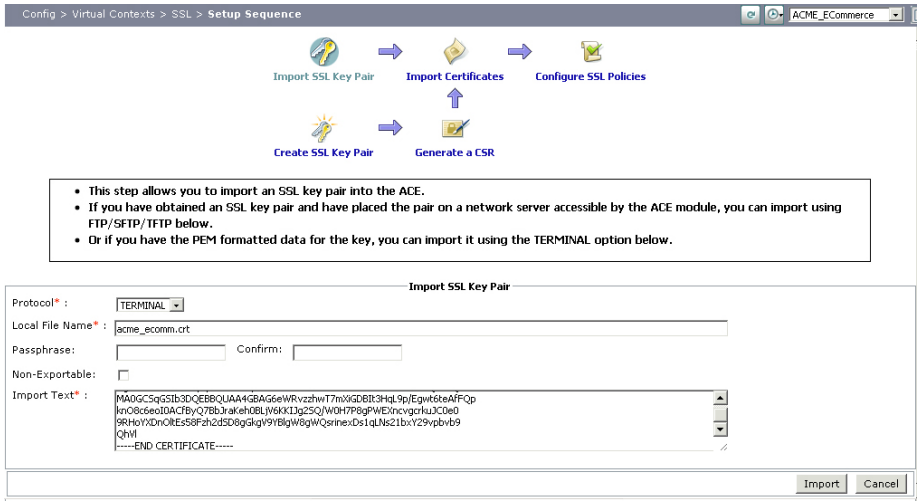
The imported key is listed under **SSL > Keys** as shown in Figure 30.

Figure 30. Keys



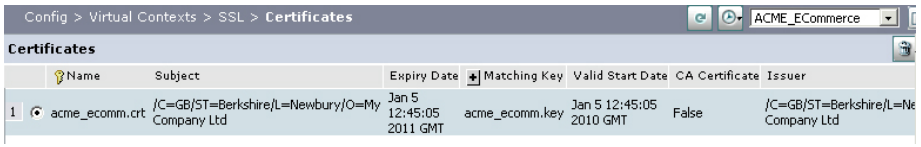
Step 2: Under **SSL > Setup Sequence**, import the SSL certificate by copying and pasting the PEM-encoded certificate into Import Text form field under TERMINAL Protocol, as shown in Figure 31.

Figure 31. Startup Sequence



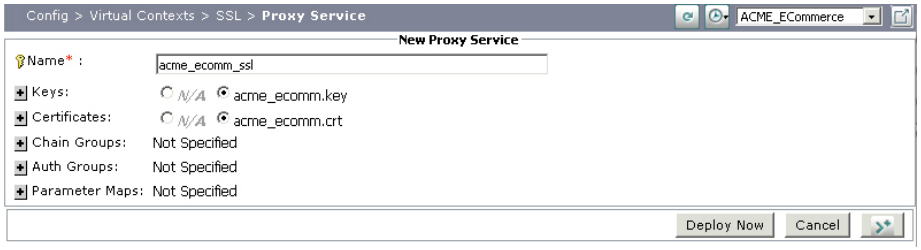
The imported key is listed under **SSL > Certificates** as shown in Figure 32.

Figure 32. Certificates



Step 3: Under **SSL > Proxy Service**, click to add a new Proxy Service called **acme_ ecomm_ssl** to associate the imported private key and certificate, as shown in Figure 33.

