Cisco Application Control Engine
Module Security Configuration Guide

Software Version A4(1.0)
November 2010

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-23568-01
THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS” WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)
CONTENTS

Preface xvii
   Audience xviii
   How to Use This Guide xviii
   Related Documentation xix
   Symbols and Conventions xxii
   Obtaining Documentation, Obtaining Support, and Security Guidelines xxiv

CHAPTER 1 Configuring Security Access Control Lists 1-1
   ACL Overview 1-2
      ACL Types and Uses 1-3
      ACL Guidelines 1-3
         ACL Entry Order 1-3
         ACL Implicit Deny 1-4
         Maximum Number of ACLs and ACL Entries 1-4
   ACL Configuration Quick Start 1-5
   Configuring ACLs 1-6
      Configuring an Extended ACL 1-6
      Configuring Comments in an Extended ACL 1-16
      Configuring an EtherType ACL 1-17
      Resequencing Entries 1-19
   Simplifying Access Control Lists with Object Groups 1-20
      Overview of Object Groups 1-20
      Configuring Network Object Groups 1-21
Contents

Creating a Network Object Group  1-21
Adding a Description to a Network Object Group  1-22
Configuring a Network IP Address for a Network Object Group  1-22
Configuring a Host IP Address  1-23
Configuring Service Object Groups  1-24
Creating a Service Object Group  1-24
Adding a Description to a Service Object Group  1-25
Defining Protocol Parameters for a Service Object Group  1-25
Using Object Groups in an ACL  1-27
Example of Configuring an Extended ACL Without Object Groups  1-28
Example of Configuring the Equivalent Extended ACL Using Object Groups  1-29
Example of How an Object Group Expands into Multiple ACL Entries  1-29
Applying an ACL to an Interface  1-30
Applying an ACL Globally to All Interfaces in a Context  1-31
Filtering Traffic with an ACL  1-32
ACL Configuration Examples  1-33
Examples of Extended ACLs  1-33
Inbound and Outbound ACLs  1-34
IP Addresses for ACLs with NAT  1-37
Examples of EtherType ACLs  1-41
Displaying ACL Configuration Information and Statistics  1-41
Displaying ACL Configuration Information  1-42
Displaying ACL Statistics  1-42
Displaying the ACL Merge Tree Node Usage  1-44
Clearing ACL Statistics  1-44

CHAPTER 2

Configuring Authentication and Accounting Services  2-1
AAA Overview  2-2
Local Database and Remote Server Support 2-3
  Local Database 2-5
  TACACS+ Server 2-5
  RADIUS Server 2-5
  LDAP Directory Server 2-6
Authentication Overview 2-7
Accounting Overview 2-7
Authentication and Accounting Configuration Quick Start 2-8
Configuring the AAA Server 2-10
  Configuring a TACACS+ Server 2-10
    Configuring Authentication Settings on the TACACS+ Server 2-10
    Configuring Accounting Settings on the TACACS+ Server 2-11
    Defining Private Attributes for Virtualization Support in a TACACS+ Server 2-12
  Configuring a RADIUS Server 2-14
    Configuring Authentication Settings on the RADIUS Server 2-14
    Configuring Accounting Settings on the RADIUS Server 2-15
    Defining Private Attributes for Virtualization Support in a RADIUS Server 2-16
  Configuring an LDAP Server 2-18
    Defining Private Attributes for Virtualization Support in an LDAP Server 2-19
Creating User Accounts 2-21
Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server 2-22
  Configuring RADIUS on the ACE 2-23
    Setting the RADIUS Server Parameters 2-23
    Configuring the RADIUS NAS-IP-Address Attribute 2-26
    Setting the Global RADIUS Server Preshared Key 2-26
    Configuring the Global RADIUS Server Dead-Time Interval 2-27
    Setting the Global RADIUS Server Number of Retransmissions 2-28
Setting the Global RADIUS Server Timeout Value 2-29
Configuring TACACS+ on the ACE 2-29
Setting the TACACS+ Server Parameters 2-30
Setting the Global Preshared Key 2-31
Setting the Global TACACS+ Server Dead-Time Interval 2-32
Setting the Global TACACS+ Server Timeout Value 2-33
Configuring LDAP on the ACE 2-33
Setting the LDAP Server Parameters 2-34
Setting the Global LDAP Server Port Setting 2-35
Setting the Global LDAP Server Timeout Value 2-36
Configuring AAA Server Groups 2-36
Creating a TACACS+, RADIUS, or LDAP Server Group 2-37
Setting the Dead-Time Interval for a TACACS+ Server Group 2-39
Setting the Dead-Time Interval for a RADIUS Server Group 2-39
Configuring the User Profile Attribute Type for an LDAP Server Group 2-40
Configuring the Base DN for an LDAP Server Group 2-41
Configuring the Search Filter for an LDAP Server Group 2-42
Defining the Login Authentication Method 2-43
Defining the Default Accounting Method 2-45
Viewing AAA Status and Statistics 2-46
Displaying AAA Groups 2-47
Displaying RADIUS Server Configuration Information 2-47
Displaying TACACS+ Server Configuration Information 2-48
Displaying LDAP Server Configuration Information 2-49
Displaying Accounting Configuration Information 2-50
Displaying Accounting Log Information 2-50
Displaying Authentication Configuration Information 2-53
CHAPTER 3

Configuring Application Protocol Inspection 3-1

Application Protocol Inspection Overview 3-2
Performing Application Protocol Inspection 3-2
Application Inspection Protocol Overview 3-9

DNS Inspection 3-9
FTP Inspection 3-10
HTTP Deep Packet Inspection 3-12
ICMP Inspection 3-15
ILS Inspection 3-16
RTSP Inspection 3-18
SCCP Inspection 3-19
SIP Inspection 3-19

Application Protocol Inspection Configuration Quick Start Procedures 3-22

Configuring a Layer 7 FTP Command Inspection Policy 3-30

Configuring an FTP Inspection Class Map 3-31

Creating an FTP Inspection Class Map 3-31
Adding a Layer 7 FTP Inspection Class Map Description 3-32
Defining FTP Match Request Methods 3-33

Configuring a Layer 7 FTP Command Inspection Policy Map 3-34

Creating a Layer 7 FTP Command Inspection Policy Map 3-34
Adding a Layer 7 FTP Inspection Policy Map Description 3-35
Including Inline Match Statements in a Layer 7 FTP Command Inspection Policy Map 3-35

Associating a Layer 7 FTP Command Inspection Traffic Class with the Traffic Policy 3-37
Specifying the Layer 7 FTP Command Inspection Policy Actions 3-37

Configuring a Layer 7 HTTP Deep Inspection Policy 3-38

Configuring a Layer 7 HTTP Deep Inspection Class Map 3-39

Creating an HTTP Deep Inspection Class Map 3-40
Adding a Layer 7 HTTP Deep Packet Inspection Class Map
Description 3-42
Defining HTTP Content Match Criteria 3-42
Defining the Length of the HTTP Content for Inspection 3-43
Defining a Secondary Cookie for HTTP Inspection 3-45
Defining an HTTP Header for Inspection 3-48
Defining the HTTP Maximum Header Length for Inspection 3-52
Defining a Header MIME-Type Messages for Inspection 3-53
Defining an HTTP Traffic Restricted Category 3-56
Defining HTTP Request Methods and Extension Methods 3-57
Defining an HTTP Transfer Encoding Type 3-59
Defining an HTTP URL for Inspection 3-60
Defining an HTTP Maximum URL Length for Inspection 3-62
Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map 3-63
Creating a Layer 7 HTTP Deep Packet Inspection Policy Map 3-63
Adding a Layer 7 HTTP Deep Packet Inspection Policy Map Description 3-64
Including Inline Match Statements in a Layer 7 HTTP Deep Packet Inspection Policy Map 3-65
Associating a Layer 7 HTTP Inspection Traffic Class with the Traffic Policy 3-67
Specifying the Layer 7 HTTP Deep Packet Policy Actions 3-68
Configuring a Layer 7 SCCP Inspection Policy 3-70
Creating a Layer 7 SCCP Inspection Policy Map 3-70
Adding a Description to the Layer 7 SCCP Inspection Policy Map 3-71
Including an Inline Match Statement in a Layer 7 SCCP Inspection Policy Map 3-71
Specifying the Layer 7 SCCP Inspection Policy Map Action 3-72
Configuring a Layer 7 SIP Inspection Policy 3-73
Configuring a Layer 7 SIP Inspection Class Map 3-74
Creating a Layer 7 SIP Inspection Class Map 3-74
Configuring a DNS Parameter Map  3-106
  Defining a Description of the DNS Parameter Map  3-106
  Configuring a DNS Query Timeout  3-107
  Associating a DNS Parameter Map with a Layer 3 and Layer 4 Policy Map  3-108
Configuring an HTTP Parameter Map  3-109
  Disabling Case-Sensitivity Matching  3-109
  Setting the Maximum Number of Bytes to Parse in HTTP Headers  3-110
  Setting the Maximum Number of Bytes to Parse in HTTP Content  3-110
  Associating an HTTP Parameter Map with a Layer 3 and Layer 4 Policy Map  3-111
Configuring an SCCP Parameter Map  3-112
  SCCP Inspection Configuration Considerations  3-112
  Creating an SCCP Parameter Map  3-113
  Defining a Description of the SCCP Parameter Map  3-113
  Enabling Registration Enforcement  3-114
  Setting the Maximum Message ID  3-114
  Setting the Minimum and Maximum SCCP Prefix Length  3-115
  Associating an SCCP Parameter Map with a Layer 3 and Layer 4 Policy Map  3-115
Configuring a SIP Parameter Map  3-116
  SIP Inspection Configuration Considerations  3-117
  Creating a SIP Parameter Map  3-117
  Defining a Description of the SIP Parameter Map  3-118
  Configuring a Timeout for a SIP Media Secure Port  3-118
  Enabling Instant Messaging  3-119
  Enabling Maximum Forward Field Validation  3-119
  Configuring User Agent Software Version Options  3-120
  Enabling Strict Header Validation  3-121
  Enabling Non-SIP URI Detection in SIP Messages  3-122
Enabling Logging of All Received and Transmitted SIP Packets  3-123
Associating a SIP Parameter Map with a Layer 3 and Layer 4 Policy Map  3-124

Applying a Service Policy  3-124
Examples of Application Protocol Inspection Configurations  3-126
    Layer 7 HTTP Protocol Deep Packet Inspection  3-126
    Layer 7 FTP Command Inspection  3-128
    Layer 3 and Layer 4 Application Protocol Inspection for DNS Inspection  3-129

Viewing Application Protocol Inspection Statistics and Service Policy Information  3-130
    Displaying HTTP Protocol Inspection Statistics  3-130
    Displaying Service Policy Configuration Information  3-131

CHAPTER 4  Configuring TCP/IP Normalization and IP Reassembly Parameters  4-1

    TCP Normalization Overview  4-2
    IP Normalization Overview  4-3
    TCP/IP Normalization and Termination Configuration Quick Start  4-3

Configuring a Connection Parameter Map for TCP/IP Normalization and Termination  4-5
    Creating a Connection Parameter Map for TCP/IP, UDP, and ICMP  4-7
    Defining a Description of the Connection Parameter Map  4-8
    Configuring Rate Limits for a Policy Map  4-8
    Setting the Maximum Receive or Transmit Buffer Share  4-9
    Setting a Range for the Maximum Segment Size  4-10
    Configuring ACE Behavior for a Segment That Exceeds the Maximum Segment Size  4-12
    Setting the Maximum Number of TCP SYN Retries  4-12
    Enabling Nagle’s Algorithm  4-13
    Enabling Random TCP Sequence Numbers  4-13
    Configuring How the ACE Handles Reserved Bits  4-14
Configuring the Timeout for an Embryonic Connection 4-15
Configuring the Timeout for a Half-Closed Connection 4-15
Setting the TCP Reassembly Timeout 4-16
Configuring the Connection Inactivity Timeout 4-17
Setting How the ACE Applies TCP Optimizations to Packets 4-17
Setting the Window Scale Factor 4-19
Enabling the TCP Slow Start Algorithm 4-20
Setting the ACK Delay Timer 4-20
Configuring How the ACE Handles TCP SYN Segments that Contain Data 4-21
Configuring How the ACE Handles TCP Options 4-21
Setting the Urgent Pointer Policy 4-25
Setting the Type of Service 4-26

Configuring a Traffic Policy for TCP/IP Normalization and Termination 4-26
Configuring a Layer 4 Class Map 4-26
   Defining a Class Map Description 4-28
   Specifying IP Address Match Criteria 4-28
   Defining a TCP or UDP Port Number or Port Range Match Criteria 4-29
Configuring a Layer 3 and Layer 4 Policy Map 4-31
   Associating a Layer 3 and Layer 4 Class Map with a Policy Map 4-32
   Associating a Connection Parameter Map with a Policy Map 4-33
   Associating a Layer 3 and Layer 4 Policy Map with a Service Policy 4-33

Configuring Interface Normalization Parameters 4-34
Disabling TCP Normalization on an Interface 4-34
Enabling Normalization Send Reset on an Interface 4-36
Disabling the ICMP Security Checks on an Interface 4-36
Configuring SYN-Cookie Denial-of-Service Protection 4-37
   Overview of SYN Cookie DoS Protection 4-38
   Configuration and Operational Considerations 4-39
   Configuring SYN Cookie DoS Protection on an Interface 4-40
<table>
<thead>
<tr>
<th>Configuring How the ACE Handles the Don’t Fragment Bit</th>
<th>4-41</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring How the ACE Handles IP Options</td>
<td>4-41</td>
</tr>
<tr>
<td>Setting the IP Packet TTL</td>
<td>4-42</td>
</tr>
<tr>
<td>Configuring Unicast Reverse-Path Forwarding</td>
<td>4-43</td>
</tr>
<tr>
<td>Configuring IP Fragment Reassembly Parameters</td>
<td>4-43</td>
</tr>
<tr>
<td>IP Fragment Reassembly Configuration Quick Start</td>
<td>4-44</td>
</tr>
<tr>
<td>Configuring the MTU for an Interface</td>
<td>4-45</td>
</tr>
<tr>
<td>Configuring the Maximum Number of Fragments in a Packet</td>
<td>4-46</td>
</tr>
<tr>
<td>Configuring the Minimum Fragment Size for Reassembly</td>
<td>4-46</td>
</tr>
<tr>
<td>Configuring an IP Reassembly Timeout</td>
<td>4-47</td>
</tr>
<tr>
<td>Configuring the Switch Mode Feature</td>
<td>4-47</td>
</tr>
<tr>
<td>Example of a TCP/IP Normalization and IP Reassembly Configuration</td>
<td>4-49</td>
</tr>
<tr>
<td>Displaying Configurations and Statistics for TCP/IP and UDP Connections, IP Reassembly, and SYN Cookie</td>
<td>4-51</td>
</tr>
<tr>
<td>Displaying TCP/IP and UDP Connection Configurations</td>
<td>4-51</td>
</tr>
<tr>
<td>Displaying a Connection Parameter Map</td>
<td>4-52</td>
</tr>
<tr>
<td>Displaying TCP/IP and UDP Connection Statistics</td>
<td>4-55</td>
</tr>
<tr>
<td>Displaying Global Context Connection Statistics</td>
<td>4-57</td>
</tr>
<tr>
<td>Displaying IP Statistics</td>
<td>4-58</td>
</tr>
<tr>
<td>Displaying IP Traffic Information</td>
<td>4-58</td>
</tr>
<tr>
<td>Displaying IP Fragmentation and Reassembly Statistics</td>
<td>4-61</td>
</tr>
<tr>
<td>Displaying TCP Statistics</td>
<td>4-62</td>
</tr>
<tr>
<td>Displaying UDP Statistics</td>
<td>4-63</td>
</tr>
<tr>
<td>Displaying Service Policy Statistics</td>
<td>4-64</td>
</tr>
<tr>
<td>Displaying SYN Cookie Statistics</td>
<td>4-65</td>
</tr>
<tr>
<td>Clearing TCP/IP and UDP Connections and Statistics</td>
<td>4-66</td>
</tr>
<tr>
<td>Clearing Connections</td>
<td>4-67</td>
</tr>
<tr>
<td>Clearing Connection Statistics</td>
<td>4-67</td>
</tr>
<tr>
<td>Clearing IP, TCP, and UDP Statistics</td>
<td>4-68</td>
</tr>
<tr>
<td>Clearing IP Statistics</td>
<td>4-68</td>
</tr>
</tbody>
</table>
Contents

Clearing TCP Statistics  4-68
Clearing UDP Statistics  4-69
Clearing IP Fragmentation and Reassembly Statistics  4-69
Clearing SYN Cookie Statistics  4-70

CHAPTER 5

Configuring Network Address Translation  5-1

Network Address Translation Overview  5-1
  Dynamic NAT  5-3
  Dynamic PAT  5-4
  Server Farm-Based Dynamic NAT  5-5
  Static NAT  5-6
  Static Port Redirection  5-7
  Maximum Number of NAT Commands  5-7
  Global Address Guidelines  5-7

Configuring an Idle Timeout for NAT  5-8

Configuring Dynamic NAT and PAT  5-9
  Dynamic NAT and PAT Configuration Quick Start  5-9
  Configuring an ACL  5-11
  Configuring Interfaces for Dynamic NAT and PAT  5-12
  Creating a Global IP Address Pool for NAT  5-12
  Configuring a Class Map  5-15
  Configuring a Class Map for Passive FTP  5-16
  Configuring a Policy Map  5-16
  Configuring Dynamic NAT and PAT as a Layer 3 and Layer 4 Policy-Map Action  5-17
  Applying the Dynamic NAT and PAT Policy Map to an Interface Using a Service Policy  5-18

Configuring Server Farm-Based Dynamic NAT  5-20
  Server Farm-Based Dynamic NAT Configuration Quick Start  5-20
  Configuring an ACL for Server Farm-Based Dynamic NAT  5-24
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring Interfaces for Server Farm-Based Dynamic NAT</td>
<td>5-24</td>
</tr>
<tr>
<td>Creating a Global IP Address Pool for Dynamic NAT</td>
<td>5-24</td>
</tr>
<tr>
<td>Configuring Real Servers and a Server Farm</td>
<td>5-26</td>
</tr>
<tr>
<td>Configuring a Layer 7 Load-Balancing Class Map for Server Farm-Based Dynamic NAT</td>
<td>5-26</td>
</tr>
<tr>
<td>Configuring a Layer 7 Load-Balancing Policy Map for Server Farm-Based Dynamic NAT</td>
<td>5-27</td>
</tr>
<tr>
<td>Configuring Server Farm-Based Dynamic NAT as a Layer 7 Policy Action</td>
<td>5-28</td>
</tr>
<tr>
<td>Configuring a Layer 3 and Layer 4 Class Map for Server Farm-Based Dynamic NAT</td>
<td>5-29</td>
</tr>
<tr>
<td>Configuring a Layer 3 and Layer 4 Policy Map for Server Farm-Based Dynamic NAT</td>
<td>5-30</td>
</tr>
<tr>
<td>Applying the Layer 3 and Layer 4 Policy Map to an Interface Using a Service Policy</td>
<td>5-31</td>
</tr>
<tr>
<td>Configuring Static NAT and Static Port Redirection</td>
<td>5-32</td>
</tr>
<tr>
<td>Static NAT Configuration Quick Start</td>
<td>5-32</td>
</tr>
<tr>
<td>Configuring an ACL for Static NAT and Static Port Redirection</td>
<td>5-34</td>
</tr>
<tr>
<td>Configuring Interfaces for Static NAT and Static Port Redirection</td>
<td>5-35</td>
</tr>
<tr>
<td>Configuring a Class Map</td>
<td>5-35</td>
</tr>
<tr>
<td>Configuring a Policy Map</td>
<td>5-36</td>
</tr>
<tr>
<td>Configuring Static NAT and Static Port Redirection as a Policy Action</td>
<td>5-37</td>
</tr>
<tr>
<td>Applying the Static NAT and Static Port Redirection Policy Map to an Interface Using a Service Policy</td>
<td>5-39</td>
</tr>
<tr>
<td>Configuring Static NAT Overwrite</td>
<td>5-40</td>
</tr>
<tr>
<td>Displaying NAT Configurations and Statistics</td>
<td>5-43</td>
</tr>
<tr>
<td>Displaying NAT and PAT Configurations</td>
<td>5-43</td>
</tr>
<tr>
<td>Displaying IP Address and Port Translations</td>
<td>5-43</td>
</tr>
<tr>
<td>Dynamic NAT Example</td>
<td>5-44</td>
</tr>
<tr>
<td>Dynamic PAT Example</td>
<td>5-45</td>
</tr>
<tr>
<td>Static NAT Example</td>
<td>5-46</td>
</tr>
<tr>
<td>Static Port Redirection (Static PAT) Example</td>
<td>5-46</td>
</tr>
</tbody>
</table>
Clearing Xlates  5-47

NAT Configuration Examples  5-48
  Dynamic NAT and PAT (SNAT) Configuration Example  5-48
  Server Farm-Based Dynamic NAT (SNAT) Configuration Example  5-49
  Static Port Redirection (DNAT) Configuration Example  5-50
  SNAT with Cookie Load Balancing Example  5-51
Preface

This guide describes how to configure the security features of the Cisco Application Control Engine (ACE) module in a Catalyst 6500 series switch or a Cisco 7600 series router, hereinafter referred to as the switch or router, respectively.

This guide describes how to perform the following ACE security configuration tasks:

- Security 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 IP fragmentation
- Network Address Translation (NAT)

You can configure the ACE by using the following interfaces:

- The command-line interface (CLI), a line-oriented user interface that provides commands for configuring, managing, and monitoring the ACE.
- Device Manager graphic user interface (GUI), a Web browser-based GUI interface that provides a graphical user interface for configuring, managing, and monitoring the ACE.

This preface contains the following major sections:

- Audience
- How to Use This Guide
- Related Documentation
Symbols and 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

How to Use This Guide

This guide is organized as follows:

| Chapter |
|---------------------|-------------------------------------------------|
| Chapter 1, Configuring Security Access Control Lists | Describes how to configure security access control lists (ACLs) on your ACE. ACLs provide basic security for your network by filtering traffic and controlling network connections. |
| Chapter 2, Configuring Authentication and Accounting Services | Describes how to configure the ACE to perform user authentication and accounting (AAA) services to provide a higher level of security for users accessing the ACE. |
| Chapter 3, Configuring Application Protocol Inspection | Describes how to configure application protocol inspection for the ACE. |
Preface

Related Documentation

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

<table>
<thead>
<tr>
<th>Document Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release Note for the Cisco Application Control Engine Module</td>
<td>Provides information about operating considerations, caveats, and command-line interface (CLI) commands for the ACE.</td>
</tr>
<tr>
<td>Cisco Application Control Engine Module Hardware Installation Note</td>
<td>Provides information for installing the ACE into the Catalyst 6500 series switch or a Cisco 7600 series router.</td>
</tr>
<tr>
<td>Cisco Application Control Engine Module Getting Started Guide</td>
<td>Describes how to perform the initial setup and configuration tasks for the ACE.</td>
</tr>
<tr>
<td><strong>Document Title</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------</td>
</tr>
</tbody>
</table>
| *Cisco Application Control Engine Module Administration Guide* | 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 Application Control Engine Module Virtualization Configuration Guide* | Describes how to operate your ACE in a single context or in multiple contexts. |
| *Cisco Application Control Engine Module Routing and Bridging Configuration Guide* | Describes how to perform the following routing and bridging tasks on the ACE:  
  - Configuring VLAN interfaces  
  - Configuring routing  
  - Configuring bridging  
  - Configuring Dynamic Host Configuration Protocol (DHCP) |
<table>
<thead>
<tr>
<th>Document Title</th>
<th>Description</th>
</tr>
</thead>
</table>
| Cisco Application Control Engine Module Server Load-Balancing Configuration Guide | Describes how to configure the following server load-balancing features 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                                                                                                                                                                                     |
| Cisco Application Control Engine Module SSL Configuration Guide               | Describes how to configure the following Secure Sockets Layer (SSL) features on the ACE:  
  - SSL certificates and keys  
  - SSL initiation  
  - SSL termination  
  - End-to-end SSL                                                                                                                                                                              |
| Cisco Application Control Engine Module System Message Guide                  | 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.                                                            |
| Cisco Application Control Engine Module Command Reference                     | Provides an alphabetical list and descriptions of all CLI commands by mode, including syntax, options, and related commands.                                                                                  |
| 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-configuration or startup-configuration files to the ACE.                                         |
## Symbols and Conventions

This publication uses the following conventions:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong> font</td>
<td>Commands, command options, and keywords are in <strong>boldface</strong>. Bold text also indicates a command in a paragraph.</td>
</tr>
<tr>
<td><em>italic</em> font</td>
<td>Arguments for which you supply values are in <em>italics</em>. Italic text also indicates the first occurrence of a new term, book title, emphasized text.</td>
</tr>
<tr>
<td><code>{ }</code></td>
<td>Encloses required arguments and keywords.</td>
</tr>
<tr>
<td><code>[ ]</code></td>
<td>Encloses optional arguments and keywords.</td>
</tr>
<tr>
<td>`{x</td>
<td>y</td>
</tr>
<tr>
<td>`[x</td>
<td>y</td>
</tr>
<tr>
<td>string</td>
<td>A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.</td>
</tr>
<tr>
<td><strong>screen</strong> font</td>
<td>Terminal sessions and information the system displays are in <strong>screen</strong> font.</td>
</tr>
</tbody>
</table>
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.

For additional information about CLI syntax formatting, see the *Cisco Application Control Engine Module Command Reference*. 