Figure 33. New Proxy Service

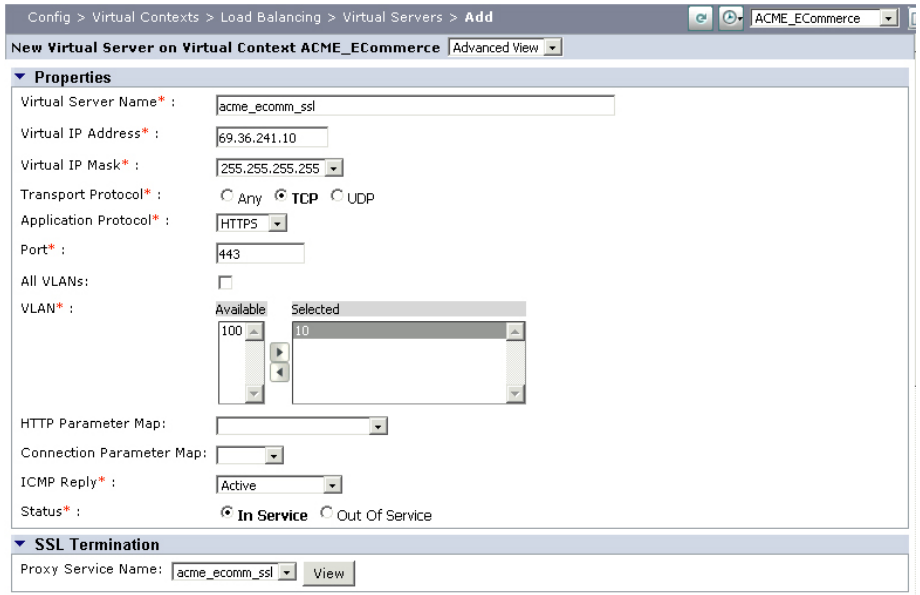


Step 4: Under **Load Balancing > Virtual Servers**, click to add a new virtual server to respond to the HTTPS requests destined for the VIP for the ACME e-commerce website.

Step 5: Select Advanced View to expand the configuration options available, and complete the form fields for Properties as shown in the picture below.

Step 6: Under **SSL Termination**, select the **acme_ecomm_ssl** Proxy Service, as shown in Figure 34.

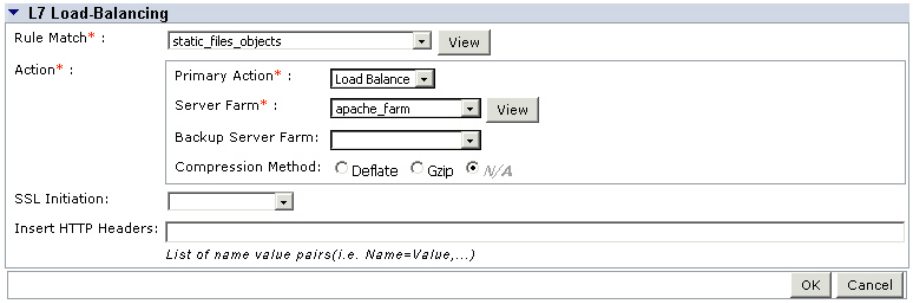
Figure 34. Properties



Step 7: Under **L7 Load-Balancing**, select the **static_files_objects** rule for static files that are served from the Apache Web Servers, as shown in Figure 35.

Step 8: Set the Action to load balance traffic to the **apache_farm** server farm.

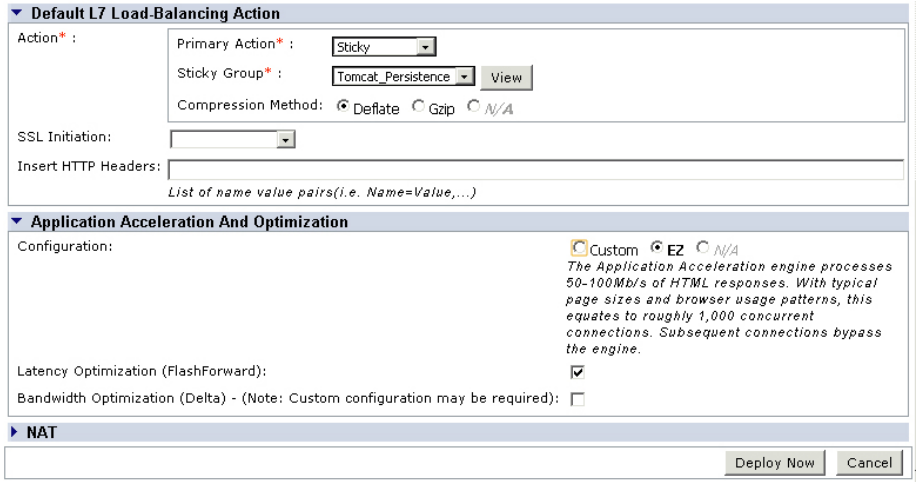
Figure 35. Static File Load Balancing for Web Service



Step 9: Under **Default L7 Load-Balancing Action**, set the Action to use **Sticky** with the **Tomcat_Persistence** Sticky Group. Select **Deflate** as the Compression Method, as shown in Figure 36.

Step 10: Under **Application Acceleration and Optimization**, select **EZ**, and check **Latency Optimization (FlashForward)**.

Figure 36. Default L7 Load Balancing Action



Appendix A: ACE 4710 Configuration

```
crypto csr-params acme_ecomm.csr
  country US
  state CA
  common-name www.acme.com

probe http HTTP_Head_Probe
  description Basic HTTP Probe that checks / returns 200 OK
  port 80
  interval 15
  passdetect interval 60
  request method head
  open 1

parameter-map type http cisco_avs_parametermap
  case-insensitive
  persistence-rebalance

rserver host apache1
  description Apache Web Server
  ip address 192.168.100.11
  inservice
rserver host apache2
  description Apache Web Server
  ip address 192.168.100.12
  inservice
rserver redirect force_https
  description Redirect traffic to HTTPS
  webhost-redirection https://%h%p 302
  inservice
rserver redirect maintenance_page
  description Maintenance page displayed when all Tomcat
  servers fail
  webhost-redirection /sry.html 302
  inservice
rserver host tomcat1
  description Tomcat Application Server
  ip address 192.168.100.21
  inservice
rserver host tomcat2
  description Tomcat Application Server
  ip address 192.168.100.22
  inservice
rserver host tomcat3
  description Tomcat Application Server
  ip address 192.168.100.23
  inservice
action-list type optimization http cisco_avs_container_latency
  flashforward
action-list type optimization http cisco_avs_img_latency
  flashforward-object
action-list type optimization http cisco_avs_obj_latency
  flashforward-object

serverfarm redirect HTTPS_redirect_farm
  description Redirect traffic to HTTPS
  rserver force_https
    inservice
serverfarm host apache_farm
  description Apache Web Server Farm
  probe HTTP_Head_Probe
  rserver apache1 80
    inservice
  rserver apache2 80
    inservice
serverfarm redirect maintenance_farm
  description Send users to maintenance page
  rserver maintenance_page
    inservice
serverfarm host tomcat_farm
  description Tomcat Application Server Farm
  probe HTTP_Head_Probe
  rserver tomcat1 8080
    inservice
  rserver tomcat2 8080
    inservice
  rserver tomcat3 8080
    inservice

ssl-proxy service acme_ecomm_ssl
  key acme_ecomm.key
  cert acme_ecomm.crt

sticky http-cookie ACEPSESSIONID Tomcat_Persistence
  cookie insert browser-expire
  serverfarm tomcat_farm backup maintenance_farm
```

```
class-map match-all acme_ecomm_http
  2 match virtual-address 69.36.241.10 tcp eq www
class-map match-all acme_ecomm_https
  2 match virtual-address 69.36.241.10 tcp eq https
class-map type http loadbalance match-all cisco_avs_container_
latency
  2 match http url .*
class-map type http loadbalance match-any cisco_avs_img_
latency
  2 match http url .*jpg
  3 match http url .*jpeg
  4 match http url .*jpe
  5 match http url .*png
class-map type http loadbalance match-any cisco_avs_obj_
latency
  2 match http url .*gif
  3 match http url .*css
  4 match http url .*js
  5 match http url .*class
  6 match http url .*jar
  7 match http url .*cab
  8 match http url .*txt
  9 match http url .*ps
  10 match http url .*vbs
  11 match http url .*xsl
  12 match http url .*xml
  13 match http url .*pdf
  14 match http url .*swf
class-map type http loadbalance match-any default-compression-
exclusion-mime-type
  description DM generated classmap for default LB compression
  exclusion mime types.
  2 match http url .*gif
  3 match http url .*css
  4 match http url .*js
  5 match http url .*class
  6 match http url .*jar
  7 match http url .*cab
  8 match http url .*txt
  9 match http url .*ps
  10 match http url .*vbs
  11 match http url .*xsl
  12 match http url .*xml
  13 match http url .*pdf
  14 match http url .*swf
  15 match http url .*jpg
  16 match http url .*jpeg
  17 match http url .*jpe
  18 match http url .*png