---

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface screen font</strong></td>
<td>Information you must enter in a command line is in <strong>boldface screen font</strong>.</td>
</tr>
<tr>
<td><strong>italic screen font</strong></td>
<td>Arguments for which you supply values are in <strong>italic screen font</strong>.</td>
</tr>
<tr>
<td>^</td>
<td>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.</td>
</tr>
<tr>
<td>&lt; &gt;</td>
<td>Nonprinting characters, such as passwords are in angle brackets.</td>
</tr>
</tbody>
</table>
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:

This chapter describes how to configure security access control lists (ACLs) on your Cisco Application Control Engine (ACE) module. ACLs provide basic security for your network by filtering traffic and controlling network connections. This chapter contains the following major sections:

- ACL Overview
- ACL Configuration Quick Start
- Configuring ACLs
- Simplifying Access Control Lists with Object Groups
- Applying an ACL to an Interface
- Applying an ACL Globally to All Interfaces in a Context
- Filtering Traffic with an ACL
- ACL Configuration Examples
- Displaying ACL Configuration Information and Statistics
- Clearing ACL Statistics
ACL Overview

An ACL consists of a series of statements called ACL entries that define the network traffic profile. Each entry permits or denies network traffic (inbound and outbound) to the parts of your network specified in the entry. Each entry also contains a filter element that is based on criteria such as the source address, the destination address, the protocol, and protocol-specific parameters such as ports and so on.

An implicit deny-all entry exists at the end of each ACL, so you must configure an ACL on each interface that you want to permit connections. Otherwise, the ACE denies all traffic on the interface.

ACLs allow you to control network connection setups rather than processing each packet. Such ACLs are commonly referred to as security ACLs.

You can configure ACLs as parts of other features (for example, security, Network Address Translation (NAT), server load balancing (SLB), and so on). The ACE merges these individual ACLs into one large ACL called a merged ACL. The ACL compiler then parses the merged ACL and generates the ACL lookup mechanisms. A match on this merged ACL can result in multiple actions.

For example, one use of ACLs could be to permit all e-mail traffic on a VLAN, but block Telnet traffic. You can also use ACLs to allow one client to access a part of the network and prevent another client from accessing that same area.

When configuring ACLs, you must apply an ACL to an interface to control traffic on that interface. Applying an ACL on an interface assigns the ACL and its entries to that interface.

You can apply only one extended ACL to each direction (inbound or outbound) of an interface. You can also apply the same ACL on multiple interfaces. You can apply EtherType ACLs only in the inbound direction and only on Layer 2 interfaces.

This section contains the following topics:

- ACL Types and Uses
- ACL Guidelines
ACL Types and Uses

You can configure the following two types of ACLs on the ACE:

- Extended—Control network access for IP traffic
- EtherType—Control network access for non-IP traffic

Note

The ACE does not explicitly support standard ACLs. To configure a standard ACL, specify the destination address as any and do not specify ports in an extended ACL. For details about configuring an extended ACL, see the “Configuring an Extended ACL” section.

ACL Guidelines

This section describes the guidelines to observe when you configure and use ACLs in your network.

Note

If you configure an ACL on an interface to block certain traffic and a management policy on that same interface allows that traffic, the management policy overrides the ACL and the ACE allows the traffic.

This section contains the following topics:

- ACL Entry Order
- ACL Implicit Deny
- Maximum Number of ACLs and ACL Entries

ACL Entry Order

An ACL consists of one or more entries. Depending on the ACL type, you can specify the source and destination addresses, the protocol, the ports (for TCP or UDP), the ICMP type, the ICMP code, or the EtherType as the match criteria. By default, the ACE appends each ACL entry at the end of the ACL. You can also specify the location of each entry within an ACL.
The order of the entries is important. When the ACE decides whether to accept or refuse a connection, the ACE tests the packet against each ACL entry in the order in which the entries are listed. After it finds a match, the ACE does not check any more entries. For example, if you create an entry at the beginning of an ACL that explicitly permits all traffic, the ACE does not check any other statements in the ACL.

**ACL Implicit Deny**

All ACLs have an implicit deny entry at the end of the ACL, so, unless you explicitly permit it, traffic cannot pass. For example, if you want to allow all users to access a network through the ACE except for those users with particular IP addresses, then you must deny the particular IP addresses in one entry and permit all other IP addresses in another entry.

**Maximum Number of ACLs and ACL Entries**

The ACE supports a maximum of 8192 unique ACLs and 64,000 ACL entries. Some ACLs use more memory than others, such as an ACL that uses large port number ranges or overlapping networks (for example, one entry specifies 10.0.0.0/8 and another entry specifies 10.1.1.0/24). Depending on the type of ACL, the actual limit that the ACE can support may be less than 64,000 entries.

If you use object groups in ACL entries, you enter fewer actual ACL entries, but the same number of expanded ACL entries as you did when you entered entries without object groups. Expanded ACL entries count toward the system limit. To view the number of expanded ACL entries in an ACL, use the `show access-list name` command.

If you exceed the memory limitations of the ACE, the module generates a syslog message and increments the Download Failures counter in the output of the `show interface vlan number` command. The configuration remains in the running-config file and the interface stays enabled. The ACL entries stay the same as they were before the failing configuration was attempted.

For example, if you add a new ACL with ten entries, but the addition of the sixth entry fails because the ACE runs out of memory, the ACE removes the five entries that you successfully entered.
ACL Configuration Quick Start

Table 1-1 provides a quick overview of the steps required to configure ACLs. Each step includes the CLI command or a reference to the procedure required to complete the task. For a complete description of each feature and all the options associated with the CLI commands, see the sections following Table 1-1.

Table 1-1    ACL Configuration Quick Start

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, change to the correct context.</td>
</tr>
<tr>
<td>host1/Admin# changeto C1</td>
</tr>
<tr>
<td>host1/C1#</td>
</tr>
<tr>
<td>The rest of the examples in this table use the Admin context, unless otherwise specified. For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.</td>
</tr>
<tr>
<td>2. Enter configuration mode.</td>
</tr>
<tr>
<td>host1/Admin# config</td>
</tr>
<tr>
<td>host1/Admin(config)#</td>
</tr>
<tr>
<td>3. Create an ACL.</td>
</tr>
<tr>
<td>host1/Admin(config)# access-list INBOUND extended deny ip 192.168.12.0 255.255.255.0 any</td>
</tr>
<tr>
<td>4. As required by your application, add entries to the ACL using the same access list name. For example, enter:</td>
</tr>
<tr>
<td>host1/Admin(config)# access-list INBOUND extended permit ip any any</td>
</tr>
<tr>
<td>5. Apply the ACL to an individual interface.</td>
</tr>
<tr>
<td>host1/Admin(config)# interface vlan 10</td>
</tr>
<tr>
<td>host1/Admin(config-if)# access-group input INBOUND</td>
</tr>
<tr>
<td>6. Alternatively, you can apply an ACL globally to all interfaces in a context.</td>
</tr>
<tr>
<td>host1/Admin(config)# access-group input INBOUND</td>
</tr>
</tbody>
</table>
Configuring ACLs

This section contains the following topics:

- Configuring an Extended ACL
- Configuring Comments in an Extended ACL
- Configuring an EtherType ACL
- Resequencing Entries

Configuring an Extended ACL

An extended ACL allows you to specify both the source and the destination IP addresses of traffic as well as the following parameters:

- Protocol
- TCP or UDP ports
- ICMP types and codes

You can specify these parameters directly when you use the `access-list` command or you can use object groups for each parameter. For more information about object groups, see the “Simplifying Access Control Lists with Object Groups” section.

For TCP, UDP, and ICMP connections, you do not need to apply an ACL on the destination interface to allow returning traffic, because the ACE allows all returning traffic for established connections.

---

Table 1-1  ACL Configuration Quick Start (continued)

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. (Optional) Save your configuration changes to flash memory.</td>
</tr>
<tr>
<td>host1/Admin(config)# exit</td>
</tr>
<tr>
<td>host1/Admin# copy running-config startup-config</td>
</tr>
<tr>
<td>8. Display and verify the ACL configuration information.</td>
</tr>
<tr>
<td>host1/Admin# show running-config access-list</td>
</tr>
</tbody>
</table>
Note
The ACE does not explicitly support standard ACLs. To configure a standard ACL, specify the destination address as **any** and do not specify the ports in an extended ACL.

Tip
Enter the ACL name in uppercase letters so that the name is easy to see in the configuration. You may want to name the ACL for the interface (for example, INBOUND) or for the purpose (for example, NO_NAT or VPN).

To create an extended ACL, use the `access-list extended` command in configuration mode. There are two major types of extended ACLs:

- Non-ICMP ACLs
- ICMP ACLs

You can permit or deny network connections based on the IP protocol, source and destination IP addresses, and TCP or UDP ports. To syntax of a non-ICMP extended ACL is as follows:

```
access-list name [line number] extended {deny | permit} 
{protocol {any | host src_ip_address | src_ip_address netmask | object-group net_obj_grp_name} [operator port1 [port2]] {any | host dest_ip_address | dest_ip_address netmask | object-group net_obj_grp_name} [operator port3 [port4]]}
| {object-group service_obj_grp_name} {any | host src_ip_address | src_ip_address netmask | object-group net_obj_grp_name} {any | host dest_ip_address | dest_ip_address netmask | object-group net_obj_grp_name}
```

You can also permit or deny network connections based on the ICMP type (for example, echo, echo-reply, unreachable, and so on). To syntax of an ICMP extended ACL is as follows:

```
access-list name [line number] extended {deny | permit} 
{icmp {any | host src_ip_address | src_ip_address netmask | object_group net_obj_grp_name} {any | host dest_ip_address | dest_ip_address netmask | object_group net_obj_grp_name} [icmp_type [code operator code1 [code2]]]}
| {object-group service_obj_grp_name} {any | host src_ip_address | src_ip_address netmask | object_group net_obj_grp_name} {any | host dest_ip_address | dest_ip_address netmask | object-group net_obj_grp_name}
```
The keywords, options, and arguments are as follows:

- **name**—Unique identifier of the ACL. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

- **line number**—(Optional) Specifies the line number position where you want the entry that you are configuring to appear in the ACL. The position of an entry affects the lookup order of the entries in an ACL. If you do not configure the line number of an entry, the ACE applies a default increment and a line number to the entry and appends it at the end of the ACL.

- **extended**—Specifies an extended ACL. Extended ACLs allow you to specify the destination IP address and subnet mask and other parameters not available with a standard ACL.

- **deny**—Blocks connections on the assigned interface.

- **permit**—Allows connections on the assigned interface.

- **protocol**—Name or number of an IP protocol. Enter a protocol name or an integer from 0 to 255 that represents an IP protocol number from Table 1-2.

<table>
<thead>
<tr>
<th>Protocol Name</th>
<th>Protocol Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ah</td>
<td>51</td>
<td>Authentication Header</td>
</tr>
<tr>
<td>eigrp</td>
<td>88</td>
<td>Enhanced IGRP</td>
</tr>
<tr>
<td>esp</td>
<td>50</td>
<td>Encapsulated Security Payload</td>
</tr>
<tr>
<td>gre</td>
<td>47</td>
<td>Generic Routing Encapsulation</td>
</tr>
<tr>
<td>icmp</td>
<td>1</td>
<td>Internet Control Message Protocol</td>
</tr>
<tr>
<td>igmp</td>
<td>2</td>
<td>Internet Group Management Protocol</td>
</tr>
<tr>
<td>ip</td>
<td>any</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>ip-in-ip</td>
<td>4</td>
<td>IP-in-IP Layer 3 Tunneling Protocol</td>
</tr>
<tr>
<td>ospf</td>
<td>89</td>
<td>Open Shortest Path First</td>
</tr>
<tr>
<td>pim</td>
<td>103</td>
<td>Protocol Independent Multicast</td>
</tr>
<tr>
<td>tcp</td>
<td>6</td>
<td>Transmission Control Protocol</td>
</tr>
</tbody>
</table>
Table 1-2  Supported Protocol Keywords and Numbers

<table>
<thead>
<tr>
<th>Protocol Name</th>
<th>Protocol Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp-udp</td>
<td>6 and 17</td>
<td>TCP and UDP</td>
</tr>
<tr>
<td>udp</td>
<td>17</td>
<td>User Datagram Protocol</td>
</tr>
</tbody>
</table>

- **any**—Specifies network traffic from any source.
- **host src_ip_address**—Specifies the IP address of the host from which the network traffic originates. Use this keyword and argument to specify the network traffic from a single IP address.
- **src_ip_address netmask**—Traffic from a source defined by the IP address and the network mask. Use these arguments to specify network traffic from a range of source IP addresses.

**Note**  For the source IP address netmask, the ACE supports only standard subnet mask entries in an ACL. Wildcard entries (for example, 0.0.0.15) and non-standard subnet masks are not supported.

- **object-group net_obj_grp_name**—Specifies the identifier of an existing network object group. For details, see the “Simplifying Access Control Lists with Object Groups” section.
- **operator**—(Optional) Operand used to compare source and destination port numbers for TCP, TCP-UDP, and UDP protocols. The operators are as follows:
  - **eq**—Equal to.
  - **gt**—Greater than.
  - **lt**—Less than.
  - **neq**—Not equal to.
  - **range**—An inclusive range of port values. If you enter this operator, enter a second port number value to define the upper limit of the range.
- **port1 [port2]**—TCP or UDP source port name or number from which you permit or deny services access. Enter an integer from 0 to 65535. To enter an inclusive range of ports, enter two port numbers. **port2** must be greater than
or equal to \textit{port1}. See \textit{Table 1-3} for a list of well-known TCP port names and numbers and \textit{Table 1-4} for a list of well-known UDP port names and numbers.

\begin{table}[h]
\centering
\begin{tabular}{|l|l|l|}
\hline
\textbf{Keyword} & \textbf{Port Number} & \textbf{Description} \\
\hline
\texttt{aol} & 5190 & America-Online \\
\hline
\texttt{bgp} & 179 & Border Gateway Protocol \\
\hline
\texttt{chargen} & 19 & Character Generator \\
\hline
\texttt{citrix-ica} & 1494 & Citrix Independent Computing Architecture Protocol \\
\hline
\texttt{cmd} & 514 & Same as exec, with automatic authentication \\
\hline
\texttt{ctiqbe} & 2748 & Computer Telephony Interface Quick Buffer Encoding \\
\hline
\texttt{daytime} & 13 & Daytime \\
\hline
\texttt{discard} & 9 & Discard \\
\hline
\texttt{domain} & 53 & Domain Name System \\
\hline
\texttt{echo} & 7 & Echo \\
\hline
\texttt{exec} & 512 & Exec (RSH) \\
\hline
\texttt{finger} & 79 & Finger \\
\hline
\texttt{ftp} & 21 & File Transfer Protocol \\
\hline
\texttt{ftp-data} & 20 & FTP data connections \\
\hline
\texttt{gopher} & 70 & Gopher \\
\hline
\texttt{h323} & 1720 & H.323 call signaling \\
\hline
\texttt{hostname} & 101 & NIC hostname server \\
\hline
\texttt{http} & 80 & Hypertext Transfer Protocol \\
\hline
\texttt{https} & 443 & HTTP over TLS/SSL \\
\hline
\texttt{ident} & 113 & Ident Protocol \\
\hline
\texttt{imap4} & 143 & Internet Message Access Protocol, version 4 \\
\hline
\end{tabular}
\end{table}
### Table 1-3  Well-Known TCP Port Numbers and Keywords (continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>irc</td>
<td>194</td>
<td>Internet Relay Chat</td>
</tr>
<tr>
<td>kerberos</td>
<td>88</td>
<td>Kerberos</td>
</tr>
<tr>
<td>klogin</td>
<td>543</td>
<td>Kerberos Login</td>
</tr>
<tr>
<td>kshell</td>
<td>544</td>
<td>Kerberos Shell</td>
</tr>
<tr>
<td>ldap</td>
<td>389</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>ldaps</td>
<td>636</td>
<td>LDAP over TLS/SSL</td>
</tr>
<tr>
<td>login</td>
<td>513</td>
<td>Login (rlogin)</td>
</tr>
<tr>
<td>lotusnotes</td>
<td>1352</td>
<td>IBM Lotus Notes</td>
</tr>
<tr>
<td>lpd</td>
<td>515</td>
<td>Printer Service</td>
</tr>
<tr>
<td>matip-a</td>
<td>350</td>
<td>Mapping of Airline Traffic over Internet Protocol Type A</td>
</tr>
<tr>
<td>netbios-ssn</td>
<td>139</td>
<td>NetBIOS Session Service</td>
</tr>
<tr>
<td>nntp</td>
<td>119</td>
<td>Network News Transport Protocol</td>
</tr>
<tr>
<td>pcanywhere-data</td>
<td>5631</td>
<td>PC Anywhere data</td>
</tr>
<tr>
<td>pim-auto-rp</td>
<td>496</td>
<td>PIM Auto-RP</td>
</tr>
<tr>
<td>pop2</td>
<td>109</td>
<td>Post Office Protocol v2</td>
</tr>
<tr>
<td>pop3</td>
<td>110</td>
<td>Post Office Protocol v3</td>
</tr>
<tr>
<td>pptp</td>
<td>1723</td>
<td>Point-to-Point Tunneling Protocol, RFC 2637</td>
</tr>
<tr>
<td>rtsp</td>
<td>554</td>
<td>Real-Time Streaming Protocol</td>
</tr>
<tr>
<td>sip</td>
<td>5060</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>skinny</td>
<td>2000</td>
<td>Cisco Skinny Client Control Protocol (SCCP)</td>
</tr>
<tr>
<td>smtp</td>
<td>25</td>
<td>Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>sqlnet</td>
<td>1521</td>
<td>Structured Query Language Network</td>
</tr>
<tr>
<td>ssh</td>
<td>22</td>
<td>Secure Shell</td>
</tr>
<tr>
<td>sunrpc</td>
<td>111</td>
<td>Sun Remote Procedure Call</td>
</tr>
</tbody>
</table>
### Table 1-3 Well-Known TCP Port Numbers and Keywords (continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tacacs</td>
<td>49</td>
<td>Terminal Access Controller Access Control System</td>
</tr>
<tr>
<td>talk</td>
<td>517</td>
<td>Talk</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>Telnet</td>
</tr>
<tr>
<td>time</td>
<td>37</td>
<td>Time</td>
</tr>
<tr>
<td>uucp</td>
<td>540</td>
<td>UNIX-to-UNIX Copy Program</td>
</tr>
<tr>
<td>whois</td>
<td>43</td>
<td>Nicname</td>
</tr>
<tr>
<td>www</td>
<td>80</td>
<td>World Wide Web (HTTP)</td>
</tr>
</tbody>
</table>

### Table 1-4 Well-Known UDP Key Words and Port Numbers

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>biff</td>
<td>512</td>
<td>Mail notification</td>
</tr>
<tr>
<td>bootpc</td>
<td>68</td>
<td>Bootstrap Protocol client</td>
</tr>
<tr>
<td>bootps</td>
<td>67</td>
<td>Bootstrap Protocol server</td>
</tr>
<tr>
<td>discard</td>
<td>9</td>
<td>Discard</td>
</tr>
<tr>
<td>dnsix</td>
<td>195</td>
<td>DNSIX Security protocol auditing (dn6-nlm-aud)</td>
</tr>
<tr>
<td>domain</td>
<td>53</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>echo</td>
<td>7</td>
<td>Echo</td>
</tr>
<tr>
<td>isakmp</td>
<td>500</td>
<td>Internet Security Association Key Management Protocol</td>
</tr>
<tr>
<td>kerberos</td>
<td>88</td>
<td>Kerberos</td>
</tr>
<tr>
<td>mobile-ip</td>
<td>434</td>
<td>Mobile IP registration</td>
</tr>
<tr>
<td>nameserver</td>
<td>42</td>
<td>Host Name Server</td>
</tr>
<tr>
<td>netbios-dgm</td>
<td>138</td>
<td>NetBIOS datagram service</td>
</tr>
<tr>
<td>netbios-ns</td>
<td>137</td>
<td>NetBIOS name service</td>
</tr>
</tbody>
</table>
Table 1-4  Well-Known UDP Key Words and Port Numbers (continued)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netbios-ssn</td>
<td>139</td>
<td>NetBIOS Session Service</td>
</tr>
<tr>
<td>ntp</td>
<td>123</td>
<td>Network Time Protocol</td>
</tr>
<tr>
<td>pcanywhere-status</td>
<td>5632</td>
<td>PC Anywhere status</td>
</tr>
<tr>
<td>radius</td>
<td>1812</td>
<td>Remote Authentication Dial-in User Service</td>
</tr>
<tr>
<td>radius-acct</td>
<td>1813</td>
<td>RADIUS Accounting</td>
</tr>
<tr>
<td>rip</td>
<td>520</td>
<td>Routing Information Protocol</td>
</tr>
<tr>
<td>snmp</td>
<td>161</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>snmptrap</td>
<td>162</td>
<td>SNMP Traps</td>
</tr>
<tr>
<td>sunrpc</td>
<td>111</td>
<td>Sun Remote Procedure Call</td>
</tr>
<tr>
<td>syslog</td>
<td>514</td>
<td>System Logger</td>
</tr>
<tr>
<td>tacacs</td>
<td>49</td>
<td>Terminal Access Controller Access Control System</td>
</tr>
<tr>
<td>talk</td>
<td>517</td>
<td>Talk</td>
</tr>
<tr>
<td>tftp</td>
<td>69</td>
<td>Trivial File Transfer Protocol</td>
</tr>
<tr>
<td>time</td>
<td>37</td>
<td>Time</td>
</tr>
<tr>
<td>who</td>
<td>513</td>
<td>Who service (rwho)</td>
</tr>
<tr>
<td>wsp</td>
<td>9200</td>
<td>Connectionless Wireless Session Protocol</td>
</tr>
<tr>
<td>wsp-wtls</td>
<td>9202</td>
<td>Secure Connectionless WSP</td>
</tr>
<tr>
<td>wsp-wtp</td>
<td>9201</td>
<td>Connection-based WSP</td>
</tr>
<tr>
<td>wsp-wtp-wtls</td>
<td>9203</td>
<td>Secure Connection-based WSP</td>
</tr>
<tr>
<td>xdmcp</td>
<td>177</td>
<td>X Display Manager Control Protocol</td>
</tr>
</tbody>
</table>

- dest_ip_address netmask—IP address of the network or host to which the packet is being sent and the network mask bits to be applied to the destination IP address. Use these arguments to specify a range of destination IP addresses.
For the destination IP address netmask, the ACE supports only standard subnet mask entries in an ACL. Wildcard entries (for example, 0.0.0.15) and non-standard subnet masks are not supported.

- **any**—Specifies the network traffic that goes to any destination.
- **host dest_address**—Specifies the IP address and subnet mask of the destination of the packets in a flow. Use this keyword and argument to specify the network traffic destined to a single IP address.
- **operator**—(Optional) Operand used to compare source and destination port numbers for TCP and UDP protocols. The operators are as follows:
  - **lt**—Less than.
  - **gt**—Greater than.
  - **eq**—Equal to.
  - **neq**—Not equal to.
  - **range**—Inclusive range of port values. If you enter this operator, enter a second port number value to define the upper limit of the range.
- **port3 [port4]**—TCP or UDP destination port name or number to which you permit or deny services access. To enter an optional inclusive range of ports, enter two port numbers. *port4* must be greater than or equal to *port3*. See Table 1-3 for a list of well-known ports.
- **object-group service_obj_grp_name**—(Optional) Specifies the identifier of an existing service object group. For details, see the “Simplifying Access Control Lists with Object Groups” section.
- **icmp_type**—(Optional) Type of ICMP messaging. Enter either an integer that corresponds to the ICMP code number or an ICMP type as described in Table 1-5.

### Table 1-5 ICMP Types

<table>
<thead>
<tr>
<th>ICMP Code Number</th>
<th>ICMP Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>echo-reply</td>
</tr>
<tr>
<td>3</td>
<td>unreachable</td>
</tr>
<tr>
<td>4</td>
<td>source-quench</td>
</tr>
</tbody>
</table>
### Table 1-5  ICMP Types (continued)

<table>
<thead>
<tr>
<th>ICMP Code Number</th>
<th>ICMP Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>redirect</td>
</tr>
<tr>
<td>6</td>
<td>alternate-address</td>
</tr>
<tr>
<td>8</td>
<td>echo</td>
</tr>
<tr>
<td>9</td>
<td>router-advertisement</td>
</tr>
<tr>
<td>10</td>
<td>router-solicitation</td>
</tr>
<tr>
<td>11</td>
<td>time-exceeded</td>
</tr>
<tr>
<td>12</td>
<td>parameter-problem</td>
</tr>
<tr>
<td>13</td>
<td>timestamp-request</td>
</tr>
<tr>
<td>14</td>
<td>timestamp-reply</td>
</tr>
<tr>
<td>15</td>
<td>information-request</td>
</tr>
<tr>
<td>16</td>
<td>information-reply</td>
</tr>
<tr>
<td>17</td>
<td>mask-request</td>
</tr>
<tr>
<td>18</td>
<td>mask-reply</td>
</tr>
<tr>
<td>30</td>
<td>traceroute</td>
</tr>
<tr>
<td>31</td>
<td>conversion-error</td>
</tr>
<tr>
<td>32</td>
<td>mobile-redirect</td>
</tr>
</tbody>
</table>

- **code**—(Optional) Specifies that a numeric operator and ICMP code follows.
- **operator**—Operator that the ACE applies to the ICMP code that follows. Enter one of the following operators:
  - *lt*—Less than.
  - *gt*—Greater than.
  - *eq*—Equal to.
  - *neq*—Not equal to.
  - **range**—Inclusive range of ICMP code values. When you use this operator, specify two code numbers to define the range.
• code1, code2—ICMP code number that corresponds to an ICMP type. See Table 1-5. If you entered the range operator, enter a second ICMP code value to define the upper limit of the range.

For security reasons, the ACE does not allow pings from an interface on a VLAN on one side of the ACE through the module to an interface on a different VLAN on the other side of the module. For example, a host can ping the ACE address that is on the IP subnet using the same VLAN as the host but cannot ping IP addresses configured on other VLANs on the ACE.

For example, to configure a TCP extended ACL, enter:

```
host1/Admin(config)# access-list INBOUND line 10 extended permit tcp 192.168.12.0 255.255.255.0 gt 1024 172.27.16.0 255.255.255.0 lt 4000
```

For example, to remove an entry from an extended ACL, enter:

```
host1/Admin(config)# no access-list INBOUND line 10
```

To control a ping, specify echo (8) (host to ACE).

For example, to allow an external host with IP address 192.168.12.5 to ping a host behind the ACE with an IP address of 10.0.0.5, enter:

```
host1/Admin(config)# access-list INBOUND extended permit icmp host 192.168.12.5 host 10.0.0.5 echo code eq 0
```

For example, to remove an entry from an ICMP ACL, enter:

```
host1/Admin(config)# no access-list INBOUND extended permit icmp host 192.168.12.5 echo
```

## Configuring Comments in an Extended ACL

You can add comments about an extended ACL to clarify the function of the ACL. To add a comment to an ACL, use the `access-list name remark` command in configuration mode. You can enter only one comment per ACL and the comment always appears at the beginning of the ACL. The syntax of this command is as follows:

```
access-list name remark text
```

The keywords and arguments are as follows:
• **name**—Unique identifier of the ACL. Enter an unquoted text string with a maximum of 64 alphanumeric characters.

• **remark text**—Specifies any comments that you want to include about the ACL. Comments appear at the top of the ACL. Enter an unquoted text string with a maximum of 100 alphanumeric characters. You can enter leading spaces at the beginning of the text. Trailing spaces are ignored.

For example, enter:

```
host1/Admin(config)# access-list INBOUND remark This is a remark
```

For example, to remove entry comments from an ACL, enter:

```
host1/Admin(config)# no access-list INBOUND line 200 remark
```

If you delete an ACL using the **no access-list name** command, then all the remarks are also removed.

## Configuring an EtherType ACL

You can configure an ACL that controls traffic based on its EtherType. An EtherType is a subprotocol identifier. EtherType ACLs support Ethernet V2 frames. EtherType ACLs do not support 802.3-formatted frames because they use a length field instead of a type field. The only exception is a bridge protocol data unit (BPDU), which is SNAP encapsulated. The ACE can specifically handle BPDUs.

You can permit or deny BPDUs. By default, all BPDUs are denied. The ACE receives trunk port (Cisco proprietary) BPDUs because ACE ports are trunk ports. Trunk BPDUs have VLAN information inside the payload, so the ACE modifies the payload with the outgoing VLAN if you allow BPDUs. If you configure redundancy, you must allow BPDUs on both interfaces with an EtherType ACL to avoid bridging loops. For details about configuring redundancy, see the *Cisco Application Control Engine Module Administration Guide*.

If you allow Multiprotocol Label Switching (MPLS), ensure that Label Distribution Protocol (LDP) and Tag Distribution Protocol (TDP) TCP connections are established through the ACE by configuring both MPLS routers connected to the ACE to use the IP address on the ACE interface as the router ID for LDP or TDP sessions. LDP and TDP allow MPLS routers to negotiate the labels (addresses) used to forward packets.
You can configure an EtherType ACL on a Layer 2 interface in the inbound direction only.

On Cisco IOS routers, enter the appropriate command for your protocol: LDP or TDP. The interface is the interface connected to the ACE:

```
host1/Admin(config)# mpls ldp router-id interface force
```

or

```
host1/Admin(config)# tag-switching tdp router-id interface force
```

Enter the ACL name in uppercase letters so that the name is easy to see in the configuration. You may want to name the ACL for the interface (for example, INBOUND), or for the purpose (for example, MPLS).

To configure an EtherType ACL, use the `access-list ethertype` command in configuration mode. The syntax of this command is as follows:

```
access-list name ethertype {deny | permit} {any | bpdu | ipv6 | mpls}
```

The keywords and arguments are as follows:

- **name**—Unique identifier of the ACL. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.
- **ethertype**—Name that specifies a subprotocol. Valid values are as follows:
  - **deny**—Blocks connections on the assigned interface
  - **permit**—Allows connections on the assigned interface
  - **any**—Specifies any EtherType
  - **bpdu**—Specifies a bridge protocol data unit
  - **ipv6**—Specifies Internet Protocol version 6
  - **mpls**—Specifies Multiprotocol Label Switching

Note: The ACE does not forward multiple spanning tree (MST) BPDUs.
Note: When you specify the `mpls` keyword in an EtherType ACL, the ACE denies or permits both MPLS-unicast and MPLS-multicast traffic.

For example, to configure an EtherType ACL for MPLS, enter:

```
host1/Admin(config)# access-list INBOUND ethertype permit mpls
```

To remove an entry from an EtherType ACL, enter:
```
host1/Admin(config)# no access-list INBOUND ethertype permit mpls
```

### Resequencing Entries

You can resequence the entries in an ACL with a specific starting number and interval by using the `access-list name resequence` command in configuration mode. The ability to resequence entries in an ACL is supported only for extended ACLs.

The syntax of this command is as follows:

```
access-list name resequence [number1] [number2]
```

The keywords, options, and arguments are as follows:

- `name`—Unique identifier of the ACL. Enter an unquoted text string with a maximum of 64 alphanumeric characters.
- `resequence`—Specifies the renumbering of the entries in an ACL.
- `number1`—(Optional) Number assigned to the first entry in the ACL. Enter any integer. The default is 10.
- `number2`—(Optional) Number added to each entry in the ACL after the first entry. Enter any integer. The default is 10.

For example, enter:
```
host1/Admin(config)# access-list INBOUND resequence 5 15
```
Simplifying Access Control Lists with Object Groups

This section describes how to use object groups to simplify ACL creation and maintenance. It contains the following topics:

- Overview of Object Groups
- Configuring Network Object Groups
- Configuring Service Object Groups
- Using Object Groups in an ACL
- Applying an ACL to an Interface
- Applying an ACL Globally to All Interfaces in a Context
- Filtering Traffic with an ACL

Overview of Object Groups

Object groups allow you to streamline the configuration of multiple ACL entries in an ACL. By grouping like objects together, you can use an object group in an ACL entry instead of having to enter an ACL entry for each object separately. You can create the following types of object groups:

- Network object groups
- Service object groups

For example, consider the following three object groups:

- MyServices—Includes the TCP and UDP port numbers of the service requests that are allowed access to the internal network
- TrustedHosts—Includes the host and network addresses that are allowed access to the greatest range of services and servers
- PublicServers—Includes the host addresses of servers to which the greatest access is provided

After you create these groups, you can use a single ACL entry to allow trusted hosts to make specific service requests to a group of public servers.
You can configure a maximum of 4 K object groups in an ACE. Each object group can have up to 64,000 elements. The maximum number of ACL entries in an ACE is 64,000.

The system-wide ACL entry limit of 64,000 entries applies to expanded ACL entries. An expanded ACL entry is the individually entered entry equivalent of an object-group element. If you use object groups in an ACL, you enter fewer actual ACL entries. When the ACE expands an ACL that references an object group, internally, multiple ACL entries will exist based on the number of elements present in the object group. To view the number of expanded ACL entries in an ACL, enter the `show access-list name` command. For details, see the “Displaying ACL Configuration Information and Statistics” section.

### Configuring Network Object Groups

This section describes how to configure object groups to streamline the creation of ACL entries in an ACL. It includes the following topics:

- Creating a Network Object Group
- Adding a Description to a Network Object Group
- Configuring a Network IP Address for a Network Object Group
- Configuring a Host IP Address

#### Creating a Network Object Group

To create an object group, use the `object-group` command in configuration mode. The syntax of this command is as follows:

```
object-group network name
```

The keywords and arguments are as follows:

- **network**—Specifies a group of hosts or subnet IP addresses.
- **name**—Unique identifier of the object group. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to create a network object group, enter:

```
host1/Admin(config)# object-group network NET_OBJ_GROUP1
```
Simplifying Access Control Lists with Object Groups

Chapter 1 Configuring Security Access Control Lists

To remove the network object group from the configuration, enter:

```
host1/Admin(config)# no object-group network NET_OBJ_GROUP1
```

Note

If you add new elements to an existing object group that is already in use by an entry in a large ACL, recommitting the ACL can take a long time, depending on the size of the ACL and the number of elements in the object group. In extreme cases, recommitting this ACL may cause the ACE to respond to commands slowly or even to become temporarily unresponsive. We recommend that you first remove the ACL entry that refers to the object group, make your modifications to the relevant object group, and then add the ACL entry back into the ACL.

Adding a Description to a Network Object Group

To add an optional description to a network object group, use the `description` command in object group network configuration mode. The syntax of this command is as follows:

```
description text
```

The `text` argument is an unquoted text string with a maximum of 240 alphanumeric characters.

For example, to add a description to a network object group, enter:

```
host1/Admin(config-objgrp-netw)# description intranet network object group
```

To remove a description from a network object group, enter:

```
host1/Admin(config-objgrp-netw)# no description intranet network object group
```

Configuring a Network IP Address for a Network Object Group

To associate a network IP address with a network object group, use the `ip_address` argument in object-group network configuration mode. The syntax of this command is as follows:

```
ip_address netmask
```
Simplifying Access Control Lists with Object Groups

The arguments are as follows:

- *ip_address* — IP address assigned to the network object group. Enter an IP address in dotted-decimal notation (for example, 192.168.12.15).

- *netmask* — Network mask applied to the IP address. Enter a network mask in dotted decimal notation (for example, 255.255.255.0).

For example, to add the IP address 192.168.12.15 and network mask 255.255.255.0 to a network object group, enter:

```
host1/Admin(config-objgrp-netw)# 192.168.12.15 255.255.255.0
```

Enter additional object-group IP addresses as required.

To remove an IP address from the network object group, enter:

```
host1/Admin(config-objgrp-netw)# no 192.168.12.15 255.255.255.0
```

Configuring a Host IP Address

To associate a host IP address with a network object group, use the *host* command in object-group network configuration mode. The syntax of this command is as follows:

```
host ip_address
```

The *ip_address* specifies the IP address of the host. Use this argument to specify a single IP address. Enter an IP address in dotted-decimal notation (for example, 192.168.12.15).

For example, to create a network object group that includes three host addresses, enter:

```
host1/Admin(config)# object-group network NET_OBJ_GROUP1
host1/Admin(config-objgrp-netw)# description Administrator Addresses
host1/Admin(config-objgrp-netw)# host 192.168.12.15
host1/Admin(config-objgrp-netw)# host 192.168.12.21
host1/Admin(config-objgrp-netw)# host 192.168.12.27
```
Configuring Service Object Groups

This section describes how to configure service object groups to streamline the creation of ACL entries that include protocol names and port names in an ACL. It includes the following sections:

- Creating a Service Object Group
- Adding a Description to a Service Object Group
- Defining Protocol Parameters for a Service Object Group

Creating a Service Object Group

To create a service object group, use the `object-group` command in configuration mode. The syntax of this command is as follows:

```
object-group service name
```

The keywords and arguments are as follows:

- `service`—Specifies a group of IP protocol and port specifications.
- `name`—Unique identifier of the object group. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to create a service object group, enter:

```
host1/Admin(config)# object-group service SERV_OBJ_GROUP1
host1/Admin(config-objgrp-serv)#
```

To remove the service object group from the configuration, enter:

```
host1/Admin(config)# no object-group service SERV_OBJ_GROUP1
```

Note

If you add new elements to an existing object group that is already in use by an entry in a large ACL, recommitting the ACL can take a long time, depending on the size of the ACL and the number of elements in the object group. In extreme cases, recommitting this ACL may cause the ACE to respond to commands slowly or even to become temporarily unresponsive. We recommend that you first remove the ACL entry that refers to the object group, make your modifications to the relevant object group, and then add the ACL entry back into the ACL.
Simplifying Access Control Lists with Object Groups

Adding a Description to a Service Object Group

To add an optional description to a service object group, use the `description` command in object group service configuration mode. The syntax of this command is as follows:

```
description text
```

The `text` argument is an unquoted text string with a maximum of 240 alphanumeric characters.

For example, to add a description to a service object group, enter:

```
host1/Admin(config)# object-group service SERV_OBJ_GROUP1
host1/Admin(config-objgrp-serv)# description intranet network object group
```

To remove a description from a service object group, enter:

```
host1/Admin(config)# object-group service SERV_OBJ_GROUP1
host1/Admin(config-objgrp-serv)# no description intranet network object group
```

Defining Protocol Parameters for a Service Object Group

To define protocol parameters for a service object group, use the `protocol` argument in object group service configuration mode. For TCP or UDP, the syntax of this command is as follows:

```
protocol [source {{ operator } port1 | port1 port2}] [{ { operator } port3 | port3 port4}]
```

For ICMP, the syntax of this command is:

```
icmp [icmp-type] [code {{ operator } icmp-code1 | range icmp-code1 icmp-code2}]
```

The keywords, arguments, and options are as follows:

- `protocol`—Name or number of an IP protocol. Enter a protocol name or an integer from 1 to 255 that represents an IP protocol number. See Table 1-2.
- `source`—(Optional) Specifies a source port for TCP, TCP-UDP, or UDP.
Note

To specify a destination port for TCP or UDP, use the **operator** argument with no preceding keyword. The destination keyword is implied.

- **operator**—Operand used to compare source and destination port numbers for TCP and UDP protocols or ICMP code numbers for the ICMP protocol. The operators are as follows:
  - **lt**—Less than.
  - **gt**—Greater than.
  - **eq**—Equal to.
  - **neq**—Not equal to.
  - **range**—An inclusive range of port values or ICMP message codes. If you enter this operator, enter a second port number value or a second ICMP message code to define the upper limit of the range.

- **port1 port2**—IP protocol source port name or port number from which you permit or deny access to services. Enter a port name or an integer from 0 to 65535. To enter an inclusive range of ports, enter two port numbers following the **range** keyword. The **port2** value must be greater than or equal to the **port1** value. See Table 1-3 for a list of well-known TCP keywords and port numbers and Table 1-4 for a list of well-known UDP keywords and port numbers.

- **port3 port4**—IP protocol destination port name or port number to which you permit or deny access to services. To enter an optional inclusive range of ports, enter the **range** keyword followed by two port numbers. The **port4** value must be greater than or equal to the **port3** value. See Table 1-3 for a list of well-known TCP keywords and port numbers and Table 1-4 for a list of well-known UDP keywords and port numbers.

- **icmp-type**—(Optional) If you entered ICMP as the protocol, specifies the type of ICMP messaging. Enter either an integer corresponding to the ICMP code number or one of the ICMP types listed in Table 1-5.

- **code**—(Optional) Specifies that a numeric operator and ICMP code follows.
Simplifying Access Control Lists with Object Groups

- `icmp-code1 icmp-code2`— Specifies an ICMP code number that corresponds to an ICMP type. See Table 1-5. To enter an optional inclusive range of ICMP codes, enter the `range` keyword followed by two ICMP code numbers. The `icmp-code1` value must be greater than or equal to the `icmp-code2` value. See Table 1-5 for list of ICMP codes and corresponding ICMP types.

For example, to add only a destination (destination keyword is implied) TCP port to a service object group, enter:

```
host1/Admin(config-objgrp-serv)# tcp eq 41
```

Enter additional object-group protocols as required.

To remove the destination TCP port from a service object group, enter:

```
host1/Admin(config-objgrp-prot)# no tcp
```

For example, to create a service object group for TCP (source port only), UDP (source and destination ports), and ICMP, enter:

```
host1/Admin(config)# object-group service TCP_UDP_ICMP
host1/Admin(config-objgrp-serv)# tcp source eq domain
host1/Admin(config-objgrp-serv)# udp source eq radius eq radius-acct
host1/Admin(config-objgrp-serv)# icmp echo code eq 0
```

To remove the ICMP protocol from the above service object group, enter:

```
host1/Admin(config-objgrp-prot)# no icmp echo code eq 0
```

### Using Object Groups in an ACL

To use object groups in an ACL, replace the normal network (`source_address mask`, and so on), service (`protocol operator port`), or ICMP type (`icmp_type`) arguments with the `object-group name` keyword and argument.

For example, to use object groups for all available parameters in the `access-list extended` command, enter the following command:

```
host1/Admin(config)# access-list acl_name extended {deny | permit} object-group service_grp_name object-group network_grp_name
```

You do not have to use object groups for all parameters. For example, you can use an object group for the source address, but identify the destination address with an IP address and subnet mask.
The following subsections provide examples of configuring extended ACLs with and without object groups, and how object group entries expand into multiple ACL entries:

- **Example of Configuring an Extended ACL Without Object Groups**
- **Example of Configuring the Equivalent Extended ACL Using Object Groups**
- **Example of How an Object Group Expands into Multiple ACL Entries**

### Example of Configuring an Extended ACL Without Object Groups

The following example shows how to configure an extended ACL that does not use object groups to restrict several hosts on the inside network from accessing several web servers. All other traffic is allowed.

```
host1/Admin(config)# access-list ACL_IN remark “object-group acl to deny specific hosts”
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.29 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.29 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.29 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.16 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.16 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.16 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.78 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.78 eq www
host1/Admin(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.78 eq www
host1/Admin(config)# access-list ACL_IN extended permit ip any any
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input ACL_IN
```
Example of Configuring the Equivalent Extended ACL Using Object Groups

The following example shows how to configure the equivalent of the extended ACL in the “Example of Configuring an Extended ACL Without Object Groups” section using two network object groups, one for the inside hosts, and one for the web servers. Notice how object groups simplify the configuration and allow you to easily modify it to add more hosts as follows:

```
host1/Admin(config)# object-group network DENIED
host1/Admin(config-objgrp-network)# host 10.1.1.4
host1/Admin(config-objgrp-network)# host 10.1.1.78
host1/Admin(config-objgrp-network)# host 10.1.1.89

host1/Admin(config)# object-group network WEB
host1/Admin(config-objgrp-network)# host 209.165.201.29
host1/Admin(config-objgrp-network)# host 209.165.201.16
host1/Admin(config-objgrp-network)# host 209.165.201.78

host1/Admin(config)# access-list ACL_IN remark “object-group acl to deny specific hosts”
host1/Admin(config)# access-list ACL_IN extended deny tcp object-group DENIED object-group WEB eq www
host1/Admin(config)# access-list ACL_IN extended permit ip any any

host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input ACL_IN
```

Example of How an Object Group Expands into Multiple ACL Entries

The following examples of show command output demonstrate how the ACE expands the single ACL entry that has an object group (see the “Example of Configuring the Equivalent Extended ACL Using Object Groups” section) into multiple ACL entries. The output of the show running-config access-list command displays the unexpanded object-group configuration of the ACL_IN ACL. The output of the show access-list ACL_IN command displays the expanded ACL entries.

```
host1/Admin# show running-config access-list
Generating configuration....

access-list ACL_IN remark “object-group acl to deny specific hosts”
access-list ACL_IN line 8 extended deny tcp object-group DENIED object-group WEB eq www
access-list ACL_IN line 16 extended permit ip any any
```
Applying an ACL to an Interface

Before you can start using a configured ACL, you must apply it to one or more interfaces.

To apply an ACL to the inbound or outbound direction of an interface and make the ACL active, use the **access-group** command in interface configuration mode. You can apply one ACL of each type (extended and EtherType) to both directions of the interface. See the “Inbound and Outbound ACLs” section for more information about ACL directions.

**Note**

If you have already applied a global ACL to all interfaces in a context, you cannot apply another ACL to an individual interface in that context. For details about applying an ACL globally, see the “Applying an ACL Globally to All Interfaces in a Context” section.
For connectionless protocols, you must apply the ACL to the source and destination interfaces if you want traffic to pass in both directions. For example, you can allow BGP in an ACL in transparent mode, and you must apply the ACL to both interfaces.

The syntax of this command is as follows:

```
access-group {input | output} acl_name
```

The keywords and arguments are as follows:

- **input | output**—Specifies the direction (inbound or outbound) of the interface to which you want to apply the ACL.
- **acl_name**—Identifier of an existing ACL that you want to apply to an interface. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input INBOUND
```

To remove an ACL from an interface, enter:

```
host1/Admin(config-if)# no access-group input INBOUND
```

### Applying an ACL Globally to All Interfaces in a Context

You can apply an ACL to all interfaces in a context at once, subject to the following conditions:

- No interface in the context has an ACL applied to it.
- You can globally apply one Layer 2 and one Layer 3 ACL in the inbound direction only.
- On Layer 2 bridged-group virtual interfaces (BVIs), you can apply both Layer 3 and Layer 2 ACLs.
- On Layer 3 virtual LAN (VLAN) interfaces, you can apply only Layer 3 ACLs.
In a redundancy configuration, the ACE does not apply a global ACL to the FT VLAN. For details about redundancy, see the *Cisco Application Control Engine Module Administration Guide*.

To apply an ACL globally to all interfaces in a context in the inbound direction, use the `access-group input` command in configuration mode. The syntax of this command is as follows:

```
access-group input acl_name
```

For the `acl_name` argument, enter the identifier of an existing ACL as an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

You can use this command to allow all traffic on all interfaces in a context by applying an ACL similar to the following example:

```
host1/Admin(config)# access-list ALL_ACCESS permit ip any any
```

Then, apply the ACL globally by entering:

```
host1/Admin(config)# access-group input ALL_ACCESS
```

To remove the ACL from all interfaces in the context, enter:

```
host1/Admin(config)# no access-group input ALL_ACCESS
```

### Filtering Traffic with an ACL

You can use an ACL to filter interesting traffic and instruct the ACE to either permit or deny the traffic based on the action in the ACL. To filter traffic using an ACL, use the `match access-list` command in a Layer 3 and Layer 4 class map.

When a packet matches an entry in an ACL, and if it is a `permit` entry, the ACE allows the matching result. If it is a `deny` entry, the ACE blocks the matching result. For details about configuring a Layer 3 and Layer 4 class map and policy map, see *Chapter 4, Configuring Security Access Control Lists*. 
ACL Configuration Examples

This section provides the following examples of the different types of ACLs available in the ACE:

- Examples of Extended ACLs
- Examples of EtherType ACLs

Examples of Extended ACLs

This section provides examples of extended ACLs. Use extended ACLs when you want to specify both the source IP address and the destination IP address (IP), ports (TCP or UDP), and ICMP types. For details about configuring extended ACLs, see the “Configuring an Extended ACL” section.

The following ACL allows all hosts (on the interface to which you apply the ACL) to go through the ACE:

```
host1/Admin(config)# access-list ACL_IN extended permit ip any any
```

The following ACL prevents hosts on 192.168.1.0/24 from accessing the 209.165.201.0/27 network. All other addresses are permitted.

```
host1/Admin(config)# access-list ACL_IN extended deny tcp 192.168.1.0 255.255.255.0 209.165.201.0 255.255.255.224
host1/Admin(config)# access-list ACL_IN extended permit ip any any
```

If you want to restrict access to only some hosts, then enter a limited permit entry. By default, all other traffic is denied unless explicitly permitted.

```
host1/Admin(config)# access-list ACL_IN extended permit ip 192.168.1.0 255.255.255.0 209.165.201.0 255.255.255.224
```

For a list of permitted keywords and well-known port assignments, see Table 1-3. DNS, Discard, Echo, Ident, NTP, RPC, SUNRPC, and Talk each require one definition for TCP and one for UDP. TACACS+ requires one definition for port 49 on TCP.

The following ACL example restricts all hosts (on the interface to which you apply the ACL) from accessing a website at address 209.165.201.29. All other traffic is allowed.

```
host1/Admin(config)# access-list ACL_IN extended deny tcp any host 209.165.201.29 eq www
host1/Admin(config)# access-list ACL_IN extended permit ip any any
```
The following ACLs allow all inside hosts to communicate with the outside network but only specific outside hosts to access the inside network:

```
host1/Admin(config)# access-list OUT extended permit ip any any
host1/Admin(config)# access-list IN extended permit ip host 209.168.200.3 any
host1/Admin(config)# access-list IN extended permit ip host 209.168.200.4 any
```

The following examples show how to configure ICMP ACLs. For details about configuring ICMP ACLs, see the “Configuring an Extended ACL” section.