class-map type http loadbalance match-any https_redirect
  2 match http url /cart/.*
class-map type management match-any mgmt
  201 match protocol snmp any
  202 match protocol xml-https any
  203 match protocol telnet any
  204 match protocol ssh any
  205 match protocol kalap-udp any
  206 match protocol icmp any
  207 match protocol https any
  208 match protocol http any
class-map type http loadbalance match-any static_files_objects
  2 match http url /images/.*
  3 match http url /css/.*
  4 match http url /js/.*
  5 match http url /sry.html

policy-map type management first-match mgmt
  class mgmt
    permit

policy-map type loadbalance first-match acme_ecomm_http-l7slb
  class default-compression-exclusion-mime-type
    sticky-serverfarm Tomcat_Persistence
  class static_files_objects
    serverfarm apache_farm
  class https_redirect
    serverfarm HTTPS_redirect_farm
  class class-default
    compress default-method deflate
    sticky-serverfarm Tomcat_Persistence
policy-map type loadbalance first-match acme_ecomm_https-l7slb
  class default-compression-exclusion-mime-type
    sticky-serverfarm Tomcat_Persistence
  class static_files_objects
    serverfarm apache_farm
  class class-default
    compress default-method deflate
    sticky-serverfarm Tomcat_Persistence
policy-map type optimization http first-match acme_ecomm_http-
l7opt
  class cisco_avs_obj_latency
    action cisco_avs_obj_latency
  class cisco_avs_img_latency
    action cisco_avs_img_latency
  class cisco_avs_container_latency
    action cisco_avs_container_latency
```

```
policy-map type optimization http first-match acme_ecomm_
https-l7opt
  class cisco_avs_obj_latency
    action cisco_avs_obj_latency
  class cisco_avs_img_latency
    action cisco_avs_img_latency
  class cisco_avs_container_latency
    action cisco_avs_container_latency
policy-map multi-match int10
  class acme_ecomm_http
    loadbalance vip inservice
    loadbalance policy acme_ecomm_http-l7slb
    optimize http policy acme_ecomm_http-l7opt
    loadbalance vip icmp-reply active
    appl-parameter http advanced-options cisco_avs_parametermap
  class acme_ecomm_https
    loadbalance vip inservice
    loadbalance policy acme_ecomm_https-l7slb
    optimize http policy acme_ecomm_https-l7opt
    loadbalance vip icmp-reply active
    appl-parameter http advanced-options cisco_avs_parametermap
    ssl-proxy server acme_ecomm_ssl

interface vlan 10
  description "Client VLAN"
  ip address 69.36.241.4 255.255.255.240
  peer ip address 69.36.241.5 255.255.255.240
  service-policy input mgmt
  service-policy input int10
  no shutdown
interface vlan 100
  description "Server VLAN"
  ip address 192.168.100.2 255.255.255.0
  alias 192.168.100.1 255.255.255.0
  peer ip address 192.168.100.3 255.255.255.0
  no shutdown

ip route 0.0.0.0 0.0.0.0 69.36.241.1
```

Appendix B: Glossary

This section provides a quick overview of the terminologies commonly used when working with application servers and server load balancers.

Real Server

Real servers are Application Servers. They usually sit behind a server load-balancer. In a typical network, there would be one or more real servers running the same application instance to service clients/users. Load balancers allow administrators to add or remove real servers from the available pool without impacting service availability.

Virtual Server

Virtual server represents the logical instance of the application residing on the load balancer, which in turn is mapped to a pool of real servers that actually provide content and application services. There can be one or more virtual server instances defined on the load balancer. Clients connect to the virtual server IP address assigned to the load balancer, and the load balancer distributes these requests among multiple real servers.