```
host1/Admin(config)# access-list INBOUND extended permit icmp any any echo
host1/Admin(config)# access-list INBOUND extended permit icmp host 10.0.0.1 host 20.0.0.1 unreachable code range 0 3
```

This section contains the following topics:

- Inbound and Outbound ACLs
- IP Addresses for ACLs with NAT

### Inbound and Outbound ACLs

Traffic that flows across an interface in the ACE can be controlled in two ways:

- You can control traffic that enters the ACE by attaching an inbound ACL to the source interface.
- You can control traffic that exits the ACE by attaching an outbound ACL to the destination interface.

To allow any traffic to enter the ACE, you must attach an inbound permit ACL to an interface; otherwise, the ACE automatically refuses all traffic that enters that interface. By default, traffic can exit the ACE on any interface unless you restrict it by using an outbound ACL, which adds restrictions to those ACLs already configured in the inbound ACL.

---

**Note**

Inbound and outbound refer to the application of an ACL on an interface, either to traffic entering the ACE on an interface or traffic exiting the ACE on an interface.
You may choose to use an outbound ACL to simplify your ACL configuration. For example, if you want to allow three inside networks on three different interfaces to access each other, you can create a simple inbound ACL on each interface that allows all traffic on each inside interface (see Figure 1-1).

**Figure 1-1 Inbound ACLs**

The following commands create three inbound ACLs that allow all traffic on each inside interface:

```bash
host1/Admin(config)# access-list INSIDE extended permit ip any any
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input INSIDE

host1/Admin(config)# access-list HR extended permit ip any any
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input HR

host1/Admin(config)# access-list ENG extended permit ip any any
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input ENG
```

If you choose to allow only certain hosts on the inside networks to access a web server on the outside network, you can create a more restrictive ACL that allows only the specified hosts and apply it to the outbound direction of the outside
interface (see Figure 1-2). For information about NAT and IP addresses, see the “IP Addresses for ACLs with NAT” section. The outbound ACL prevents any other hosts from reaching the outside network.

The following commands create an ACL that allows only specified hosts and apply it to the outbound direction of the outside interface:

```
host1/Adm(config)# access-list OUTSIDE extended permit tcp host 209.165.201.4 host 209.165.200.225 eq www

host1/Adm(config)# access-list OUTSIDE extended permit tcp host 209.165.201.6 host 209.165.200.225 eq www

host1/Adm(config)# access-list OUTSIDE extended permit tcp host 209.165.201.8 host 209.165.200.225 eq www

host1/Adm(config)# interface vlan 100
host1/Adm(config-if)# access-group output OUTSIDE
```
When you use NAT, the IP addresses that you specify for an ACL depend on the interface to which the ACL is attached. You must use addresses that are valid on a network that is connected to the interface. This guideline applies for both inbound and outbound ACLs. The ACL direction does not determine the address used, only the interface to which the ACL is attached determines the address that is used.

For example, suppose that you want to apply an ACL to the inbound direction of the interface. You configure the ACE to perform NAT on the inside source addresses when they access outside addresses. Because the ACL is applied to the
inside interface, the source addresses are the original untranslated addresses. Because the outside addresses are not translated, the destination address used in the ACL is the real address (see Figure 1-3).

**Figure 1-3  **  *IP Addresses in ACLs: NAT Used for Source Addresses*

The following commands create an ACL that allows inside source network 10.1.1.0/24 to access the outside destination host 209.165.200.225 and apply the ACL to VLAN interface 100:

```bash
host1/Admin(config)# access-list INSIDE extended permit ip 10.1.1.0
255.255.255.0 host 209.165.200.225
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input INSIDE
```

...
If you want to allow an outside host to access an inside host, you can apply an inbound ACL to the outside interface. You must specify the translated address of the inside host in the ACL because that address is the address that can be used on the outside network (see Figure 1-4).

**Figure 1-4  IP Addresses in ACLs: NAT used for Destination Addresses**

The following commands create an ACL that allows outside host 209.165.200.225 to access inside host 209.165.201.5 (the translated address of the host 10.1.1.34). The last command applies the ACL to VLAN interface 100.

```
host1/Admin(config)# access-list OUTSIDE extended permit ip host 209.165.200.225 host 209.165.201.5
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input OUTSIDE
```
If you perform NAT on both interfaces, you must verify the addresses that are visible on each interface when you create and apply ACLs. In Figure 1-5, an outside server uses static NAT so that a translated address appears on the inside network.

**Figure 1-5**  **IP Addresses in ACLs: NAT used for Source and Destination Addresses**

The following commands create an ACL that allows inside source network 10.1.1.0/24 to access the outside destination host 10.1.1.56 (the translated address of the host 209.165.200.225). The last command applies the ACL to VLAN interface 100.

```
host1/Admin(config)# access-list INSIDE extended permit ip 10.1.1.0 255.255.255.0 host 10.1.1.56
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input INSIDE
```

For an example of IP addresses used in outbound ACLs, see Figure 1-2.
Examples of EtherType ACLs

This section provides examples of EtherType ACLs. For details about configuring an EtherType ACL, see the “Configuring an EtherType ACL” section.

The following example shows an ACL that allows common EtherTypes to originate on the inside interface:

```plaintext
host1/Admin(config)# access-list ETHER ethertype permit ipv6
host1/Admin(config)# access-list ETHER ethertype permit bpdu
host1/Admin(config)# access-list ETHER ethertype permit mpls
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group output ethertype ETHER
```

The following example shows an ACL that allows some EtherTypes through the ACE but denies IPv6:

```plaintext
host1/Admin(config)# access-list ETHER ethertype deny ipv6
host1/Admin(config)# access-list ETHER ethertype permit bpdu
host1/Admin(config)# access-list ETHER ethertype permit mpls
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input ethertype ETHER
```

The following example shows an ACL that denies traffic with an EtherType BPDU but allows all others on both interfaces:

```plaintext
host1/Admin(config)# access-list nonIP ethertype deny bpdu
host1/Admin(config)# access-list nonIP ethertype permit any
host1/Admin(config)# interface vlan 100
host1/Admin(config-if)# access-group input ethertype nonIP
```

Displaying ACL Configuration Information and Statistics

This section describes the `show` commands that you can use to display ACL configurations and statistics. It contains the following topics:

- Displaying ACL Configuration Information
- Displaying ACL Statistics
Displaying ACL Configuration Information

You can display all ACL configuration information, including the interfaces on which you applied the ACLs by using the `show running-config` command. The syntax of this command is as follows:

```
show running-config
```

To display only the ACLs and their entries, use the `show running-config access-list` command in Exec mode. The syntax of this command is as follows:

```
show running-config access-list
```

Displaying ACL Statistics

You can display ACL statistics for a particular ACL by using the `show access-list` command. The syntax of this command is as follows:

```
show access-list name [detail]
```

The argument and optional keyword are as follows:

- `name`—Identifier of an existing ACL. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.
- `detail`—(Optional) Displays detailed ACL information, including a 4-byte MD5-hash value that the ACE uses to identify the ACL entry that caused a deny syslog (106023). See the description of the 0xnnnnnnnn output field in Table 1-6.

Table 1-6 describes the fields in the `show access-list detail` command output.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access-list</td>
<td>Name of the security ACL.</td>
</tr>
<tr>
<td>Elements</td>
<td>Number of entries in the ACL.</td>
</tr>
</tbody>
</table>
### Table 1-6  Field Descriptions for the show access-list detail Command Output (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Current status of the ACL: ACTIVE when the ACL is associated with at least one interface or NOT-ACTIVE when the ACL is not associated with at least one interface.</td>
</tr>
<tr>
<td>Remark</td>
<td>Configured comments that describe the ACL.</td>
</tr>
<tr>
<td>Entries</td>
<td>Full text of all entries in the ACL.</td>
</tr>
<tr>
<td>Hitcounts</td>
<td>Hit counts for each ACL entry.</td>
</tr>
<tr>
<td>hash 1</td>
<td>32-bit hexadecimal MD5-hash value that the ACE computes from the access-list command immediately when you configure an ACL. The ACE includes this hash value in deny syslog messages to help you identify the ACL entry that caused the syslog in the output of this command. This hash value is line-number independent. To prevent possible discrepancies between the hash values in the deny syslog message and the output of this command after a reboot, be sure to use Tab completion or type entire keywords in the CLI when configuring individual entries in an ACL.</td>
</tr>
<tr>
<td>hash 2</td>
<td>16-bit hexadecimal (0xnnnn) MD5-hash value that the ACE computes from the expanded access-list entries resulting from the object groups that you configure in an ACL. The ACE computes the hash 2 value when you activate the ACL on an interface. For ACLs that do not have object groups, the hash 2 value is always 0x0. The ACE also includes the hash 2 value in deny syslog messages to help you identify the expanded ACL entry that caused the syslog. This hash value is also line-number independent. To uniquely identify the expanded ACL entry that caused the syslog, you need to search for an entry in this command output that matches both the hash 1 and the hash 2 hexadecimal values.</td>
</tr>
</tbody>
</table>
Displaying the ACL Merge Tree Node Usage

You can display the ACL merge tree node usage by entering the following command:

```
show np 1 | 2 access-list resource
```

The output of this command shows the used, guaranteed, and maximum values of the various ACL merge tree nodes. For more details about this command and troubleshooting acl-memory Denied counts in the output of the `show resource usage` command, see Step 5 in the Troubleshooting Access Control Lists section of the ACE Troubleshooting Wiki.

Clearing ACL Statistics

You can clear ACL statistics (hit counts for ACL entries) by using the `clear access-list` command in Exec mode. The syntax of this command is as follows:

```
clear access-list name
```

The `name` argument is an existing ACL. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/Admin# clear access-list acl1
```

**Note**

If you configured redundancy, then you must explicitly clear ACL statistics (hit counts) on both the active and the standby ACEs. Clearing statistics on the active module only will leave the standby module’s statistics at the old value.
CHAPTER 2

Configuring Authentication and Accounting Services

This chapter describes how to configure the Cisco Application Control Engine (ACE) module to perform user authentication and accounting (AAA) services to provide a higher level of security for users accessing the ACE. The AAA services allow you to use multiple AAA servers to control who can access the ACE and to track the actions of each user who accesses the ACE. Based on the username and password combination provided, the ACE performs local user authentication using the local database or remote user authentication and accounting using external AAA servers.

This chapter contains the following major sections:

- AAA Overview
- Authentication and Accounting Configuration Quick Start
- Configuring the AAA Server
- Creating User Accounts
- Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server
- Defining the Login Authentication Method
- Defining the Default Accounting Method
- Viewing AAA Status and Statistics
AAA Overview

AAA provides management security for user access to the ACE through a combination of authentication and accounting services. AAA informs the ACE who the user is and what the user did. You can use authentication alone or with accounting. ACE provides security for the management access methods to the ACE, including the command-line interface (CLI) or Simple Network Management Protocol (SNMP).

You can access the ACE CLI through the console port or by a Telnet or SSH session. When you log in to the ACE using either a Telnet or SSH connection, and if the ACE is configured for AAA server-based authentication, a temporary SNMP user entry is automatically created. The SNMPv3 protocol data units (PDUs) with the associated Telnet or SSH login name as the SNMPv3 user are authenticated by the ACE.

As part of the authentication process, the ACE associates each user with a user role and a domain privilege pair under a specific virtual context. Each virtual context behaves like an independent device with its own configuration, security policies, interfaces, and domains. A user context can be independently managed with other user contexts. A domain provides a namespace in which a user operates, and each user is associated with at least one domain. The role assigned to a user determines the operations that a user can perform on the objects in a domain and the command set available to that user. Each context has a virtual AAA instance running to provide authentication for the users logging in and accounting services to log user activity.

Each virtual context on the ACE can have its own IP address. You can access each virtual context in an ACE through the console port or a Telnet or SSH session by specifying this IP address. Users can also send SNMP requests to the ACE by using this IP address.

Note

Only the Admin context is accessible through the console port; all other contexts can be reached through Telnet or SSH.

The administrator of each virtual context is able to perform, independent from other contexts, the following actions:

- Configure different AAA servers and their parameters
• Create the same username across contexts, and associate the username with a unique role in a context and multiple domains

• Share AAA servers. Each user, however, must be authenticated for each virtual context and must use the same password

• Log user accounting activities, which are distinguished by the context in which a user has signed in

• Display the users currently authenticated on the virtual context

Each user who accesses the ACE from a specific IP address needs to authenticate once only. The user authentication sequence remains in effect until the authentication session expires on the ACE.

The ACE runs the AAA client, which sits between the users and the AAA server. On one side, the ACE prompts each user for their credentials (username and password). On the other side, the ACE queries the identified AAA servers to determine if the user being authenticated has supplied the correct credentials and is authorized access to the ACE.

The ACE performs authentication using either the local user database that resides on the ACE or a remote AAA server. The ACE can use a Remote Access Dial-In User Service (RADIUS), Terminal Access Controller Access Control System Plus (TACACS+), or Lightweight Directory Access Protocol (v3) (LDAP) server for remote authentication and designation of access rights.

This section contains the following topics:

• **Local Database and Remote Server Support**

• **Authentication Overview**

• **Accounting Overview**

---

**Local Database and Remote Server Support**

The ACE supports local authentication using a local database on the ACE or remote authentication using one or more AAA servers. AAA remote servers are grouped into independent groups of TACACS+, RADIUS, or LDAP servers. For a group of servers, the ACE bases the selection of the server to use on the first active server in the group.

---

**Note**

“First” refers to the order in which servers have been configured.
When a user logs in to an ACE, the servers are accessed one at a time, starting with the first server specified in the configuration, until a server responds to the ACE.

When you configure server groups using the server group authentication method, the ACE sends an authentication request to the first AAA server in the group as follows:

- If the remote AAA server fails to respond, the ACE attempts to contact the next server in the group until a remote AAA server responds to the authentication request.
- If all AAA servers in the server group fail to respond, the ACE tries to contact the AAA servers in the next configured server group.
- If all remote AAA servers fail to respond, by default, the ACE attempts to authenticate the user against the local database.

If the username and password are successfully authenticated either locally or remotely, the ACE allows the user to log in, and the user is assigned a unique role (as specified through the `role` command, which determines the commands and resources available to each user).

Each server within a group can assume an active or an inactive state if a network connection failure occurs. The policy used to select the AAA server takes the server state into account. The ACE monitors the AAA server operation by sending authentication requests to a timed-out server. If the ACE does not receive confirmation from the server within a user-specified number of retries, the ACE declares the server to be unresponsive and initiates the sequence to contact the next available server specified in the server group.

If a dead-time interval is specified for a AAA server and the connection to server A fails, the ACE marks server A as out of service and skips server A for the duration of the dead-time interval. The ACE then sends probe access-request packets to verify that the AAA server is available and can receive authentication requests. When the server responds to a probe access-request packet, the connection resumes to server A.

This section contains the following topics:

- Local Database
- TACACS+ Server
- RADIUS Server
- LDAP Directory Server
Local Database

You can configure user account access to the local database on the ACE for CLI access authentication. When a user attempts to access the ACE CLI by using the console port or a Telnet or SSH session, the ACE consults the local user database for the username and password. By default, each user assumes the Network-Monitor role and is allowed to operate on all domains.

If you specify local authentication as the fallback method and the specified AAA servers in a server group are unavailable for authentication, the ACE then attempts to access the local database to perform user authentication and accounting.

TACACS+ Server

TACACS+ controls user access to the ACE by exchanging Network Access Server (NAS) information between the ACE and a centralized database to determine the identity of a user. TACACS+ is an enhanced version of TACACS, a User Datagram Protocol (UDP)-based access-control protocol that is specified by RFC 1492. TACACS+ uses TCP to ensure reliable delivery and encrypt all traffic between the TACACS+ server and the TACACS+ daemon on the ACE.

A TACACS+ server can provide user authentication and accounting functions. These services, while all part of TACACS+, are independent of one another, so a given TACACS+ configuration can use any or all of the services.

The TACACS+ protocol encrypts the user password information using the MD5 encryption algorithm and adds a TACACS+ packet header. This header information identifies the packet type being sent (for example, an authentication packet), the packet sequence number, the encryption type being used, and the total packet length. The TACACS+ protocol forwards the packet to the TACACS+ server.

To maintain security between the ACE and the TACACS+ server, you can specify an encryption key (shared secret) for all communication between the ACE and the TACACS+ server. For correct operation, you must specify the identical encryption key on both the ACE and the TACACS+ server.

RADIUS Server

RADIUS is a client-server access protocol that is used by the NAS to authenticate users attempting to connect to the ACE. The NAS functions as a client, passing user information to one or more RADIUS servers. The NAS permits or denies
network access to a user based on the response that it receives from a RADIUS server. RADIUS uses UDP for connectionless transport between the RADIUS client and server. For more information about how the RADIUS protocol operates, see RFC 2138.

To maintain security between the ACE and the RADIUS server, you can specify an encryption key (shared secret) for all communication between the ACE and the RADIUS server. For correct operation, you must specify the identical encryption key on both the ACE and the RADIUS server.

**LDAP Directory Server**

LDAP is an open-standard client-server authentication protocol for accessing X.500 Directory Access Protocol (DAP) directory services. LDAP runs over TCP/IP or other connection-oriented transfer services. The ACE supports only LDAP version 3 for simple authentication and search operations. For more information about how the LDAP protocol operates, see RFC 2251.

The LDAP information model is based on entries. An entry is a collection of attributes that has a globally unique distinguished name (DN). The DN is used in the LDAP database to refer to an entry. Each entry contains one or more attributes that describe the entry, and each attribute has a type and one or more values. The types are mnemonic strings, such as “cn” for a common name, or “mail” for an e-mail address.

The LDAP client (the ACE) requests user authentication with the LDAP server and retrieves the user profile by requesting a search through the directory database maintained by the server. The LDAP server maintains a directory of entries, which are arranged into a hierarchical structure called the Directory Information Tree (DIT).

The LDAP client performs operations on the directory data. LDAP allows you to search the directory for data that meets the arbitrary user-specified criteria. You can specify which part of the directory to search and what information to return. A search filter that uses Boolean conditions specifies the directory data that matches the search.

---

**Note**

The ACE does not support update, compare, and cancel operations with the LDAP server. In addition, the ACE does not support an unsolicited notification from the LDAP server. Supported messages include bindRequest, bindResponse, unbindRequest, searchRequest, searchResEntry, and searchResDone.
Authentication Overview

Authentication allows you to control user access to the ACE CLI by requiring specification of a valid username and password. You can access the ACE CLI through the console port or by a Telnet or SSH session. For each management access path to the ACE, you can configure one or more of the following security control options: local database, remote (RADIUS, TACACS+, or LDAP), or no password verification.

The host is prompted by the ACE to provide a valid username and password. After the designated RADIUS, TACACS+, or LDAP server authenticates the username and password, the ACE provides access rights to the user.

Accounting Overview

Accounting tracks and maintains a log of useful information during each user management session with the ACE. You can use this information to generate reports for troubleshooting and auditing purposes. Accounting logs can be stored locally on the ACE or sent to remote AAA servers. If the ACE is also configured to authenticate the user, the AAA server maintains accounting information by username.

- For a TACACS+ server, accounting information includes user commands entered on the ACE, the duration of each session, and when sessions start and stop.
- For a RADIUS server, accounting information includes the duration of each session and when sessions start and stop.

When ACE user commands are being logged on a TACACS+ server, the server prefaces each command with either a “<0:>” or a “<1:>” to indicate the success or failure of the command as issued by a user on the ACE CLI. For example, when a user attempts to enter a command on the ACE and that user does not have the correct role privileges, a “<1:>” appears to indicate a failure.
Authentication and Accounting Configuration Quick Start

Table 2-1 provides a quick overview of the steps required to create and configure authentication and accounting for the ACE. Each step includes the CLI command required to complete the task.

Table 2-1  Authentication and Accounting Configuration Quick Start

<table>
<thead>
<tr>
<th>Task and Command Example</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Configure the authentication and accounting service settings on the TACACS+, RADIUS, or LDAP server.</td>
<td></td>
</tr>
<tr>
<td>2. If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, log directly in to, or change to, the correct context.</td>
<td></td>
</tr>
</tbody>
</table>

```
host1/Admin# changeto C1
host1/C1#
```

The rest of the examples in this table use the Admin context, unless otherwise specified. For details on creating contexts and user accounts to provide access to the local database on the ACE for CLI access authentication, see the Cisco Application Control Engine Module Virtualization Configuration Guide.

<table>
<thead>
<tr>
<th>3. Enter configuration mode by entering config.</th>
</tr>
</thead>
</table>
|```
host1/Admin# config
Enter configuration commands, one per line. End with CNTL/Z
host1/Admin(config)#
```|
4. Configure the individual AAA server parameters. For example, to configure RADIUS server authentication parameters, enter:

```
host1/Admin(config)# radius-server host 192.168.2.3 key HostKey
host1/Admin(config)# radius-server host 192.168.2.3 key 7 secret_1256
host1/Admin(config)# radius-server host 192.168.2.3 auth-port 1645
host1/Admin(config)# radius-server host 192.168.2.3 acct-port 1646
host1/Admin(config)# radius-server host 192.168.2.3 authentication
host1/Admin(config)# radius-server host 192.168.2.3 accounting
host1/Admin(config)# radius-server host 192.168.2.3 timeout 25
host1/Admin(config)# radius-server host 192.168.2.3 retransmit 3
```

5. Configure independent server groups of TACACS+, RADIUS, or LDAP servers. For example, to create a RADIUS server group, enter:

```
host1/Admin(config)# aaa group server radius RAD_Server_Group1
host1/Admin(config-radius)# server 192.168.252.1
host1/Admin(config-radius)# server 192.168.252.2
host1/Admin(config-radius)# server 192.168.252.3
host1/Admin(config-radius)# deadtime 15
```

6. Configure the authentication method used for login to the ACE CLI.

```
host1/Admin(config)# aaa authentication login console group RAD_Server_Group1 local none
```

7. Configure the default accounting method.

```
host1/Admin(config)# aaa accounting default group RAD_Server_Group1 local
```

8. (Optional) Save your configuration changes to flash memory.

```
host1/Admin(config)# exit
host1/Admin# copy running-config startup-config
```
Configuring the AAA Server

This section describes how to set up a TACACS+ or RADIUS server such as the Cisco Secure Access Control Server (ACS). It also covers general guidelines for setting up an LDAP directory server, such as OpenLDAP Software available from OpenLDAP Project. This section is intended as a guide to help ensure proper communication with the AAA server and an ACE operating as the AAA client.

For details on configuring the Cisco Secure ACS, OpenLDAP Software, or another AAA server, see the documentation that is provided with the software.

Configuring a TACACS+ Server

This section contains the following topics:

- Configuring Authentication Settings on the TACACS+ Server
- Configuring Accounting Settings on the TACACS+ Server
- Defining Private Attributes for Virtualization Support in a TACACS+ Server

**Note**

For the ACE to properly perform user authentication using a TACACS+ server, the username and password must be identical on both the ACE and the TACACS+ server.

Configuring Authentication Settings on the TACACS+ Server

To configure the TACACS+ authentication settings on Cisco Secure ACS, perform the following steps:

**Step 1**
Go to the Network Configuration section of the Cisco Secure ACS HTML interface, and then go to the Add AAA Client page.

**Step 2**
Configure the following selections:

- AAA Client Hostname—Enter the name that you want assigned to the ACE.
- AAA Client IP Address—Enter the IP address of the Ethernet interface that will be used for communicating with the TACACS+ server.
• Key—Enter the shared secret that the ACE and Cisco Secure ACS use to authenticate transactions. You must specify the identical shared secret on both the Cisco Secure ACS and the ACE. The key is case sensitive.

• Authenticate Using—Choose TACACS+ (Cisco IOS).

Note The TACACS+ (Cisco IOS) drop-down item is the title for the Cisco TACACS+ authentication function. The TACACS+ (Cisco IOS) selection activates the TACACS+ option when using Cisco Systems access servers, routers, and firewalls that support the TACACS+ authentication protocol. This includes support with an ACE as well.

Step 3 Click Submit + Restart.

Configuring Accounting Settings on the TACACS+ Server

To configure the TACACS+ accounting service for the Cisco Secure ACS, perform the following steps:

Step 1 In the System Configuration section of the Cisco Secure ACS interface, the Logging Configuration page, click CSV TACACS+ Accounting. The CSV TACACS+ Accounting File Configuration page appears.

Step 2 Confirm that the Log to CSV TACACS+ Accounting report check box is checked.

Step 3 Under Select Columns To Log, in the Attributes column, click the attribute that you want to log. Click -> to move the attribute into the Logged Attributes column. Click Up or Down to move the column for this attribute to the desired position in the log. Repeat until all the desired attributes are in the desired positions in the Logged Attributes column.

Step 4 Click Submit when you finish moving the attributes into the Logged Attributes.
Defining Private Attributes for Virtualization Support in a TACACS+ Server

You can create the same username across contexts and associate it with a unique role in a context and multiple domains. Contexts can share a TACACS+ server, but the user must be authenticated for each context and must use the same password.

When a user attempts to log in to the ACE, the TACACS+ client on the ACE sends the username and password to the remote TACACS+ server for authentication. The TACACS+ server retrieves a user’s profile as part of the authentication request. Once the user is successfully authenticated, the TACACS+ server returns a user profile to the TACACS+ client on the ACE with the authentication status. If the associated context of the user attempting to log in matches the contexts of the user profile obtained through the TACACS+ server, the TACACS+ client updates the user profile with the remote server user profile. If the contexts do not match, the user profile is updated with a default role (Network-Monitor) and a default domain (default-domain).

Configure the user profile on the TACACS+ server to run an Exec shell to configure a shell command authorization for the user. Define a custom attribute with a value string in the following format:

```
shell:<contextname>=<role> <domain1> <domain2>...<domainN>
```

or

```
shell:<contextname>*<role> <domain1> <domain2>...<domainN>
```

**Note**

If you are using Cisco IOS command authorization, be sure to use an asterisk (*) rather than the equals sign (=) operator in the shell command string. The equals sign indicates that Cisco IOS software expects a required field to follow. Cisco IOS software does not recognize the role field, so using the equals sign in this case will cause Cisco IOS authorization to fail.

**Note**

The user profile attribute serves an important configuration function for a TACACS+ server group. If the user profile attribute is not obtained from the server during authentication, or if the profile is obtained from the server but the context name(s) in the profile do not match the context in which the user is trying to log in, a default role (Network-Monitor) and a default domain (default-domain) are assigned to the user if the authentication is successful.
To configure the TACACS+ role and domain settings on Cisco Secure ACS, perform the following steps:

**Step 1** Go to the Interface Configuration section of the Cisco Secure ACS HTML interface and access the TACACS+ (Cisco IOS) page. Perform the following actions:

a. Under the TACACS+ Services section of the page, the User column or the Group column depending on your configuration, check the **Shell (exec)** check box.

b. Under the Advanced Configuration Options section of the page, check the **Display a window for each service selected in which you can enter customized TACACS+ attributes** check box.

c. Click **Submit**.

**Step 2** Go to the Advanced Options page of the Interface Configuration section of the Cisco Secure ACS HTML interface. Perform the following actions:

a. Check the **Per-user TACACS+/RADIUS Attributes** check box.

b. Click **Submit**.

**Step 3** Go to the User Setup section of the Cisco Secure ACS HTML interface and double-click the name of an existing user that you want to define a user profile attribute for virtualization. The User Setup page appears.

**Step 4** Under the TACACS+ Settings section of the page, configure the following settings:

- Check the **Shell (exec)** check box.

- Check the **Custom attributes** check box.

- In the text box under the Custom attributes, enter the user role and associated domain for a specific context in the following format:

  \[\text{shell:<contextname>=<role> <domain1> <domain2>...<domainN>}\]

For example, to assign the selected user to the C1 context with the role **ROLE1** and the domain **DOMAIN1**, enter **shell:C1=ROLE1 DOMAIN1**.

You can also substitute an asterisk (*) for the equals sign (=) as follows:

\[\text{shell:<contextname>*<role> <domain1> <domain2>...<domainN>}\]

Use the above shell string if you are also using Cisco IOS command authorization.
Step 5 Under the Checking This option Will PERMIT all UNKNOWN Services section of the page, check the Default (Undefined) Services check box to permit unknown services.

Step 6 Click Submit when you finish configuring the TACACS+ role and domain settings.

For example, if USER1 is assigned the role ADMIN and the domain MYDOMAIN1 (where shell:Admin=ADMIN MYDOMAIN1), then one of the following can occur:

- If USER1 logs in through the Admin context, that user is automatically assigned the Admin role and the MyDomain1 domain.
- If USER1 logs in through a different context, that user is automatically assigned the default role (Network-Monitor) and the default domain (default-domain). In this case, the user profile attribute is not obtained from the TACACS+ server during authentication.

---

Configuring a RADIUS Server

This section contains the following topics:

- Configuring Authentication Settings on the RADIUS Server
- Configuring Accounting Settings on the RADIUS Server
- Defining Private Attributes for Virtualization Support in a RADIUS Server

Configuring Authentication Settings on the RADIUS Server

To configure the RADIUS authentication settings on Cisco Secure ACS, perform the following steps:

Step 1 Go to the Network Configuration section of the Cisco Secure ACS HTML interface.

Note If you are using Network Device Groups (NDGs), you must also click the name of the NDG to which you want to add the AAA client entry.
Step 2  Under the AAA Clients table, choose Add Entry. The Add AAA Client page appears.

Step 3  Configure the entries on the Add AAA Client page as follows:

- **AAA Client Hostname**—Enter a name that you want assigned to the ACE.
- **AAA Client IP Address**—Enter the IP address of the Ethernet Management port or of an ACE circuit (depending on how the ACE is configured to communicate with the Cisco Secure ACS).
- **Key**—Enter the shared secret that the ACE and Cisco Secure ACS use to authenticate transactions. You must specify the identical shared secret on both the Cisco Secure ACS and the ACE. The key is case sensitive.
- **Authenticate Using**—Choose the AAA protocol. In the case of RADIUS, choose the vendor used for communication with the AAA client.

Step 4  Click **Submit + Restart**.

Cisco Secure ACS saves the AAA client entry and restarts its services, after which it will accept and process RADIUS requests from the ACE.

---

**Configuring Accounting Settings on the RADIUS Server**

To configure the RADIUS accounting settings on Cisco Secure ACS, perform the following steps:

Step 1  Choose **System Configuration > Logging > CSV RADIUS Accounting**. The CSV RADIUS Accounting File Configuration page appears.

Step 2  Confirm that the **Log to CSV RADIUS Accounting report** check box is checked.

Step 3  In the Select Columns To Log table, check that the RADIUS attributes that you want to see in the RADIUS accounting log appear in the Logged Attributes list. In addition to the standard RADIUS attributes, you will see several special logging attributes provided by Cisco Secure ACS, such as Real Name, ExtDB Info, and Logged Remotely. For more information about these attributes, see the user guide for your Cisco Secure ACS.
Step 4  (Optional) If you are using Cisco Secure ACS for Windows Server, you can specify log file management, which determines how large the RADIUS account files can be, how many are retained, how long they are retained, and where they are stored.

Note  You can use Cisco Secure ACS to send accounting data to other AAA servers by configuring the AAA server entry in the Network Configuration section of the HTML interface. For details, see the applicable Cisco Secure ACS user guide.

Step 5  Click Submit when you finish moving the attributes into the Logged Attributes. Cisco Secure ACS saves and implements the changes that you made to its RADIUS accounting configuration.

Defining Private Attributes for Virtualization Support in a RADIUS Server

You can create the same username across contexts and associate it with a unique role in a context and multiple domains. Contexts can share a RADIUS server, but the user must be authenticated for each context and must use the same password.

When a user attempts to log in to the ACE, the RADIUS client on the ACE sends the username and password to the remote RADIUS server for authentication. The RADIUS server retrieves a user’s profile as part of the authentication request. Once the user is successfully authenticated, the RADIUS server returns a user profile to the RADIUS client on the ACE with the authentication status. If the associated context of the user attempting to log in matches the contexts of the user profile obtained through the RADIUS server, the RADIUS client updates the user profile with the remote server user profile. If the contexts do not match, the user profile is updated with the default role (Network-Monitor) and the default domain (default-domain).

Configure the user profile on the RADIUS server as a Vendor Specific Attribute with vendor Id Cisco (09) and subattribute type CiscoAVPair (type 01) with a value string in the following format:

shell:<contextname>=<role> <domain1> <domain2>...<domainN>
The user profile attribute serves an important configuration function for a RADIUS server group. If the user profile attribute is not obtained from the server during authentication, or if the profile is obtained from the server but the context name(s) in the profile do not match the context in which the user is trying to log in, a default role (Network-Monitor) and a default domain (default-domain) are assigned to the user if the authentication is successful.

To configure the RADIUS role and domain settings on Cisco Secure ACS, perform the following steps:

**Step 1**
Go to the User Setup section of the Cisco Secure ACS HTML interface and double-click the name of an existing user that you want to define a user profile attribute for virtualization. The User Setup page appears.

**Step 2**
Under the Cisco IOS/PIX RADIUS Attributes section of the page, configure the following settings:

- Check the [009\001] cisco-av-pair check box.
- In the text box below the [009\001] cisco-av-pair check box, enter the user role and associated domain for a specific context in the following format:
  
  shell:<contextname>=<role> <domain1> <domain2>...<domainN>

  For example, to assign the selected user to the C1 context with the role ROLE1 and the domain DOMAIN1, enter shell:C1=ROLE1 DOMAIN1.

**Step 3**
Click Submit when you finish configuring the RADIUS role and domain settings.

For example, if USER1 is assigned the role ADMIN and the domain MYDOMAIN1 (where shell:Admin=ADMIN MYDOMAIN1), then one of the following can occur:

- If USER1 logs in through the Admin context, that user is automatically assigned the Admin role and the MyDomain1 domain.
- If USER1 logs in through a different context, that user is automatically assigned the default role (Network-Monitor) and the default domain (default-domain). In this case, the user profile attribute is not obtained from the RADIUS server during authentication.
Configuring an LDAP Server

This section describes how to set up an LDAP directory server, such as the OpenLDAP and Microsoft Active Directory Servers. This section is intended as a general guide to help ensure proper communication with an LDAP server and an ACE operating as an LDAP client.

To configure the OpenLDAP directory server, perform the following steps:

---

**Step 1** Edit the provided slapd.conf example (usually installed as /usr/local/etc/openldap/slapd.conf) to contain a BDB database definition, schema definition, rootDN, and root password.

**Step 2** Add a private schema to include the definition of the private attributes (context ID and user profile) and private objectClass, or modify the existing object class. Include this schema in the slapd.conf.

**Step 3** Start the LDAP server, slapd.

*Note* slapd is a standalone LDAP directory server that runs on many different platforms.

**Step 4** Create the LDAP database; that is, create a file in LDIF format that contains the database. Ensure that the LDIF file (example.ldif) contains the following:

- `dn: dc=example,dc=com`
- `objectClass: dcObject`
- `objectClass: organization`
- `dc: example`
- `o: Example Corporation`
- `description: The Example Corporation`
- `dn: cn=Manager,dc=example,dc=com`
- `objectclass: organizationalRole`
- `cn: Manager`

**Step 5** Run `ldapadd` to insert these entries into your directory. For example:

```
ldapadd -x -D "cn=Manager,dc=example,dc=com" -w secret -f example.ldif
```
Defining Private Attributes for Virtualization Support in an LDAP Server

The LDAP client on the ACE does not assume any specifics about the database structure maintained by the LDAP server. Instead, it assumes that the \{userid, contextid\} pair uniquely identifies an entry in the database and that this entry contains the user profile attribute. The LDAP client performs a search based on these two attributes using the search filter configured on the ACE. The LDAP server locates the correct user entry and the user profile attribute, which is part of that entry, and returns this information in the search response.

The LDAP client can operate in applications where virtualization is not a requirement. In this case, the username alone uniquely identifies the user entry. You configure the search filter to include only the $userid variable (no $contextid). You define these two private attributes from the ACE CLI by entering the `attribute user-profile` command (see the “Configuring the User Profile Attribute Type for an LDAP Server Group” section).

You define the user profile attribute value in the following format:

```
shell:<contextname>=<role> <domain1> <domain2>...<domainN>
```

**Note**
The user profile attribute serves an important configuration function for an LDAP server group. If the user profile attribute is not obtained from the server during authentication, or if the profile is obtained from the server but the context name(s) in the profile do not match the context in which the user is trying to log in, a default role (Network-Monitor) and a default domain (default-domain) are assigned to the user if the authentication is successful.

When virtualization is a requirement, the LDAP server must have the contextid attributes defined in the schema. The user-profile attribute (the role-domain information) is required if you need to assign different roles and domains to different users. See the LDAP client documentation for information about how to extend the attributetype directive used by the slapd LDAP directory server.
To define private attributes for virtualization support in an LDAP server, perform the following steps:

**Step 1** Add a private schema to include the definition of the private attributes (context ID and user profile) and the private objectClass. An example is as follows:

```
attributetype (2.5.4.55 NAME ( 'ctxid' 'contextid' )
   DESC 'virtual context name'
   SUP name )

attributetype ( 2.5.4.56 NAME ( 'usrprof' 'userprofile' )
   DESC 'user profile'
   SUP name )

objectclass ( 2.5.6.30 NAME 'ctxperson'
   DESC 'a person'
   SUP top STRUCTURAL
   MUST cn
   MAY ( $ ctxid $ usrprof ) )
```

The example includes arbitrary OIDs. The OIDs that you define must not overlap with any existing OIDs in the LDAP server database.

**Step 2** Include this private schema in the configuration, which would be sladp.conf in the case of OpenLDAP.

**Step 3** Define the LDAP database in LDAP Data Interchange Format (LDIF) with entries that contain the context ID and the user profile. LDIF formats are defined in RFC 2849. An example is as follows:

```
dn: ctxid=admin,cn=john,ou=employees,dc=example,dc=com
objectClass: ctxperson
ctxid: admin
cn: john
usrprof: shell:Admin=ROLE-1 DOMAIN-1
userPassword: xxxxxxxxx
```

**Step 4** Start the LDAP server, which is slapd in the case of OpenLDAP.
Chapter 2  Configuring Authentication and Accounting Services

Creating User Accounts

Every user associated with a virtual context has account information stored on the ACE. The authentication information, username, user password, password expiration date, and user role membership are all stored as part of each user’s profile.

As the ACE global administrator, you can assign one user in each context as the context administrator. The context administrator can then log in to the context or contexts on the ACE for which he or she is responsible and create additional users.

If you do not assign a user role to a new user, the default user role is Network-Monitor. By default, the user is allowed to operate on all domains. For users that you create in the Admin context, the default scope of access is the entire device. For users that you create in other contexts, the default scope of access is the entire context. If you need to restrict a user’s access, you must assign a role-domain pair.
Note the following when assigning a user for a context in the ACE:

- The same username can be created across contexts and can be associated with a unique role in a context and multiple domains. A user can have up to ten domains associated with a unique role in a context.
- Virtual contexts can share RADIUS, TACACS+, and LDAP servers; however, the user must be explicitly authenticated for each context and use the same password.
- All logged user accounting activities are distinguished in the ACE by the context in which a user has signed in.

For detailed information about creating contexts and user accounts to provide access to the local database on the ACE for CLI access authentication, see the *Cisco Application Control Engine Module Virtualization Configuration Guide*.

### Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server

You can specify one or more AAA server groups to identify the server and the remote authentication protocol, RADIUS, TACACS+, or LDAP. You can configure multiple AAA servers (of the same server type) for each server group.

For each AAA server, you can specify the following:

- The server IP address and port.
- Encryption key (shared secret) to authenticate communication between the ACE and AAA server (RADIUS and TACACS+ servers only).
- The number of times that the ACE retransmits an authentication request to a timed-out server before it declares the AAA server to be unresponsive and contacts the next AAA server in the group (RADIUS and TACACS+ servers only).
- The time interval that the ACE waits for a server to reply to an authentication request before retransmitting another request to the server.
- The time interval in which the ACE sends probes to a AAA server to verify whether the server is available and can receive authentication requests. The dead-time interval starts when the server does not respond to the number of authentication request transmissions.
• Independent server groups of TACACS+, RADIUS, or LDAP servers.

This section contains the following topics:
• Configuring RADIUS on the ACE
• Configuring TACACS+ on the ACE
• Configuring LDAP on the ACE
• Configuring AAA Server Groups

Configuring RADIUS on the ACE

The ACE supports the RADIUS protocol to communicate with a remote RADIUS server for authentication and accounting services. This section defines the configuration of the ACE to operate as a client of a RADIUS server.

This section contains the following topics:
• Setting the RADIUS Server Parameters
• Configuring the RADIUS NAS-IP-Address Attribute
• Setting the Global RADIUS Server Preshared Key
• Configuring the Global RADIUS Server Dead-Time Interval
• Setting the Global RADIUS Server Number of Retransmissions
• Setting the Global RADIUS Server Timeout Value

Setting the RADIUS Server Parameters

You can use the `radius-server host` command to specify the RADIUS server IP address, encrypted key, destination UDP port, and other options. You can also define multiple `radius-server host` commands to configure multiple RADIUS servers.

The syntax of this command is as follows:

```
radius-server host ip_address [key shared_secret [0 shared_secret | 7 shared_secret]] [auth-port port_number] [acct-port port_number] [authentication] [accounting] [timeout seconds] [retransmit count]
```

The arguments, keywords, and options are as follows:
- **ip_address** — IP address for the RADIUS server. Enter the address in dotted-decimal IP notation (for example, 192.168.11.1).

- **key** — (Optional) Enables an authentication key for communication between the ACE and the RADIUS daemon running on the RADIUS server. The key is a text string that must match the encryption key used on the RADIUS server. This key overrides the global setting of the `radius-server key` command. If you do not specify a key, the global value is used. RADIUS keys are always stored in encrypted form in persistent storage. The running configuration also displays keys in encrypted form.

- **shared_secret** — Key used to authenticate communication between the RADIUS client and server. The shared secret must match the one configured on the RADIUS server. Enter the shared secret as a case-sensitive string with no spaces with a maximum of 63 alphanumeric characters.

- **0** — (Optional) Configures a key specified in clear text (indicated by 0) to authenticate communication between the RADIUS client and server.

- **7** — (Optional) Configures a key specified in encrypted text (indicated by 7) to authenticate communication between the RADIUS client and server.

- **auth-port port_number** — (Optional) Specifies the UDP destination port for communicating authentication requests to the RADIUS server. By default, the RADIUS authentication port is 1812 (as defined in RFC 2138 and RFC 2139). If your RADIUS server uses a port other than 1812, use the `auth-port` keyword to configure the ACE for the appropriate port before you start the RADIUS service. The `port_number` argument is the RADIUS port number. Valid values are from 1 to 65535.

- **acct-port port_number** — (Optional) Specifies the UDP destination port for communicating accounting requests to the RADIUS server. By default, the RADIUS accounting port is 1813 (as defined in RFC 2138 and RFC 2139). If your RADIUS server uses a port other than 1813, use the `acct-port` keyword to configure the ACE for the appropriate port before you start the RADIUS service. The `port_number` argument is the RADIUS port number. Valid values are from 1 to 65535.

- **authentication** — (Optional) Specifies that the RADIUS server is used only for authentication.

---

**Note** If you do not specify either the authentication or accounting options, the RADIUS server is used for both accounting and authentication.
• **accounting**—(Optional) Specifies that the RADIUS server is used only for accounting.

    Note
    If you do not specify either the authentication or accounting options, the RADIUS server is used for both accounting and authentication.

• **timeout seconds**—(Optional) By default, the ACE waits 1 second for the RADIUS server to reply to an authentication request before retransmitting an authentication request to the server. Use the `timeout` keyword to change the time interval that the ACE waits for the RADIUS server to reply to an authentication request before retransmitting a request. Valid entries are from 1 to 60 seconds. The default is 1 second. For the specified server, this command overrides the global setting that was assigned by using the `radius-server timeout` command.

• **retransmit count**—(Optional) By default, the ACE send a single authentication request to a timed-out RADIUS server before it stops transmission and attempts to contact the next identified AAA server. The retransmit option is the number of times that the ACE retransmits an authentication request to a timed-out RADIUS server before it declares the server to be unresponsive and contacts the next server in the group. If all servers in the group are unavailable for authentication and accounting, the ACE tries the local database if you configured it as a local fallback method using the `aaa authentication login` or the `aaa accounting default` command. Valid entries are from 1 to 5 attempts. The default is 1 attempt. For the specified server, this command overrides the global setting that was assigned by using the `radius-server retransmit` command.

For example, to configure RADIUS server authentication parameters, enter:

```plaintext
host1/Admin(config)# radius-server host 192.168.2.3 key HostKey
host1/Admin(config)# radius-server host 192.168.2.3 key 7 secret_1256
host1/Admin(config)# radius-server host 192.168.2.3 auth-port 1645
host1/Admin(config)# radius-server host 192.168.2.3 acct-port 1646
host1/Admin(config)# radius-server host 192.168.2.3 authentication
host1/Admin(config)# radius-server host 192.168.2.3 accounting
host1/Admin(config)# radius-server host 192.168.2.3 timeout 25
host1/Admin(config)# radius-server host 192.168.2.3 retransmit 3
```

To revert to a default RADIUS server authentication setting, enter:

```plaintext
host1/Admin(config)# no radius-server host 192.168.2.3 acct-port 1646
```
Configuring the RADIUS NAS-IP-Address Attribute

Typically, RADIUS servers check the source IP address in the IP header of the RADIUS packets to track the source of the RADIUS requests. Also, some servers use the NAS-IP-Address RADIUS attribute to identify the RADIUS clients that can expose your ACE internal private network interface IP addresses.

By default, the NAS-IP-Address is not configured. The ACE performs a route lookup on the RADIUS server IP address and uses the result. Use the `radius-server attribute nas-ipaddr` command to specify a RADIUS NAS-IP-Address attribute. This attribute allows you to configure an arbitrary IP address to be used as RADIUS attribute 4, NAS-IP-Address for each context. The `radius-server attribute nas-ipaddr` command allows the ACE to behave as a single RADIUS client from the perspective of the RADIUS server. The configured NAS-IP-Address is encapsulated in all outgoing RADIUS authentication request and accounting packets.

The syntax of this command is as follows:

```
radius-server attribute nas-ipaddr nas_ip_address
```

The `nas_ip_address` argument configures an IP address to be used as the RADIUS NAS-IP-Address, attribute 4.

For example, to specify a RADIUS NAS-IP-Address, enter:

```
host1/Admin(config)# radius-server attribute nas-ipaddr 192.168.1.1
```

To delete the RADIUS NAS-IP-Address and return to the default configuration, enter:

```
host1/Admin(config)# no radius-server attribute nas-ipaddr 192.168.1.1
```

Setting the Global RADIUS Server Preshared Key

You can globally configure an authentication key for communication between the ACE and the RADIUS daemon running on each RADIUS server by using the `radius-server key` command. The key is a text string that must match the encryption key used on the RADIUS server. RADIUS keys are always stored in encrypted form in persistent storage on the ACE. This global key is applied to those RADIUS servers in a named server group for which a shared secret is not individually configured by the `radius-server host` command.

The syntax of this command is as follows:
Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server

Configuring Authentication and Accounting Services

Chapter 2

radius-server key \{shared_secret | 0 shared_secret | 7 shared_secret\}

The arguments and keywords are as follows:

- **shared_secret**—Key used to authenticate communication between the RADIUS client and server. The shared secret must match the one configured on the RADIUS server. Enter the shared secret as a case-sensitive string with no spaces and a maximum of 63 alphanumeric characters.

- **0**—Configures a key specified in clear text (indicated by 0) to authenticate communication between the RADIUS client and server.

- **7**—Configures a key specified in encrypted text (indicated by 7) to authenticate communication between the RADIUS client and server.

For example, to globally configure an authentication key to be sent in encrypted text (indicated by 7) to the RADIUS server, enter:

```
host1/Admin(config)# radius-server key 7 abe4DFeeweo00o
```

To delete the key, enter:

```
host1/Admin(config)# no radius-server key 7 abe4DFeeweo00o
```

Configuring the Global RADIUS Server Dead-Time Interval

During the dead-time interval, the ACE sends probe access-request packets to verify that the RADIUS server is available and can receive authentication requests. The dead-time interval starts when the server does not respond to the number of authentication request transmissions configured through either the `radius-server retransmit` command or the `radius-server host retransmit` command. When the server responds to a probe access-request packet, the ACE transmits the authentication request to the server.

Use the `radius-server deadtime` command to globally set the time interval in which the ACE verifies whether a nonresponsive server is operational.

This command causes the ACE to mark any RADIUS servers that fail to respond to authentication requests as dead. This action avoids the wait for the request to time out before trying the next configured server. The ACE skips a RADIUS server that is marked as dead by sending additional requests for the duration of the specified `minutes` argument.

The syntax of this command is as follows:

```
radius-server deadtime minutes
```
The *minutes* argument is the length of time that the ACE skips a nonresponsive RADIUS server for transaction requests. Valid entries are from 0 to 1440 minutes (24 hours). The default is 0.

For example, to globally configure a 15-minute dead-time interval for RADIUS servers that fail to respond to authentication requests, enter:

```
host1/Admin(config)# radius-server deadtime 15
```

To set the RADIUS server dead-time interval to 0, enter:

```
host1/Admin(config)# no radius-server deadtime 15
```

### Setting the Global RADIUS Server Number of Retransmissions

By default, the ACE sends one authentication request to a RADIUS server before it declares the server to be unresponsive and contacts the next server in the group. Use the `radius-server retransmit` command to globally change the number of times that the ACE sends an authentication request to a RADIUS server. If all servers in the group are unavailable for authentication and accounting, the ACE tries the local database if you configured it as a local fallback method using the `aaa authentication login` or the `aaa accounting default` command. If you do not have a fallback method, the ACE continues to contact one of the AAA servers listed in the server group.

The ACE applies this global retransmission value to those RADIUS servers for which a value is not individually configured by the `radius-server host` command.

The syntax of this command is as follows:

```
radius-server retransmit count
```

The `count` argument is the number of times that the ACE attempts to connect to a RADIUS server before trying to contact the next available server. The range is from 1 to 5 times. The default is 1.

For example, to globally configure the number of retransmissions to 3, enter:

```
host1/Admin(config)# radius-server retransmit 3
```

To revert to the default of one transmission attempt, enter:

```
host1/Admin(config)# no radius-server retransmit 3
```
Chapter 2  Configuring Authentication and Accounting Services

Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server

### Setting the Global RADIUS Server Timeout Value

By default, the ACE waits 1 second for the RADIUS server to send a reply to an authentication request to an unresponsive server before retransmitting an authentication request to the server. Use the `radius-server timeout` command to globally change the time interval that the ACE waits for the RADIUS server to reply before retransmitting an authentication request to the RADIUS server. The ACE applies this global timeout value to those RADIUS servers for which a timeout value is not individually configured by the `radius-server host` command.

The syntax of this command is as follows:

```
radius-server timeout seconds
```

The `seconds` argument is the time in seconds between retransmissions to the RADIUS server. Valid entries are from 1 to 60 seconds. The default is 1 second.

For example, to globally configure the timeout value to 30 seconds, enter:

```
host1/Admin(config)# radius-server timeout 30
```

To revert to the default of 1 second between transmission attempts, enter:

```
host1/Admin(config)# no radius-server timeout 30
```

### Configuring TACACS+ on the ACE

The ACE supports the TACACS+ protocol to communicate with a TACACS+ server for authentication and accounting services. This section defines the configuration of the ACE to operate as a client of a TACACS+ server.

This section contains the following topics:

- Setting the TACACS+ Server Parameters
- Setting the Global Preshared Key
- Setting the Global TACACS+ Server Dead-Time Interval
- Setting the Global TACACS+ Server Timeout Value
Setting the TACACS+ Server Parameters

You can use the `tacacs-server host` command to specify the TACACS+ server IP address, encrypted key, destination port, and other options. You can define multiple `tacacs-server host` commands to configure multiple TACACS+ servers.

The syntax of this command is as follows:

```
tacacs-server host ip_address [key [0 | 7] shared_secret]] [port port_number] [timeout seconds]
```

The arguments, keywords, and options are as follows:

- `ip_address` — IP address for the TACACS+ server. Enter the address in dotted-decimal IP notation (for example, 192.168.11.1).
- `key` — (Optional) Enables an authentication key for communication between the ACE and the daemon that runs on the TACACS+ server. The key is a text string that must match the encryption key used on the TACACS+ server. This key overrides the global setting of the `tacacs-server key` command. If you do not specify a key, the global value is used. TACACS+ keys are always stored in encrypted form in persistent storage. The running configuration also displays keys in encrypted form.
- `0` — (Optional) Configures a key specified in clear text (indicated by 0) to authenticate communication between the TACACS+ client and server.
- `7` — (Optional) Configures a key specified in encrypted text (indicated by 7) to authenticate communication between the TACACS+ client and server.
- `shared_secret` — Key used to authenticate communication between the TACACS+ client and server. The shared secret must match the one configured on the TACACS+ server. Enter the shared secret as a case-sensitive string with no spaces and a maximum of 63 alphanumeric characters. Alternatively, you can use spaces if you enclose the entire string in quotation marks.
- `port port_number` — (Optional) Specifies the TCP destination port for communicating authentication requests to the TACACS+ server. By default, the TACACS+ authentication port is 49 (as defined in RFC 1492). If your TACACS+ server uses a port other than 49, use the `port` keyword to configure the ACE for the appropriate port prior to starting the TACACS+ service. The `port_number` argument specifies the TACACS+ port number. Valid values are from 1 to 65535.
• **timeout seconds**—(Optional) By default, the ACE waits 1 second for the TACACS+ server to reply to an authentication request before it declares a timeout failure and attempts to contact the next server in the group. If all servers in the group are unavailable for authentication and accounting, the ACE tries the local database if you configured it as a local fallback method using the `aaa authentication login` or the `aaa accounting default` command. Use the `timeout` keyword to change the time interval that the ACE waits for the TACACS+ server to reply to an authentication request. Valid entries are from 1 to 60 seconds. The default is 1 second. For the specified server, this command overrides the global setting that was assigned by using the `tacacs-server timeout` command.

For example, to configure TACACS+ server authentication parameters, enter:

```
host1/Admin(config)# tacacs-server host 192.168.3.2 key HostKey
host1/Admin(config)# tacacs-server host 192.168.3.2 port 1645
host1/Admin(config)# tacacs-server host 192.168.3.2 timeout 5
```

To remove the TACACS+ server from the configuration, enter:

```
host1/Admin(config)# no tacacs-server host 192.168.3.2 key HostKey
```

### Setting the Global Preshared Key

You can globally configure an authentication key for communication between the ACE and the TACACS+ daemon that runs on each TACACS+ server by using the `tacacs-server key` command. The key is a text string that must match the encryption key used on the TACACS+ server. TACACS+ keys are always stored in encrypted form in persistent storage on the ACE. This global key is applied to those TACACS+ servers in a named server group for which a shared secret is not individually configured by the `tacacs-server host` command.

The syntax of this command is as follows:

```
tacacs-server key [0 | 7] shared_secret [timeout seconds]
```

The arguments and keywords are as follows:

• **shared_secret**—Key used to authenticate communication between the TACACS+ client and server. The shared secret must match the one configured on the TACACS+ server. Enter the shared secret as a case-sensitive string with no spaces with a maximum of 63 alphanumeric characters or you can enter spaces if you enclose the entire key with quotation marks (for example, “my key”).
• **0**—(Optional) Configures a key specified in clear text (indicated by 0) to authenticate communication between the TACACS+ client and server.

• **7**—(Optional) Configures a key specified in encrypted text (indicated by 7) to authenticate communication between the TACACS+ client and server.

• **timeout seconds**—(Optional) Globally configures the time interval that the ACE waits for the TACACS+ server to reply before retransmitting an authentication request to the TACACS+ server. The *seconds* argument is the timeout value in seconds. Valid entries are from 1 to 60 seconds. By default, the ACE waits 1 second to receive a response from a TACACS+ server before it declares a timeout failure and attempts to contact the next server in the group. This option configures the same time interval as the `tacacs-server timeout` command.

For example, to globally configure an authentication key in encrypted text (indicated by 7) to authenticate communication between the TACACS+ client and server, enter:

```
host1/Admin(config)# tacacs-server key 7 abe4DFeeweo00o
```

To delete the key, enter:

```
host1/Admin(config)# no tacacs-server key 7 abe4DFeeweo00o
```

### Setting the Global TACACS+ Server Dead-Time Interval

During the dead-time interval, the ACE sends probe access-request packets to verify that the TACACS+ server is available and can receive authentication requests. The dead-time interval starts when the server does not respond to an authentication request transmission. When the server responds to a probe access-request packet, the ACE retransmits the authentication request to the server.

Use the `tacacs-server deadtime` command to globally set the time interval in which the ACE verifies whether a nonresponsive server is operational.

This command causes the ACE to mark any TACACS+ servers that fail to respond to authentication requests as dead. This action avoids the wait for the request to time out before trying the next configured server. The ACE skips a TACACS+ server that is marked as dead by sending additional requests for the duration of the *minutes* argument.

The syntax of this command is as follows:
**tacacs-server deadtime minutes**

The *minutes* argument is the length of time that the ACE skips a nonresponsive TACACS+ server for transaction requests. Valid entries are from 0 to 1440 minutes (24 hours). The default is 0.

For example, to globally configure a 15-minute dead-time interval for TACACS+ servers that fail to respond to authentication requests, enter:

```
host1/Admin(config)# tacacs-server deadtime 15
```

To set the TACACS+ server dead-time interval to 0, enter:

```
host1/Admin(config)# no tacacs-server deadtime 15
```

### Setting the Global TACACS+ Server Timeout Value

By default, the ACE waits 1 second to receive a response from a TACACS+ server before it declares a timeout failure and attempts to contact the next server in the group. Use the **tacacs-server timeout** command to globally change the time interval that the ACE waits for the TACACS+ server to reply before retransmitting an authentication request to the TACACS+ server. The ACE applies this global timeout value to those TACACS+ servers for which a timeout value is not individually configured by the **tacacs-server host** command.

The syntax of this command is as follows:

**tacacs-server timeout seconds**

The *seconds* argument is the timeout value in seconds. Valid entries are from 1 to 60 seconds. The default is 1 second.

For example, to globally configure the timeout value to 30 seconds, enter:

```
host1/Admin(config)# tacacs-server timeout 30
```

To revert to the default of 1 second between transmission attempts, enter:

```
host1/Admin(config)# no tacacs-server timeout 30
```

### Configuring LDAP on the ACE

The ACE supports the LDAP protocol to communicate with a remote LDAP directory server for authentication services. This section defines the configuration of the ACE to operate as a client of an LDAP server.
This section contains the following topics:

- Setting the LDAP Server Parameters
- Setting the Global LDAP Server Port Setting
- Setting the Global LDAP Server Timeout Value

### Setting the LDAP Server Parameters

You can use the `ldap-server host` command to specify the LDAP server hostname or IP address, destination port, and other options. You can define multiple `ldap-server host` commands to configure multiple LDAP servers.

The syntax of this command is as follows:

```
ldap-server host ip_address [port port_number] [timeout seconds]
    [rootDN "DN_string" [password bind_password]]
```

The arguments, keywords, and options are as follows:

- **ip_address** —IP address for the LDAP server. Enter the address in dotted-decimal IP notation (for example, 192.168.11.1).

- **port port_number**—(Optional) Specifies the TCP destination port for communicating authentication requests to the LDAP directory server. By default, the LDAP server port is 389. If your LDAP server uses a port other than 389, use the `port` keyword to configure the ACE for the appropriate port before you start the LDAP service. The `port_number` argument is the LDAP port number. Valid values are from 1 to 65535. For the specified server, this command overrides the global setting that was assigned by using the `ldap-server port` command.

- **timeout seconds**—(Optional) Specifies the time in seconds to wait for a response from the LDAP server before the ACE can declare a timeout failure with the LDAP server. By default, the ACE waits 5 seconds for the LDAP server to reply to an authentication request before the ACE declares a timeout failure and attempts to contact the next server in the group. Use the `timeout` keyword to change the time interval that the ACE waits for the LDAP server to reply to an authentication request. Valid entries are from 1 to 60 seconds. The default is 5 seconds. For the specified server, this command overrides the global setting that was assigned by using the `ldap-server timeout` command.
Chapter 2  Configuring Authentication and Accounting Services

Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server

- **rootDN “DN_string”**—(Optional) Defines the distinguished name (DN) for a user who is unrestricted by access controls or administrative limit parameters to perform operations on the LDAP server directory. The rootDN user is the root user for the LDAP server database. Enter a quoted string that has a maximum of 63 alphanumeric characters. The default is an empty string.

- **password bind_password**—(Optional) Defines the bind password (rootpw) applied to the rootDN of the LDAP server directory. Enter an unquoted string that has a maximum of 63 alphanumeric characters. The default is an empty string.

For example, to configure LDAP server authentication parameters, enter:

```
host1/Admin(config)# ldap-server host 192.168.2.3 port 2003
host1/Admin(config)# ldap-server host 192.168.2.3 timeout 60
host1/Admin(config)# ldap-server host 192.168.2.3 rootDN
   "cn=manager,dc=cisco,dc=com" password lab
```

To remove the LDAP server authentication setting, enter:

```
host1/Admin(config)# no ldap-server host 192.168.2.3
```

**Setting the Global LDAP Server Port Setting**

By default, the TCP destination port for communicating authentication requests to the LDAP directory server is 389. If your LDAP server uses a port other than port 389, use the `ldap-server port` command to globally configure the ACE for the appropriate port before you start the LDAP service. This global port setting will be applied to those LDAP servers for which a TCP port value is not individually configured by the `ldap-server host` command.

The syntax of this command is as follows:

```
ldap-server port port_number
```

The `port_number` argument is the destination port to the LDAP server. Valid values are from 1 to 65535. The default is TCP port 389.

For example, to globally configure the TCP port, enter:

```
host1/Admin(config)# ldap-server port 2003
```

To revert to the default of TCP port 389, enter:

```
host1/Admin(config)# no ldap-server port 2003
```
Setting the Global LDAP Server Timeout Value

By default, the ACE waits 5 seconds to receive a response from an LDAP server before it declares a timeout failure and attempts to contact the next server in the group. Use the `ldap-server timeout` command to globally change the time interval that the ACE waits for the LDAP server to reply to a response before it declares a timeout failure. The ACE applies this global timeout value to those LDAP servers for which a timeout value is not individually configured by the `ldap-server host` command.

The syntax of this command is as follows:

```
ldap-server timeout seconds
```

The `seconds` argument is the timeout value in seconds. Valid entries are from 1 to 60 seconds. The default is 5 seconds.

For example, to globally configure the timeout value to 30 seconds, enter:
```
host1/Admin(config)# ldap-server timeout 30
```

To change to the default of 5 seconds between transmission attempts, enter:
```
host1/Admin(config)# no ldap-server timeout 30
```

Configuring AAA Server Groups

This section contains the following topics:

- Creating a TACACS+, RADIUS, or LDAP Server Group
- Setting the Dead-Time Interval for a TACACS+ Server Group
- Setting the Dead-Time Interval for a RADIUS Server Group
- Configuring the User Profile Attribute Type for an LDAP Server Group
- Configuring the Base DN for an LDAP Server Group
- Configuring the Search Filter for an LDAP Server Group
Creating a TACACS+, RADIUS, or LDAP Server Group

A server group is a list of server hosts of a particular type. The ACE allows you to configure multiple TACACS+, RADIUS, and LDAP servers as a named server group. You group the different AAA server hosts into distinct lists. The ACE searches for the server hosts in the order in which you specify them within a group.

Use the `aaa group server` command to configure independent server groups of TACACS+, RADIUS, or LDAP servers. You can configure server groups at any time, but they only take effect when you apply them to the AAA service using the `aaa authentication login` or the `aaa accounting default` commands.

You can configure a maximum of 10 server groups for each context in the ACE. The ACE attempts to contact the first server listed in the server group for user authentication and accounting. If that server is unavailable, the ACE attempts to contact the next configured server listed in the group. If all servers in the group are unavailable, the ACE then tries the servers in the next configured server group. The ACE repeats this process until the authentication request can be handled by an AAA server. If the specified AAA servers in a server group are unavailable, and you specify local authentication as the fallback method (as specified in the `aaa authentication login` command), the ACE attempts to authenticate the user against the local database on the ACE. If you do not have a fallback method, the ACE continues to contact one of the AAA servers listed in the server group.

The syntax of this command is as follows:

```
aaa group server {ldap | radius | tacacs+} group_name
```

The arguments and keywords are as follows:

- **ldap**—Specifies an LDAP directory server group.
- **radius**—Specifies a RADIUS server group.
- **tacacs+**—Specifies a TACACS+ server group.
- **group_name**—Group of servers. The server group name is a maximum of 64 alphanumeric characters with no spaces.

The CLI displays the TACACS+, RADIUS, or LDAP server configuration mode where you identify the name of one or more previously configured servers that you want added to the server group.

The syntax of this server configuration mode command is as follows:
server ip_address

The ip_address argument is the IP address for an existing RADIUS, TACACS+, or LDAP server that you want to add to the server group. Enter the address in dotted-decimal IP notation (for example, 192.168.11.1). You can add multiple servers to the server group by entering multiple server commands while in server configuration mode. The same server can belong to multiple server groups.

For example, to create a RADIUS server group, enter:

```plaintext
host1/Admin(config)# aaa group server radius RAD_Server_Group1
host1/Admin(config-radius)# server 192.168.252.1
host1/Admin(config-radius)# server 192.168.252.2
host1/Admin(config-radius)# server 192.168.252.3
```

To remove a server from a server group, enter:

```plaintext
host1/Admin(config-radius)# no server 192.168.252.3
```

To remove a server group, enter:

```plaintext
host1/Admin(config)# no aaa group server radius RAD_Server_Group1
```

For the TACACS+, RADIUS, and LDAP server groups, you can also configure the following parameters:

- For a TACACS+ server group, you can specify a dead-time interval for the server group. See the “Setting the Dead-Time Interval for a TACACS+ Server Group” section.

- For a RADIUS server group, you can specify a dead-time interval for the server group. See the “Setting the Dead-Time Interval for a RADIUS Server Group” section.

- For an LDAP server group, you may specify the following parameters:
  - User profile attribute—See the “Configuring the User Profile Attribute Type for an LDAP Server Group” section.
  - Base DN—See the “Configuring the Base DN for an LDAP Server Group” section.
  - LDAP search filter—See the “Configuring the Search Filter for an LDAP Server Group” section.
Setting the Dead-Time Interval for a TACACS+ Server Group

For a TACACS+ server group, you can specify a dead-time interval for the server group. During the dead-time interval, the ACE sends probe access-request packets to verify that the TACACS+ server is available and can receive authentication requests. The dead-time interval starts when the server does not respond to an authentication request transmission. When the server responds to a probe access-request packet, the ACE retransmits the authentication request to the server.

Use the deadtime command to globally set the time interval in which the ACE verifies whether a nonresponsive server group is operational.

This command causes the ACE to mark any TACACS+ servers that fail to respond to authentication requests as dead. This action avoids the wait for the request to time out before trying the next configured server. The ACE skips a TACACS+ server that is marked as dead by sending additional requests for the duration of the minutes argument.

The syntax of this command is as follows:

```
deadtime
```

```
deadtime minutes
```

The minutes argument is the length of time that the ACE skips a nonresponsive TACACS+ server for transaction requests. Valid entries are from 0 to 1440 minutes (24 hours). The default is 0.

For example, to globally configure a 15-minute dead-time interval for TACACS+ servers that fail to respond to authentication requests, enter:

```
host1/Admin(config-tacacs)# deadtime 15
```

To reset the RADIUS server dead-time interval to 0, enter:

```
host1/Admin(config-tacacs)# no deadtime 15
```

Setting the Dead-Time Interval for a RADIUS Server Group

For a RADIUS server group, you can specify a dead-time interval for the server group. During the dead-time interval, the ACE sends probe access-request packets to verify that the RADIUS server is available and can receive authentication requests. The dead-time interval starts when the server does not respond to an authentication request transmission. When the server responds to a probe access-request packet, the ACE retransmits the authentication request to the server.

Use the deadtime command to globally set the time interval in which the ACE verifies whether a nonresponsive server group is operational.

This command causes the ACE to mark any RADIUS servers that fail to respond to authentication requests as dead. This action avoids the wait for the request to time out before trying the next configured server. The ACE skips a RADIUS server that is marked as dead by sending additional requests for the duration of the minutes argument.

The syntax of this command is as follows:

```
deadtime
```

```
deadtime minutes
```

The minutes argument is the length of time that the ACE skips a nonresponsive RADIUS server for transaction requests. Valid entries are from 0 to 1440 minutes (24 hours). The default is 0.

For example, to globally configure a 15-minute dead-time interval for RADIUS servers that fail to respond to authentication requests, enter:

```
host1/Admin(config-tacacs)# deadtime 15
```

To reset the RADIUS server dead-time interval to 0, enter:

```
host1/Admin(config-tacacs)# no deadtime 15
```
authentication request transmissions. When the server responds to a probe access-request packet, the ACE retransmits the authentication request to the server.

Use the `deadtime` command to globally set the time interval in which the ACE verifies whether a nonresponsive server group is operational.

This command causes the ACE to mark any RADIUS servers that fail to respond to authentication requests as dead. This action avoids the wait for the request to time out before trying the next configured server. The ACE skips a RADIUS server that is marked as dead by sending additional requests for the duration of the `minutes` argument.

The syntax of this command is as follows:

```
deadtime minutes
```

The `minutes` argument is the length of time that the ACE skips a nonresponsive RADIUS server for transaction requests. Valid entries are from 0 to 1440 minutes (24 hours). The default is 0.

For example, to globally configure a 15-minute dead-time interval for RADIUS servers that fail to respond to authentication requests, enter:

```
host1/Admin(config-radius)# deadtime 15
```

To reset the RADIUS server dead-time interval to 0, enter:

```
host1/Admin(config-radius)# no deadtime 15
```

### Configuring the User Profile Attribute Type for an LDAP Server Group

An LDAP server retrieves a user’s profile as part of the search request. During a search request, the LDAP client requests the user profile attribute from the LDAP server by including this attribute type (the configured string) in the search request. The search request must match the attribute type used by the LDAP server to properly identify the user profile attribute, as defined in private schema on the LDAP server. The LDAP server uses the search filter to locate the user profile entry in its database. When the LDAP server finds the entry, it replies with a search response in which it includes the value of the user profile attribute that was stored in that entry. This value contains the role and domain pair of the user for that context.

You define the user profile attribute value in the following format:
Chapter 2 Configuring Authentication and Accounting Services

Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server

Chapter 2 Configuring Authentication and Accounting Services

Configuring the ACE as a Client of a RADIUS, TACACS+, or LDAP Server

--

shell:<contextname>=<role> <domain1> <domain2>...<domainN>

Note

The user profile attribute serves an important configuration function for an LDAP server group. If the user profile attribute is not obtained from the server during authentication, or if the profile is obtained from the server but the context name(s) in the profile do not match the context in which the user is trying to log in, a default role (Network-Monitor) and a default domain (default-domain) are assigned to the user if the authentication is successful.

This attribute type is used for the user profile attribute. Since this attribute type is private, the LDAP server database should use the same attribute type for the user profile. The LDAP client (the ACE) sends the search request with this attribute type as the attribute it wants to download. If the lookup was successful, the search response contains this attribute value. The attribute value must contain a string that represents the user role and domain pair for this particular context.

Use the attribute user-profile command to specify which user profile attribute to use by the LDAP server.

You can configure the LDAP user profile attribute at the subconfiguration level for the LDAP server group (created as described in the “Configuring AAA Server Groups” section).

The syntax of this command is as follows:

attribute user-profile text

The text argument is the user profile. The user profile is an unquoted text string of a maximum of 63 alphanumeric characters without spaces.

For example, to configure an LDAP user profile attribute, enter:

host1/Admin(config-ldap)# attribute user-profile usrprof

To delete the user profile attribute, enter:

host1/Admin(config-ldap)# no attribute user-profile usrprof

Configuring the Base DN for an LDAP Server Group

When you create an LDAP server group, the top level of the LDAP directory tree is the base, referred to as the base DN. The base DN is used to perform the search operation in the LDAP server directory. A base DN can take a form such as
“dc=your,dc=domain”, where the base DN uses the DNS domain name as its basis and is split into the domain components. Use the base-DN server group command to configure the base DN that you want to use to perform search operations in the LDAP directory tree.

Note

The base DN is a mandatory configuration for an LDAP server group. Without this setting, a user cannot be authenticated.

You configure the base DN at the submode for the LDAP server group (created as described in the “Configuring AAA Server Groups” section).

The syntax of this command is as follows:

```
base-DN text
```

The `text` argument is the distinguished name of the search base. The base DN name is a quoted text string of a maximum of 63 alphanumeric characters without spaces.

For example, to configure the base DN, enter:

```
host1/Admin(config)# aaa group server ldap LDAP_Server_Group1
host1/Admin(config-ldap)# base-DN "dc=sns,dc=cisco,dc=com"
```

To delete the configured base DN, enter:

```
host1/Admin(config-ldap)# no base-DN "dc=sns,dc=cisco,dc=com"
```

### Configuring the Search Filter for an LDAP Server Group

For an LDAP server group, the ACE transmits a search filter to the LDAP server to look up a user in the database. Search filters enable you to define search criteria and provide more efficient and effective searches. The search filter is used in the search request sent by the LDAP client to the server to locate the user's node in the DIT. Use the `filter search-user` command to configure the exact filter to use. The $user and $contextid are substituted with actual values when sending the request.
The search filter should follow the format defined in RFC 2254. The LDAP client sends the search request with the configured search filter after replacing the $userid and $contextid with the userid that the client is trying to authenticate and the associated virtual context name. The ACE allows $userid and $contextid to be used as placeholders for the user ID and the context name.

**Note**

The search filter is a mandatory configuration for an LDAP server group. Without this setting, a user cannot be authenticated.

You configure the LDAP search filter at the subconfiguration level for the LDAP server group (created as described in the “Configuring AAA Server Groups” section).

The syntax of this command is as follows:

```
filter search-user text
```

The `text` argument is the search request. The search filter is a quoted text string of a maximum of 63 alphanumeric characters without spaces.

For example, to configure a search request, enter:

```
host1/Admin(config)# aaa group server ldap LDAP_Server_Group1
host1/Admin(config-ldap)# filter search-user "(&(objectclass=person) (&(cn=$userid)(cid=$contextid)))"
```

To delete the search request, enter:

```
host1/Admin(config-ldap)# no filter search-user "(&(objectclass=person)(&(cn=$userid)(cid=$contextid)))"
```

---

**Defining the Login Authentication Method**

Authentication is the process of verifying the identity of the person attempting to log in to the ACE CLI by console port or by a Telnet or SSH session. This identity verification is based on the username and password combination provided by the person attempting to access the ACE.
The ACE supports local authentication using the lookup database on the ACE or remote authentication using one or more TACACS+, RADIUS, or LDAP servers. You can specify the local database on the ACE as the fallback authentication method in case the configured AAA servers fail to respond to the authentication request.

The default login method of user authentication is by console port or by a Telnet or SSH session. You can override the default login authentication method and specify authentication through only the console port.

To configure the authentication method used for login to the ACE CLI, use the `aaa authentication login` command in configuration mode.

The syntax of this command is as follows:

```
aaa authentication login {{console | default} {{group group_name}
    {local} {none}}} | error-enable
```

The arguments, keywords, and options are as follows:

- **console**—Specifies the console port login authentication method, identified by the specified server group.
- **default**—Specifies the default login authentication method (console port or by a Telnet or SSH session), identified by the specified server group.
- **group group_name**—Associates the login authentication process with a TACACS+, RADIUS, or LDAP server defined through the `aaa group server` command. The server group name is a maximum of 64 alphanumeric characters with no spaces.
- **local**—Specifies to use the local database on the ACE as the login authentication method. If the server does not respond, then the local database is used as the fallback authentication method.
- **none**—Specifies that the ACE does not perform password verification. If you configure this option, users can log in to the ACE without providing a valid password. Only a user with an Admin role is allowed to specify the `none` option.

⚠️ **Caution**

Use this option with care. If you specify `none`, any user will be able to access the ACE at any time.
- **error-enable**—Enables the display of the login error message in instances where the remote AAA servers fail to respond. To view the current display status, use the `show aaa authentication login error-enable` command. When a user attempts to log in, and the remote AAA servers do not respond to the authentication request, the ACE processes the login sequence by switching to a local user database. If you activate the error-enabled feature, the following message appears on the user’s terminal:

Remote AAA servers unreachable; local authentication done.

For example, to enable console authentication using the TacServers server group, followed by local login as the fallback method, enter:

```
host1/Admin(config)# aaa authentication login console group TacServers local
```

Password verification remains enabled for login authentication.

For example, to turn off password validation, enter:

```
host1/Admin(config)# aaa authentication login console group TacServers local none
```

For example, to revert to the local authentication method, enter:

```
host1/Admin(config)# no aaa authentication login console group TacServers local none
```

### Defining the Default Accounting Method

Accounting refers to the log information that is maintained for each user’s management session with an ACE. This information may be used to generate reports for troubleshooting and auditing purposes. Accounting can be implemented locally on the ACE or remotely using a RADIUS or TACACS+ server.

Use the `aaa accounting default` command to configure the default accounting method. You specify either a previously created AAA server group that identifies separate groups of TACACS+ or RADIUS servers or the local database on the ACE.

The syntax of this command is as follows:

```
aaa accounting default {group group_name} {local} {none}
```
The arguments and keywords are as follows:

- **group group_name**—Associates the accounting method with a TACACS+ or RADIUS server defined previously through the `aaa group server` command. The server group name is a maximum of 64 alphanumeric characters with no spaces.

- **local**—Specifies to use the local database on the ACE as the accounting method.

- **none**—Specifies that the ACE does not perform password verification, which disables password verification. If you configure this option, users can log in without providing a valid password.

⚠️ **Caution**

Use this option with care. If configured, any user will be able to access the ACE at any time.

For example, to enable user accounting to be performed using remote TACACS+ servers, followed by local login as the fallback method, enter:

```
host1/Admin(config-context)# aaa accounting default group TacServers local
```

To revert to the default local accounting method, enter:

```
host1/Admin(config-context)# no aaa accounting default group TacServers local
```

### Viewing AAA Status and Statistics

This section contains the following topics:

- Displaying AAA Groups
- Displaying RADIUS Server Configuration Information
- Displaying TACACS+ Server Configuration Information
- Displaying LDAP Server Configuration Information
- Displaying Accounting Configuration Information
- Displaying Accounting Log Information
- Displaying Authentication Configuration Information
Displaying AAA Groups

You can display the configured server groups by using the `show aaa groups` command. The syntax of this command is as follows:

```
show aaa groups
```

For example, to display configured server groups, enter:

```
host1/Admin# show aaa groups
TACACS:
   TACACS_group1
RADIUS:
   RAD_group1
LDAP:
   LDAP_group2
```

Displaying RADIUS Server Configuration Information

You can display the configured RADIUS server and group parameters by using the `show radius-server` command.

The syntax of this command is as follows:

```
show radius-server [groups | sorted]
```

The optional keywords are as follows:

- `groups`—(Optional) Displays configured RADIUS server group information.
- `sorted`—(Optional) Displays RADIUS server information sorted by name.

For example, to display configured RADIUS server parameters, enter:

```
host1/Admin# show radius-server
retransmission count:1
timeout value:1
deadtime value:20
total number of servers:2

following RADIUS servers are configured:
   192.168.34.45:
      available for authentication on port:1812
      available for accounting on port:1813
   192.168.2.3:
      available for authentication on port:1812
```
available for accounting on port:1813
RADIUS shared secret:********

For example, to display the configured RADIUS server groups, enter:

host1/Admin# show radius-server groups
total number of groups:2

following RADIUS server groups are configured:
group radius:
  server: all configured radius servers
group RAD_Server_Group:
  deadtime is 0

For example, to display the sorted RADIUS server groups, enter:

host1/Admin# show radius-server sorted
retransmission count:1
timeout value:1
deadtime value:20
total number of servers:2

following RADIUS servers are configured:
192.168.34.45:
  available for authentication on port:1812
  available for accounting on port:1813
192.168.2.3:
  available for authentication on port:1812
  available for accounting on port:1813
  RADIUS shared secret:********

Displaying TACACS+ Server Configuration Information

You can display the configured TACACS+ server and group parameters by using the show tacacs-server command.

The syntax of this command is as follows:

    show tacacs-server [groups | sorted]

The optional keywords are as follows:

- **groups**—(Optional) Displays configured TACACS+ server group information.
- **sorted**—(Optional) Displays TACACS+ server information sorted by name.
For example, to display the configured TACACS+ server parameters, enter:

```
host1/Admin# show tacacs-server
    Global TACACS+ shared secret: tacacspword
    timeout value: 30
    total number of servers: 3

    following TACACS+ servers are configured:
    192.168.58.91:
      available on port: 2
    cisco.com:
      available on port: 49
    192.168.22.95:
      available on port: 49
    TACACS+ shared secret: MyKey
```

For example, to display the configured TACACS+ server groups, enter:

```
host1/Admin# show tacacs-server groups
    total number of groups: 1

    following TACACS+ server groups are configured:
    group TacServers:
      server 192.168.58.91 on port 2
```

For example, to display the sorted TACACS+ servers, enter:

```
host1/Admin# show tacacs-server sorted
    timeout value: 1
    total number of servers: 1
```

### Displaying LDAP Server Configuration Information

You can display the configured LDAP server and server group parameters by using the `show ldap-server` command.

The syntax of this command is as follows:

```
show ldap-server [groups]
```

The optional `groups` keyword displays configured LDAP server group information.

To display configured LDAP server parameters, enter:

```
host1/Admin# show ldap
    timeout : 5
    port : 389
```
total number of servers : 1

To display the configured LDAP server groups, enter:

```
host1/Admin# show ldap-server groups
```
total number of groups: 1

following LDAP server groups are configured:
group LDAP_Server_Group1:
    baseDN: "dc=sns,dc=cisco,dc=com"
    user profile attribute: usprof
    search filter: "(&(objectclass=person)
                   (&(cn=$_userid)(cid=$_contextid)))"

Displaying Accounting Configuration Information

You can display accounting configuration information for the ACE by using the `show aaa accounting` command.

The syntax of this command is as follows:

```
show aaa accounting
```

For example, to display accounting configuration information, enter:

```
host1/Admin# show aaa accounting
default: local
```

Displaying Accounting Log Information

You can display accounting log information for the ACE by using the `show accounting log` command.

The syntax of this command is as follows:

```
show accounting log [size] [all]
```

The argument and option are as follows:

- `size`—(Optional) The size of the local accounting log file to display in bytes from 0 to 250000. The default is 250000 bytes.
- `all`—(Optional) Displays the accounting logs of all contexts in the ACE. This option is available only in the Admin context.
For example, to display accounting log information, enter:

```
host1/Admin# show accounting log
Sat Jan  1 00:02:55 2000:start:/dev/ttyS00_946684975:admin:
Sat Nov  5 00:20:04 2005:update:/dev/ttyS00_946684975:admin:0:ft
interface vlan
 50
Sat Nov  5 00:20:05 2005:update:/dev/ttyS00_946684975:admin:1:ip
  address 12.1.1.2
2 255.255.255.0
Sat Nov  5 00:20:05 2005:update:/dev/ttyS00_946684975:admin:1:peer
  ip 12.1.1.1 2
55.255.255.0
Sat Nov  5 00:20:05 2005:update:/dev/ttyS00_946684975:admin:1:no
shutdown
Sat Nov  5 00:20:12 2005:update:/dev/ttyS00_946684975:admin:0:ft
peer 1
Sat Nov  5 00:20:12
2005:update:/dev/ttyS00_946684975:admin:0:ft-interface vlan
 50
Sat Nov  5 00:20:41 2005:update:/dev/ttyS00_946684975:admin:0:log
  console 6
Sat Nov  5 00:20:58 2005:update:/dev/ttyS00_946684975:admin:0:ft
  group 1
Sat Nov  5 00:20:58 2005:update:/dev/ttyS00_946684975:admin:0:peer
  1
Sat Nov  5 00:20:58
2005:update:/dev/ttyS00_946684975:admin:0:priority 50
Sat Nov  5 00:20:58
2005:update:/dev/ttyS00_946684975:admin:0:associate-context
  Admin
Sat Nov  5 00:20:58
2005:update:/dev/ttyS00_946684975:admin:0:inservice
  .
  .
```

The ACE now includes the following configuration mode commands in the accounting logs:

- `[no] ldap-server host ip_address [port port_number] [timeout seconds] [rootDN "DN_string" [password bind_password]]`
- `[no] radius-server key [0 | 7] shared_secret`
- `[no] radius-server host ip_address key [0 | 7] shared_secret`
- `[no] snmp-server community community_name`
- `[no] snmp-server host ip_address [inform | traps] [version { 1 | 2c } | { 3 { auth | noauth | priv } }] community_string_or_username`
Previously, the ACE omitted these commands from the logs because they contain sensitive information, such as a community name, shared secret, username, or password.

With this behavior change, when the ACE includes any of these commands in the log, it masks the sensitive information with five stars. For example, when you enter the `snmp-server community community_name` command, the ACE logs the following:

```
snmp-server community *****
```

**Note**
The ACE logs the sensitive information for the following commands in plain text and does not mask it:

- The `backup pass-phrase` command in Exec mode
- The `ip address` command in KAL-AP UDP configuration mode
- The `credentials` command in probe RADIUS configuration mode

The ACE does not save the following CLI commands in the accounting log because the commands are handled by VSH and they do not go through the accounting framework:

- `clear screen`
- `config terminal`
- `(no) debug`
- `end`
- `exit`
- `show debug`
- `show terminal`
- `terminal (no)`
Displaying Authentication Configuration Information

You can display authentication configuration information for the ACE by using the `show aaa authentication` command.

The syntax of this command is as follows:

```
show aaa authentication [login error-enable]
```

The optional `login error-enable` keyword allows you to view the current display status of the login error message.

For example, to display the configured authentication parameters, enter:

```
host1/Admin# show aaa authentication
   default: group TacServers local none
   console: local
```

```
host1/Admin# show aaa authentication login error-enable
   enabled
```
Viewing AAA Status and Statistics
CHAPTER 3

Configuring Application Protocol Inspection

This chapter describes how to configure application protocol inspection for the Cisco Application Control Engine (ACE) module. Application protocol inspection provides functionality for several protocols that carry Layer 3 and Layer 4 information in the application payload, require some form of deep packet inspection of the HTTP protocol, or require File Transfer Protocol (FTP) request command filtering.

This chapter contains the following major sections:

- Application Protocol Inspection Overview
- Application Protocol Inspection Configuration Quick Start Procedures
- Configuring a Layer 7 FTP Command Inspection Policy
- Configuring a Layer 7 HTTP Deep Inspection Policy
- Configuring a Layer 7 SCCP Inspection Policy
- Configuring a Layer 7 SIP Inspection Policy
- Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy
- Configuring a DNS Parameter Map
- Configuring an HTTP Parameter Map
- Configuring an SCCP Parameter Map
- Configuring a SIP Parameter Map
- Applying a Service Policy
Application Protocol Inspection Overview

Certain applications require special handling of the data portion of a packet as the packets pass through the ACE. Application protocol inspection helps to verify the protocol behavior and identify unwanted or malicious traffic that passes through the ACE. Based on the specifications of the traffic policy, the ACE accepts or rejects the packets to ensure the secure use of applications and services.

This section contains the following topics on application protocol inspection:

- Performing Application Protocol Inspection
- Application Inspection Protocol Overview

Performing Application Protocol Inspection

You can configure the ACE to perform application protocol inspection, sometimes referred to as an application protocol “fixup” for applications that do the following:

- Embed IP addressing information in the data packet including the data payload.
- Open secondary channels on dynamically assigned ports.

You may require the ACE to perform application inspection of Domain Name System (DNS), FTP (File Transfer Protocol), HTTP, Internet Control Message Protocol (ICMP), Internet Locator Service (ILS), Real-Time Streaming Protocol (RTSP), Skinny Client Control Protocol (SCCP), and Session Initiation Protocol (SIP) as a first step before passing the packets to the destination server. For HTTP, the ACE performs deep packet inspection to statefully monitor the HTTP protocol and permit or deny traffic based on user-defined traffic policies. HTTP deep packet inspection focuses mainly on HTTP attributes such as the HTTP header, the URL, and the payload. For FTP, the ACE performs FTP command inspection for FTP sessions, allowing you to restrict specific commands by the ACE.
Application inspection helps you to identify the location of the embedded IP addressing information in the TCP or UDP flow. This inspection allows the ACE to translate embedded IP addresses and to update any checksum or other fields that are affected by the translation.

Translating IP addresses embedded in the payload of protocols is especially important for NAT (explicitly configured by the user) and server load balancing (an implicit NAT).

Application inspection also monitors TCP or UDP sessions to determine the port numbers for secondary channels. Some protocols open secondary TCP or UDP ports to improve performance. The initial session on a well-known port is used to negotiate dynamically assigned port numbers. The application protocol inspection function monitors these sessions, identifies the dynamic port assignments, and permits data exchange on these ports for the duration of the session.

Table 3-1 describes the application inspection protocols supported by the ACE, the default TCP or UDP protocol and port, and whether the protocol is compatible with Network Address Translation (NAT) and Port Address Translation (PAT).

**Table 3-1 Application Inspection Support**

<table>
<thead>
<tr>
<th>Application Protocol</th>
<th>Transport Protocol</th>
<th>Port</th>
<th>NAT/PAT Support</th>
<th>Enabled by Default</th>
<th>Standards</th>
<th>Comments/Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>UDP</td>
<td>Src—Any Dest—53</td>
<td>NAT</td>
<td>No</td>
<td>RFC 1123</td>
<td>Inspects DNS packets destined to port 53. You can specify the maximum length of the DNS packet to be inspected. See the “DNS Inspection” section for more information.</td>
</tr>
<tr>
<td>FTP</td>
<td>TCP</td>
<td>Src—Any Dest—21</td>
<td>Both</td>
<td>No</td>
<td>RFC 959</td>
<td>Inspects FTP packets, translates address and port embedded in the payload, and opens up a secondary channel for data. See the “FTP Inspection” section for more information.</td>
</tr>
</tbody>
</table>
### Table 3-1  Application Inspection Support (continued)

<table>
<thead>
<tr>
<th>Application Protocol</th>
<th>Transport Protocol</th>
<th>Port</th>
<th>NAT/PAT Support</th>
<th>Enabled by Default</th>
<th>Standards</th>
<th>Comments/Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>FTP strict</td>
<td>TCP</td>
<td>Src—Any</td>
<td>Both</td>
<td>No</td>
<td>RFC 959</td>
<td>The <code>inspect ftp strict</code> command allows the ACE to track each FTP command and response sequence and also prevents an FTP client from determining valid usernames that are supported on an FTP server. See the “FTP Inspection” section for more information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dest—21</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTTP</td>
<td>TCP</td>
<td>Src—Any</td>
<td>Both</td>
<td>No</td>
<td>RFC 2616</td>
<td>Inspects HTTP packets. See the “HTTP Deep Packet Inspection” section for more information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dest—80</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICMP</td>
<td>ICMP</td>
<td>Src—N/A</td>
<td>Both</td>
<td>No</td>
<td>—</td>
<td>See the “ICMP Inspection” section for more information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dest—N/A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3-1 Application Inspection Support (continued)

<table>
<thead>
<tr>
<th>Application Protocol</th>
<th>Transport Protocol</th>
<th>Port</th>
<th>NAT/PAT Support</th>
<th>Enabled by Default</th>
<th>Standards(^1)</th>
<th>Comments/Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICMP error</td>
<td>ICMP</td>
<td>Src—N/A, Dest—N/A</td>
<td>NAT</td>
<td>No</td>
<td>—</td>
<td>The <em>error</em> keyword supports NAT of ICMP error messages. When you enable ICMP error inspection, the ACE creates translation sessions for intermediate hops that send ICMP error messages, based on the NAT configuration. The ACE overwrites the packet with the translated IP addresses. See the “ICMP Inspection” section for more information.</td>
</tr>
<tr>
<td>ILS</td>
<td>TCP</td>
<td>Src—Any, Dest—389</td>
<td>NAT</td>
<td>No</td>
<td>RFC 2251 (LDAPv3) Includes support for RFC 1777 (LDAPv2)</td>
<td>Referral requests and responses are not supported. Users in multiple directories are not unified. Single users having multiple identities in multiple directories cannot be recognized by NAT.</td>
</tr>
</tbody>
</table>
You configure rules for application protocol inspection through class maps, policy maps, and service policies. The following items summarize the role of each function in configuring application protocol inspection:

- **Layer 7 class map**—(Optional) Provides the Layer 7 network traffic classification to identify protocol inspection attributes (such as the HTTP header and the URL) and FTP request commands.

- **Layer 7 policy map**—(Optional) Configures the applicable match statements and actions that the ACE executes on the network traffic that matches the classifications defined in the Layer 7 class map.

---

### Table 3-1 Application Inspection Support (continued)

<table>
<thead>
<tr>
<th>Application Protocol</th>
<th>Transport Protocol</th>
<th>Port</th>
<th>NAT/PAT Support</th>
<th>Enabled by Default</th>
<th>Standards</th>
<th>Comments/Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTSP</td>
<td>TCP</td>
<td>Src—Any Dest—554</td>
<td>NAT</td>
<td>No</td>
<td>RFC 2326, RFC 2327, RFC 1889</td>
<td>Inspects RTSP packets and translates the payload according to NAT rules. The ACE opens up the secondary channels for audio and video. Not all the RTSP methods (packet types) specified in the RFC are supported. See the “RTSP Inspection” section for more information.</td>
</tr>
<tr>
<td>SCCP</td>
<td>TCP</td>
<td>Src—Any Dest—2000</td>
<td>NAT</td>
<td>No</td>
<td>—</td>
<td>The ACE does not support PAT with SCCP.</td>
</tr>
<tr>
<td>SIP</td>
<td>TCP and UDP</td>
<td>Src—Any Dest—5060</td>
<td>NAT</td>
<td>No</td>
<td>RFC 2543, RFC 3261, RFC 3265, RFC 3428</td>
<td>The ACE does not support PAT with SIP.</td>
</tr>
</tbody>
</table>

1. The ACE is in compliance with these standards, but it does not enforce compliance on packets being inspected. For example, FTP commands are supposed to be in a particular order, but the ACE does not enforce the order.
• Layer 3 and Layer 4 class map—Classifies the network traffic that passes through the ACE for application inspection and matches the traffic associated with the specified `inspect` commands in a policy map.

• Layer 3 and Layer 4 policy map—Enables DNS, FTP, HTTP, ICMP, ILS, RTSP, SCCP, and SIP protocol inspection and FTP command inspection for a traffic classification that matches the criteria listed in the class map.

• Service policy—Activates the policy map and associates the traffic policy with a VLAN interface or globally with all VLAN interfaces.

Figure 3-1 provides an overview of the process required to configure class maps and policy maps to perform application protocol inspection. The flow chart also shows how the ACE associates the various components of the class map and policy map configuration with each other.
Figure 3-1 Application Protocol Inspection Configuration Flow Diagram

1. Layer 7 HTTP Inspection Class Map
   (config)# class-map type http inspect match-all
   Defines multiple Layer 7 HTTP deep packet inspection match criteria, such as:
   - Content expressions and length
   - Header, header length, header MIME-type
   - Port misuse
   - URL expressions and length

2. Layer 7 HTTP Inspection Policy Map
   (config)# policy-map type inspect http all-match HTTP_INSPECT_L7POLICY
   Associates the Layer 7 HTTP inspection class map and specifies one or more of the following actions:
   - Permit
   - Reset

3. Layer 7 FTP Inspection Class Map
   (config)# class-map type ftp inspect match-any FTP_INSPECT_L7CLASS
   Defines multiple Layer 7 FTP request command inspection match criteria, including: appe, cdup, dele, get, help, mkd, put, rmd, rnfr, rnto, site, stou, and syst

4. Layer 7 FTP Inspection Policy Map
   (config)# policy-map type inspect ftp first-match FTP_INSPECT_L7POLICY
   Associates the Layer 7 FTP inspection class map and specifies one or more of the following actions:
   - Deny
   - Mask-reply

5. Layer 3 and Layer 4 Traffic Class Map
   (config)# class-map match-all | match-any HTTP_INSPECT_L4CLASS
   Defines Layer 3 and Layer 4 traffic match criteria for application protocol inspection:
   - Access list
   - Port

6. Layer 3 and Layer 4 Policy Map
   (config)# policy-map multi-match HTTP_INSPECT_L4POLICY
   Creates a Layer 3 and Layer 4 policy map to perform one or more of the following actions:
   - Associate Layer 3 and Layer 4 traffic class map
   - Associate Layer 7 HTTP deep packet inspection policy map
   - Associate Layer 7 FTP command inspection policy map
   - Perform HTTP inspection
   - Perform DNS inspection
   - Perform FTP inspection
   - Perform ICMP inspection
   - Perform RTSP inspection

7. Global Service Policy/VLAN
   (config)# service-policy input HTTP_INSPECT_L4POLICY
   Service policy applies policy map to all VLAN interfaces in the context

   Specific Service Policy/VLAN
   (config)# interface vlan 50
   (config-if)# service-policy input HTTP_INSPECT_L4POLICY
   Service policy applies policy map to a specific VLAN interface

Policy map applied globally to all VLAN interfaces or to a specific VLAN interface
Application Inspection Protocol Overview

This section provides an overview of the application inspection protocols supported by the ACE and contains the following topics:

- DNS Inspection
- FTP Inspection
- HTTP Deep Packet Inspection
- ICMP Inspection
- ILS Inspection
- RTSP Inspection
- SCCP Inspection
- SIP Inspection

DNS Inspection

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 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 NAT configuration. Only forward lookups are translated using NAT; the ACE does not handle pointer (PTR) records.

Note: The DNS rewrite function is not applicable for PAT because multiple PAT rules apply to each A-record. Using multiple PAT rules makes it difficult for the ACE to properly choose the correct PAT rule.

- Performs a maximum DNS packet length check to verify that the maximum length of a DNS reply is no greater than the value specified in the inspect dns command.
Application Protocol Inspection Overview

Note

If you enter the inspect dns command without specifying the maximum-length optional keyword, the ACE does not check the DNS packet size.

• Performs a number of security checks as follows:
  – 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

A single connection is created for multiple DNS sessions if the DNS sessions are between the same two hosts and the sessions have the same 5-tuple (source and destination IP address, source and destination port, and protocol). DNS identification is tracked by app_id, and the idle timer for each app_id runs independently.

Because the app_id expires independently, a legitimate DNS response can only pass through the ACE within a limited period of time and there is no resource build-up. However, if you enter the show conn command, you will see the idle timer of a DNS connection being reset by a new DNS session. This reset action is due to the shared DNS connection.

FTP Inspection

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.

The ACE performs the FTP command inspection process as follows:

• Prepares a dynamic secondary data connection. The channels are allocated in response to a file upload, a file download, or a directory listing event and must be prenegotiated. The port is negotiated through the PORT or PASV commands.

• Tracks the FTP command-response sequence. The ACE performs the following FTP command checks listed below.
- Truncated command—Checks the number of commas in the PORT and PASV reply command against a fixed value of five. If the value is not five, the ACE assumes that the PORT command is truncated, issues a warning message, and closes the TCP connection.

- Incorrect command—Checks the FTP command to verify if it ends with <CR><LF> characters, as required by RFC 959. If the FTP command does not end with those characters, the ACE closes the connection.

- Invalid port negotiation—Checks the negotiated dynamic port value to verify that it is greater than 1024 (port numbers from 2 to 1024 are reserved for well-known connections). If the negotiated port falls in this range, the ACE closes the TCP connection.

- Command pipelining—Checks the number of characters present after the port numbers in the PORT and PASV reply command against a constant value of 8. If the number of characters is greater than 8, the ACE closes the TCP connection.

In addition to these FTP command checks, if you specify the strict keyword with the inspect ftp command in a Layer 3 and Layer 4 policy map, the ACE tracks each FTP command and response sequence for the anomalous activity outlined below. The strict keyword can be used with a Layer 7 FTP policy map (nested within the Layer 3 and Layer 4 policy map) to deny certain FTP commands or to mask the server reply for the SYST command.

Note Using the strict keyword may affect FTP clients that do not comply with the RFC standards.

- Size of RETR and STOR commands—Checks the size of the RETR and STOR commands against a fixed constant of 256. If the size is greater, the ACE logs an error message and closes the connection.

- Command spoofing—Verifies that the PORT command is always sent from the client. If a PORT command is sent from the server, the ACE denies the TCP connection.

- Reply spoofing—Verifies that the PASV reply command (227) is always sent from the server. If a PASV reply command is sent from the client, the ACE denies the TCP connection. This denial prevents a security hole when the user executes “227 xyyyy a1, a2, a3, a4, p1, p2.”
• Translates embedded IP addresses with NAT. FTP command inspection translates the IP address within the application payload. See RFC 959 for more details.

**HTTP Deep Packet Inspection**

When you enable HTTP inspection using the `inspect http` command as an action in a Layer 4 policy map, the ACE performs a stateful deep packet inspection of the HTTP protocol. Deep packet inspection is a special case of application inspection where the ACE examines the application payload of a packet or a traffic stream and makes decisions based on the content of the data. 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 when you configure HTTP inspection at Layer 7.

You define policies to permit or deny the traffic or to send a TCP reset message to the client or server to close the connection. You can enable HTTP deep packet inspection with or without a Layer 7 policy and with or without URL logging. To enable HTTP inspection, you must configure a Layer 4 HTTP inspection policy at a minimum.

**Layer 4 HTTP Inspection Policy Map**

If you configure HTTP inspection without a Layer 7 policy map or URL logging (Layer 4 inspection policy only), the ACE performs the following basic inspection operations and checks. For details about configuring Layer 4 HTTP inspection, see the “Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy” section.

• URL parsing and deobfuscation—ACE parses and normalizes the URL to detect any encoded attacks. It also performs URI length checks. The HTTP protocol specifies that arbitrary binary characters can be passed within a URI by using %xx notation. Such characters can be used to bypass checks in the URI by encoding them in hexadecimal notation. To rectify this potential issue, the ACE replaces the hexadecimal encodings with ASCII equivalents before they enter the regex parser. For example, the following characters were detected as being encoded as part of a URL: Null %00 New Line %0a Carriage Return %0d Period (.) %2e Forward Slash (/) %2f Back Slash (\) %5c. The ACE also detects double encoding attacks. For example, the
forward slash (/) character could be encoded as %252f. Any violation of the ACE inspection rules or an error causes a static parse error, which results in a connection reset.

- Header parsing and parser validation—ACE parses HTTP headers and deobfuscates them and detects and fixes header encoding. The ACE parses standard headers for predefined header matching (for example, content-type, content-length, and so on). It also looks for end-of-line characters like CRLF, and any violation causes a static parse error, which results in a connection reset.

- Strict HTTP inspection—When you enable HTTP inspection, strict HTTP parsing is enabled by default. Strict HTTP inspection performs RFC checking and the default action of RESET can be changed by configuring a Layer 7 policy map. For details about configuring a Layer 7 policy map for HTTP inspection, see the “Configuring a Layer 7 HTTP Deep Inspection Policy” section. With strict inspection, the ACE statefully monitors the HTTP protocol as defined in RFC 2616 and performs compliance checking. The ACE also ensures the correctness of correspondence between the HTTP request and the response. Any deviation from strict inspection rules causes a connection reset.

- Method validation, permit only RFC and extension methods—ACE verifies the request method and permits only well-known RFC and Extension methods that are defined in RFC 2616 and RFC 4918. Any request with other methods is denied. The ACE supports the following RFC methods:
  - CONNECT
  - DELETE
  - GET
  - HEAD
  - OPTIONS
  - POST
  - PUT
  - TRACE
The following extension methods are supported along with Web-based Distributed Authoring and Versioning (WebDAV) methods:

<table>
<thead>
<tr>
<th>Extension Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCOPY</td>
</tr>
<tr>
<td>BDELETE</td>
</tr>
<tr>
<td>BMOVE</td>
</tr>
<tr>
<td>BPROPFIND</td>
</tr>
<tr>
<td>BPROPPATCH</td>
</tr>
<tr>
<td>COPY</td>
</tr>
<tr>
<td>EDIT</td>
</tr>
<tr>
<td>GETATTRIBUTE</td>
</tr>
<tr>
<td>GETATTRIBUTENAMES</td>
</tr>
<tr>
<td>GETPROPERTIES</td>
</tr>
<tr>
<td>INDEX</td>
</tr>
<tr>
<td>LOCK</td>
</tr>
<tr>
<td>MKCOL</td>
</tr>
<tr>
<td>MKDIR</td>
</tr>
<tr>
<td>MOVE</td>
</tr>
<tr>
<td>NOTIFY</td>
</tr>
<tr>
<td>POLL</td>
</tr>
<tr>
<td>PROPATCH</td>
</tr>
<tr>
<td>PROPFIND</td>
</tr>
<tr>
<td>PROPPATCH</td>
</tr>
<tr>
<td>REVLABEL</td>
</tr>
<tr>
<td>REVLOG</td>
</tr>
<tr>
<td>REVNUM</td>
</tr>
<tr>
<td>SAVE</td>
</tr>
<tr>
<td>SEARCH</td>
</tr>
<tr>
<td>SETATTRIBUTE</td>
</tr>
<tr>
<td>STARTREV</td>
</tr>
<tr>
<td>STOPREV</td>
</tr>
<tr>
<td>SUBSCRIBE</td>
</tr>
<tr>
<td>UNEDIT</td>
</tr>
<tr>
<td>UNLOCK</td>
</tr>
<tr>
<td>UNSUBSCRIBE</td>
</tr>
<tr>
<td>X_MS_ENUMATTS</td>
</tr>
</tbody>
</table>

**Layer 7 HTTP Inspection Policy Map**

You can enable the following optional checks and actions by configuring a Layer 7 HTTP inspection policy:

- Regular expression matching on the name in an HTTP header, URL name, or content expressions in an HTTP entity-body
- Content, URL, and HTTP header length checks
- URL query parameter and secondary cookie filtering
- MIME-type message inspection
- Content and MIME type verification and filtering
- Transfer-encoding and method filtering
- Port 80 misuse detection and filtering
URL Logging

If you enable URL logging in the Layer 4 HTTP inspection policy, the ACE generates a syslog for every URL request that is sent in the specified traffic class, including the source or destination IP address and the URL that is accessed.

ICMP Inspection

Internet Control Message Protocol (ICMP) inspection 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 do 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.

For stateful ICMP, state information, as maintained for TCP or UDP flows, is maintained for ICMP instead of performing only the ACL and NAT functions. The maintenance of ICMP state information is required to resolve the following problems:

- ICMP reply messages without request messages
- Unsolicited ICMP error messages
- Unknown ICMP types

ICMP error messages are generated by intermediate nodes situated on the network path to a destination whenever a packet sent to that destination cannot be forwarded. ICMP error messages may also be generated by endpoint nodes, as in the case of port unreachable errors. ICMP error messages carry the original packet for which the error is generated in the data part of the message. The error message also contains the addresses of the intermediate node or endpoint node in the outer header and the destination node in the inner header.

ICMP error fixup handles address translation of node address and destination address to global addresses using the NAT configuration. ICMP error fixup is user configurable. If you do not enable this feature, intermediate node or endpoint node addresses are translated in the same way as the destination address of the embedded packet. As a result, error messages appear as if they are originating from the destination and the node addresses or the route to the destination are not included.
ICMP inspection performs the following tasks for ICMP request or reply messages:

- Creates a bidirectional session or connection record. The lookup key in the forward direction is the source IP address, destination IP address, protocol, ICMP type, ICMP identifier, and VLAN.
- Verifies that the connection record contains a sequence number window that specifies the list of sequence numbers of outstanding requests for which replies are pending.
- Verifies that the connection record has a timeout, so that the inactive connection record can be reused for other flows and can protect the inside network against fraudulent ICMP reply packets.
- Allows reply packets only if a valid connection record exists and prevents the reply packets from passing through an ACL again if the connection record (or the state information) exists.
- Creates a connection record for the transit ICMP request or reply packets and also for those packets addressed to or from the ACE.

ICMP error message inspection performs the following tasks:

- Extracts the embedded IP header in the ICMP error message and checks for the presence of a connection record that corresponds to the embedded packet for which the error message has been generated.
- Performs an ACL of the ICMP error message regardless of the existence of a session for the embedded packet. The ICMP error message is itself stateless and requires access control.
- Allocates NAT translation entries (xlate) for intermediate nodes or endpoint nodes to perform NAT of a local IP address to a global IP address in any ICMP error message.
- Updates the checksum in the outer and inner headers.

**ILS Inspection**

Internet Locator Service (ILS) is used by Microsoft NetMeeting to help users find other users. ILS interfaces with the Lightweight Directory Access Protocol (LDAP) to provide directory services. The ACE ILS inspection feature provides NAT support for NetMeeting, Site Server, and Active Directory products that use
LDAP to exchange directory information with an ILS server. The ACE does not support PAT for ILS because the LDAP database stores only IP addresses and not ports.

ILS/LDAP follows the client/server model and uses a single TCP connection for each session. Depending on the client actions, several sessions may be required. During the connection setup, the client sends a BIND protocol data unit (PDU) to the server. After the client receives the BIND RESPONSE from the server, other messages (for example, ADD, DEL, SEARCH, or MODIFY) can be exchanged to perform operations on the ILS Directory.

The ADD REQUEST and SEARCH REQUEST PDUs may contain addresses of NetMeeting peers. NetMeeting version 2.x and 3.x provide ILS support.

Because ILS traffic occurs only on the secondary UDP channel, the ACE disconnects the TCP connection after the TCP inactivity interval has elapsed. By default, the TCP inactivity is 60 minutes, but you can adjust it using a connection parameter map. For information about configuring a connection parameter map, see Chapter 4, Configuring TCP/IP Normalization and IP Reassembly Parameters.

The ACE performs the following ILS inspection operations:

- Decodes the LDAP REQUEST/RESPONSE PDUs using the Basic Encoding Rules (BER) decoder functions
- Parses the LDAP packet
- Extracts IP addresses
- Translates IP addresses as necessary
- Encodes the PDU with translated addresses using BER encode functions
- Copies the newly encoded PDU back to the TCP packet
- Performs an incremental TCP checksum and sequence number adjustment

The following restrictions apply to the ACE ILS inspection feature:

- Referral requests and responses are not supported.
- Users in multiple directories are not unified.
- Single users having multiple identities in multiple directories cannot be recognized by NAT.
RTSP Inspection

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.

The TCP control channel negotiates the data channels used to transmit audio and video traffic, depending on the transport mode that is configured on the client. The supported data transport modes are rtp/avp, rtp/avp/udp, x-real-rdt, x-real-rdt/udp, and x-pn-tng/udp. The data transport types rtp/avp/tcp and x-real-rdt/tcp use the control channel to stream data. RTSP inspection is not required in this case to open a secure port (pinhole) for the data channel.

The ACE parses SETUP response messages with a status code of 200. Because RFC 2326 does not require that the client and server ports are contained in the SETUP response message, the ACE must track the state and remember the client ports in the SETUP message. QuickTime places the client ports in the SETUP message; the server responds with only the server ports.

During RTSP inspection, the ACE does not do the following:

- Inspect RTSP messages that pass through UDP ports.
- Support RealNetworks multicast mode (x-real-rdt/mcast).
- Support the ability to recognize HTTP cloaking where RTSP messages are hidden in HTTP messages.
- Perform NAT on RTSP messages because the embedded IP addresses are contained in the Session Description Protocol (SDP) files as part of HTTP or RTSP messages.

The following additional restrictions apply to RTSP inspection as performed by the ACE:

- With Cisco IP/TV, the number of translations that the ACE performs on the SDP part of the message is proportional to the number of program listings in the Content Manager (each program listing can have at least six embedded IP addresses).
- When using RealPlayer, you must properly configure the transport mode. For the ACE, add an ACL classification from the server to the client. For RealPlayer, change the transport mode by clicking **Tools>Preferences>Connection>Network Transport>RTSP Settings.**
– If you use TCP mode on the RealPlayer, check the **Attempt to use TCP for all content** check box. It is not necessary to configure RTSP application inspection on the ACE.

– If you use UDP mode on the RealPlayer, check the **Attempt to use UDP for all content** check box. Configure RTSP application inspection on the ACE.

### SCCP Inspection

Skinny Client Control Protocol (SCCP) is used in VoIP networks, for example, with Cisco IP phones and Cisco CallManager. The ACE supports all versions of the SCCP protocol through version 3.3.2.

SCCP inspection provides the following operations:

- Supports NAT for embedded IP addresses and ports.
- Dynamically opens secure ports.
- Drops messages with an SCCPPrefix length that is less than the message ID length (configurable).
- Supports video.
- Validates message ID length (configurable maximum).
- Ensures that only registered clients can make calls. This feature is configurable and disabled by default.
- Allows you to configure timeouts.

### SIP Inspection

Session Initiation Protocol (SIP) is used for call handling sessions, especially two-party conferences. SIP works with SDP for call signaling.

SIP inspection provides the following operations:

- Translates the SIP text-based messages, recalculates the content length for the SDP portion of the message, and recalculates the packet length and checksum.
- Dynamically opens media connections for ports specified in the SDP portion of the SIP message as addresses and ports on which the endpoint should listen.
- Opens RTP and RTCP connections between the two endpoints using media addresses and ports that are maintained in a SIP inspection database with CALL_ID, FROM, and TO indices from the SIP header. These indices identify the call, the source, and the destination.

- Performs RFC 3261 compliance checks, including checking the Request Message to ensure it is one of the predefined methods: OPTIONS, INVITE, REGISTER, ACK, CANCEL, BYE and validates their syntax.

- Checks whether a SIP message is compliant with the following RFC extensions:
  - RFC 2976 (INFO)
  - RFC 3262 (PRACK)
  - RFC 3265 (SUBSCRIBE/NOTIFY)
  - RFC 3311 (UPDATE)
  - RFC 3515 and RFC 3892 (REFER)
  - RFC 3428 (MESSAGE)

- Enforces the mandatory header fields (From, To, Call-Id, CSeq, Via, Max-Forwards) presence and validity.

- Enforces forbidden header fields.

- Checks URI in Header fields against a permit or deny list of callers or callees. If the user is not entitled to talk to any host on the protected network, the SIP ACE module will generate a SIP message (Response 603 Decline).

- Checks the Via field to deny messages from specific SIP proxy servers.

- Checks the validity of each header parameter in the context of each message following the syntax rules specified in RFC 3261.

- Removes the optional User-Agent and Server header fields to hide the endpoint software version.

- Checks the Max-Forwards header field. If the Max-Forwards value reaches 0 before the request reaches its destination, the ACE rejects the request with a 483 (Too Many Hops) error response.

- Validates SIP URIs and URIs present in the SIP header fields.
- Handles unknown SIP methods. Because SIP is an evolving protocol, which includes many extensions, some of the new methods may not be recognized by the ACE (only the methods defined by RFC 3261 and the extensions listed above are supported). You can configure how the ACE handles “unknown” SIP methods.

- Permits or denies third-party registrations or deregistrations and specifies which users are allowed to perform these functions. If this policy is enabled, REGISTER messages, with mismatched To and From headers and with From values that do not match any of the privileged user IDs, are dropped.

- Protects against buffer overflows as follows:
  - Enforces the Content-Length and the Content-Type (user configurable) values:
  - Allows you to configure the maximum size of a SIP message body. When a request or response SIP message passes through the ACE module, the message is checked to ensure that it meets the size constraints. If it does not, the action configured for this policy by the user will be executed.
  - Cross checks the Content-Length header field value with the actual message size.
  - Allows you to select whether a subset of Content-types are permitted through the ACE module. You can specify the Content-type string in the form of a regular expression, for example, Application/SDP, text/html. The default behavior is to allow all types.
  - Enforces SIP or SIPS URI length (user configurable).

- Enables or disables Instant Messenger (IM):
  - Allows you to disable IM over SIP, which causes the ACE to drop all messages belonging to IM as specified by the SIP for Instant Messaging and Presence Leveraging Extensions (SIMPLE) RFC. An appropriate warning message is displayed to call out the exact methods that this feature drops.
  - You can specify a list of users (in the form of a regex) that are not allowed to use IM through the ACE module.

- Allows you to configure which SIP methods that the ACE supports. You can also specify if additional SIP methods (that are not part of the RFCs or RFC extensions that the ACE is compliant with) should be denied. The ACE maintains the list of invalid methods as a regex table.
Enables you to hide or remove risky header fields (for example, Alert-Info and Call-Info) that, if provided by a malicious caller, may cause the callee to display inappropriate, offensive, dangerous, or illegal content.

Allows you to enable IP address privacy. If both the caller and the callee are on the inside network and on the same subnet, and the proxy is on the outside network, there is a possibility that the two parties may try to contact each other by bypassing the proxy. If enabled, this feature prevents such direct contact because the embedded addresses in the message from the proxy to the callee are not fixed. Therefore, the callee cannot learn the real IP address of the caller.

Application Protocol Inspection Configuration Quick Start Procedures

Table 3-2, Table 3-3, and Table 3-4 provide a quick overview of the steps required to configure application protocol inspection on the ACE:

- See Table 3-2 for a quick overview on configuring Layer 7 FTP request command inspection.
- See Table 3-3 for a quick overview on configuring Layer 7 HTTP deep inspection.
- See Table 3-4 for a quick overview on configuring Layer 3 and Layer 4 DNS, FTP, HTTP, ICMP, and RTSP application protocol inspection.
Table 3-2  Layer 7 FTP Request Command Inspection Quick Start

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. If you are operating in multiple context mode, observe the CLI prompt to verify that you are operating in the desired context. Change to the correct context if necessary.</td>
</tr>
</tbody>
</table>
| ```
host1/Admin# changeto C1
host1/C1#
``` |
| For details on creating contexts, see the *Cisco Application Control Engine Module Virtualization Configuration Guide*. |
| 2. Enter configuration mode. |
| ```
host1/Admin# config
Enter configuration commands, one per line. End with CNTL/Z
host1/Admin(config)#
``` |
| 3. Create a Layer 7 class map that is used for the inspection of FTP request commands. If you do not specify **match-all** or **match-any**, traffic must match all the match criteria to be classified as part of the traffic class. |
| The CLI displays the class map FTP command inspection configuration mode. |
| ```
host1/Admin(config)# class-map type ftp inspect match-any
FTP_INSPECT_L7CLASS
host1/Admin(config-cmap-ftp-insp)#
``` |
| 4. Configure the Layer 7 class map to define FTP request command inspection decisions through the ACE. The **match request** command identifies the FTP commands that you want filtered by the ACE. |
| ```
host1/Admin(config-cmap-ftp-insp)# match request-method mkdir
host1/Admin(config-cmap-ftp-insp)# exit
host1/Admin(config)#
``` |
| 5. Create and configure a Layer 7 policy map that enables FTP command inspection. Specify the actions that you want to apply to the Layer 7 user-defined class map and, if appropriate, to the default class map. |
| ```
host1/Admin(config)# policy-map type inspect ftp first-match
FTP_INSPECT_L7POLICY
host1/Admin(config-pmap-ftp-ins)# class FTP_INSPECT_L7CLASS
host1/Admin(config-pmap-ftp-ins-c)# deny
host1/Admin(config-pmap-ftp-ins-c)# exit
host1/Admin(config)#
```
Create a Layer 3 and Layer 4 class map to classify network traffic that passes through the ACE for FTP command inspection. If you do not specify match-all or match-any, traffic must match all the match criteria to be classified as part of the traffic class.