Health Probe

Health probe is a mechanism by which the hardware load balancer verifies that an application instance or the server is capable of delivering appropriate service in response to end-user client requests. If a real server or an application on a real server fails health checks, then the load balancer will take that instance out of rotation.

Load Balancing Predictor

The predictor algorithm determines how to balance client requests across multiple real servers. Common predictors include:

- **Least Connections:** Sends the request to the real server that currently has the fewest active connections with clients.
- **Round Robin:** Directs the service request to the next server, and treats all servers equally regardless of the number of connections or response time.
- **Weighted:** Assigns a performance weight to each server. Weighted load balancing is similar to least connections, except servers with a higher weight value receive a larger percentage of connections at a time.

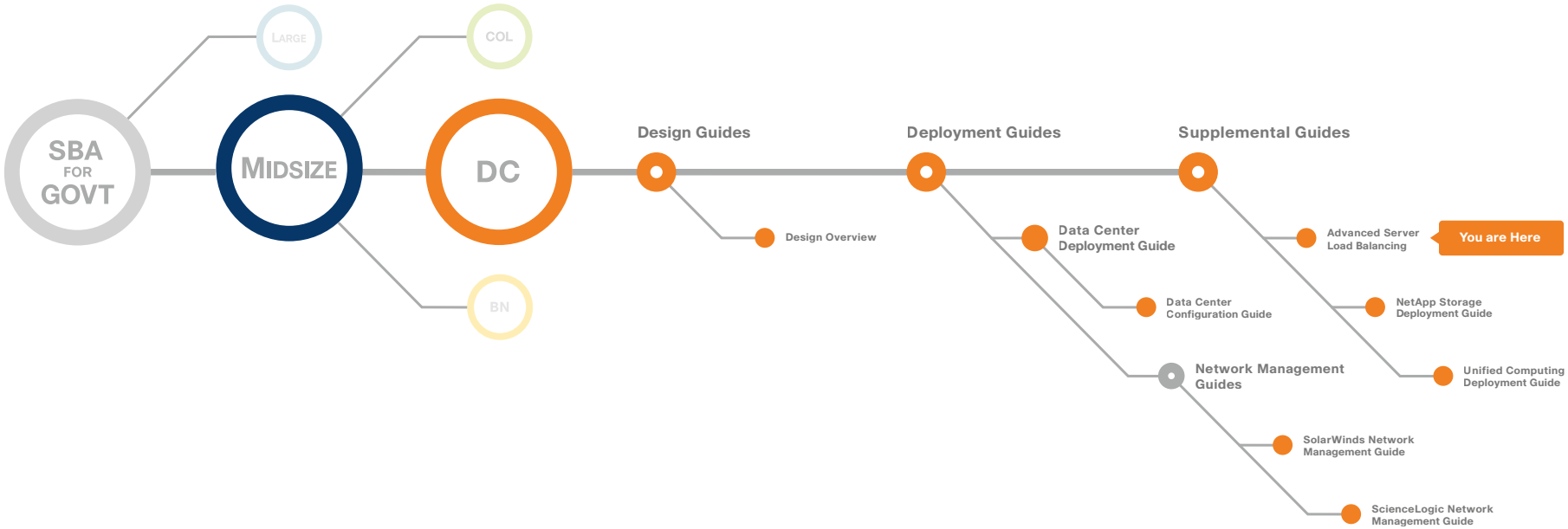
Sticky Connections

If an application requires a series of sequential TCP/UDP port connections to be serviced by the same real server, then the sticky feature can be enabled for that virtual application port.

SSL Acceleration

SSL was developed to provide security and privacy over the Internet. Today, the most secure applications over the Internet use SSL. SSL provides a secure pipe and allows protocols such as http, ftp, and LDAP to run inside it.

Appendix C:
SBA for Midsize Agencies Document System



Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV
Amsterdam, The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **www.cisco.com/go/offices.**

C07-641155-00 12/10