The CLI displays the class map configuration mode.

```
host1/Admin(config)# class-map match-all FTP_INSPECT_L4CLASS
host1/Admin(config-cmap)#
```

Include one or more of the match commands as part of the Layer 3 and Layer 4 class map.

```
host1/Admin(config-cmap)# description FTP command inspection of incoming traffic
host1/Admin(config-cmap)# match port tcp eq 21
host1/Admin(config-cmap)# exit
host1/Admin(config)#
```

Create a Layer 3 and Layer 4 policy map and associate the Layer 7 FTP command inspection policy map to activate the operation. Specify the actions that you want to apply to the Layer 3 and Layer 4 user-defined class map and, if appropriate, to the default class map.

```
host1/Admin(config)# policy-map multi-match FTP_INSPECT_L4POLICY
host1/Admin(config-pmap)# class FTP_INSPECT_L4CLASS
host1/Admin(config-pmap-c)# inspect ftp strict policy FTP_INSPECT_L7POLICY
host1/Admin(config-pmap-c)# exit
host1/Admin(config)#
```

Attach the Layer 3 and Layer 4 traffic policy to a single VLAN interface or globally to all VLAN interfaces, and specify the direction in which the policy should be applied. For example, to specify a VLAN interface and apply multiple service policies to the VLAN, enter:

```
host1/Admin(config)# interface vlan 50
host1/Admin(config-if)# ip address 172.16.1.100 255.255.255.0
host1/Admin(config-if)# service-policy input FTP_INSPECT_L4POLICY
```

(Optional) Save your configuration changes to flash memory.

```
host1/Admin(config)# exit
host1/Admin# copy running-config startup-config
```
Chapter 3  Configuring Application Protocol Inspection

Application Protocol Inspection Configuration Quick Start Procedures

Table 3-3  Layer 7 HTTP Protocol Deep Inspection Quick Start

Task and Command Example

1. If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, log directly in to, or change to, the correct context.

   host1/Admin# changeto C1
   host1/C1#

   The rest of the examples in this table use the Admin context, unless otherwise specified. For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.

2. Enter configuration mode.

   host1/Admin# config
   Enter configuration commands, one per line. End with CNTL/Z
   host1/Admin(config)#

3. Create a Layer 7 class map that is used for the deep packet inspection of HTTP traffic. If you do not specify match-all or match-any, traffic must match all the match criteria to be classified as part of the traffic class.

   The CLI displays the class map HTTP application protocol inspection configuration mode.

   host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
   host1/Admin(config-cmap-http-insp)#

   Include one or more of the match commands listed in Steps 4 through 13 as part of the Layer 7 HTTP deep packet inspection class map.

4. (Optional) Configure the class map to define HTTP application inspection decisions based on content expressions contained within the HTTP content.

   host1/Admin(config-cmap-http-insp)# match content .*newp2psig

5. (Optional) Configure the class map to define application inspection decisions in the HTTP content up to the configured maximum content parse length.

   host1/Admin(config-cmap-http-insp)# match content length eq 1000
Table 3-3  Layer 7 HTTP Protocol Deep Inspection Quick Start (continued)

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>6. (Optional) Configure the class map to define application inspection decisions based on the name and value in an HTTP header.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match header Host header-value .mycompanyexample.com</td>
</tr>
<tr>
<td>7. (Optional) Limit the HTTP traffic allowed through the ACE based on the length of the entity-body in the HTTP message.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match header length request eq 256</td>
</tr>
<tr>
<td>8. (Optional) Specify a subset of the Multipurpose Internet Mail Extension (MIME)-type messages to be permitted or denied by the ACE.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match header mime-type audio\midi</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match header mime-type audio\mpeg</td>
</tr>
<tr>
<td>9. (Optional) Configure the class map to define application inspection compliance decisions that restrict certain HTTP traffic from passing through the ACE.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match port-misuse p2p</td>
</tr>
<tr>
<td>10. (Optional) Configure the class map to define application inspection compliance decisions based on the request methods defined in RFC 2616 and by HTTP extension methods.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match request-method rfc connect</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match request-method rfc get</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match request-method rfc head</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match request-method ext index</td>
</tr>
<tr>
<td>11. (Optional) Configure the class map to define application inspection decisions that limit the HTTP transfer-encoding types that can pass through the ACE.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match transfer-encoding chunked</td>
</tr>
</tbody>
</table>
### Table 3-3  Layer 7 HTTP Protocol Deep Inspection Quick Start (continued)

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>12.</strong> (Optional) Configure the class map to define application inspection decisions based on the URL name.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match url .*.gif</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match url .*.html</td>
</tr>
<tr>
<td><strong>13.</strong> (Optional) Limit the HTTP traffic allowed through the ACE by specifying the maximum length of a URL in a request message that can be received by the ACE.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap-http-insp)# match url length eq 10000</td>
</tr>
<tr>
<td><strong>14.</strong> Create and configure a Layer 7 policy map that enables the deep packet inspection of the HTTP protocol. Specify the actions that you want to apply to the Layer 7 user-defined class map and, if appropriate, to the default class map.</td>
</tr>
<tr>
<td>host1/Admin(config)# policy-map type inspect http all-match HTTP_INSPECT_L7POLICY</td>
</tr>
<tr>
<td>host1/Admin(config-pmap-ins-http)# class HTTP_INSPECT_L7CLASS</td>
</tr>
<tr>
<td>host1/Admin(config-pmap-ins-http-c)# permit</td>
</tr>
<tr>
<td>host1/Admin(config-pmap-ins-http-c)# exit</td>
</tr>
<tr>
<td>host1/Admin(config-pmap-ins-http)# exit</td>
</tr>
<tr>
<td>host1/Admin(config)#</td>
</tr>
<tr>
<td><strong>15.</strong> Create a Layer 3 and Layer 4 class map to classify network traffic that passes through the ACE for HTTP deep packet inspection. If you do not specify <strong>match-all</strong> or <strong>match-any</strong>, traffic must match all the match criteria to be classified as part of the traffic class.</td>
</tr>
<tr>
<td>The CLI displays the class map configuration mode.</td>
</tr>
<tr>
<td>host1/Admin(config)# class-map match-all HTTP_INSPECT_L4CLASS</td>
</tr>
<tr>
<td>host1/Admin(config-cmap)#</td>
</tr>
<tr>
<td>Include one or more of the <strong>match</strong> commands as part of the Layer 3 and Layer 4 class map.</td>
</tr>
<tr>
<td>host1/Admin(config-cmap)# description HTTP protocol deep inspection of incoming traffic</td>
</tr>
<tr>
<td>host1/Admin(config-cmap)# match port tcp eq 80</td>
</tr>
<tr>
<td>host1/Admin(config-cmap)# exit</td>
</tr>
<tr>
<td>host1/Admin(config)#</td>
</tr>
</tbody>
</table>
16. Create a Layer 3 and Layer 4 policy map and associate the Layer 7 HTTP deep packet inspection policy map to activate the operation. Specify the actions that you want to apply to the Layer 3 and Layer 4 user-defined class map and, if appropriate, to the default class map.

   host1/Admin(config)# policy-map multi-match HTTP_INSPECT_L4POLICY
   host1/Admin(config-pmap)# class HTTP_INSPECT_L4CLASS
   host1/Admin(config-pmap-c)# inspect http policy HTTP_INSPECT_L7POLICY
   host1/Admin(config-pmap-c)# exit
   host1/Admin(config-pmap)# exit
   host1/Admin(config)#

17. Attach the Layer 3 and Layer 4 traffic policy to a single VLAN interface or globally to all VLAN interfaces. For example, to specify a VLAN interface and apply multiple service policies to the VLAN, enter:

   host1/Admin(config)# interface vlan50
   host1/Admin(config-if)# ip address 172.16.1.100 255.255.255.0
   host1/Admin(config-if)# service-policy input HTTP_INSPECT_L4POLICY

18. (Optional) Save your configuration changes to flash memory.

   host1/Admin(config)# exit
   host1/Admin# copy running-config startup-config
Table 3-4  Layer 3 and Layer 4 Application Protocol Inspection Quick Start

Task and Command Example

1. If you are operating in multiple context mode, observe the CLI prompt to verify that you are operating in the desired context. Change to the correct context if necessary.

   host1/Admin# change to C1
   host1/C1#

   For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.

2. Enter configuration mode.

   host1/Admin# config
   Enter configuration commands, one per line. End with CNTL/Z
   host1/Admin(config)#

3. Create a Layer 3 and Layer 4 class map to classify network traffic that passes through the ACE for DNS, FTP, HTTP, ICMP, ILS, RTSP, SCCP, or SIP application protocol inspection. If you do not specify match-all or match-any, traffic must match all the match criteria to be classified as part of the traffic class.

   The CLI displays the class map configuration mode.

   host1/Admin(config)# class-map match-all DNS_INSPECT_L4CLASS
   host1/Admin(config-cmap)#

   Include one or more of the match commands as part of the Layer 3 and Layer 4 class map.

   host1/Admin(config-cmap)# description DNS application protocol inspection of incoming traffic
   host1/Admin(config-cmap)# match port udp eq domain
   host1/Admin(config-cmap)# exit
Configuring a Layer 7 FTP Command Inspection Policy

This section describes how to create a Layer 7 class map and policy map that allows the ACE to perform FTP command inspection, which is a security feature that prevents web browsers from sending embedded commands to the ACE in FTP requests. The ACE must acknowledge each FTP command before allowing a new command. FTP inspection allows traffic by default and restricts traffic that fails the security checks. Command filtering allows you to restrict specific commands through the ACE. When the ACE denies a command, it closes the connection.

4. Create a Layer 3 and Layer 4 policy map and include the appropriate inspect command (inspect dns, inspect ftp, inspect http, inspect icmp, inspect ils, inspect rtsp, inspect sip, or inspect skinny for SCCP). Specify the actions that you want to apply to the Layer 3 and Layer 4 user-defined class map and, if appropriate, to the default class map.

For example, to specify the inspect dns command as an action for a DNS application protocol inspection policy map, enter:

```
host1/Admin(config)# policy-map multi-match DNS_INSPECT_L4POLICY
host1/Admin(config-pmap)# class DNS_INSPECT_L4CLASS
host1/Admin(config-pmap-c)# inspect dns maximum-length 1000
host1/Admin(config-pmap-c)# exit
host1/Admin(config-pmap)# exit
host1/Admin(config)#
```

5. Attach the Layer 3 and Layer 4 traffic policy to a single VLAN interface or globally on all VLAN interfaces. For example, to specify a VLAN interface and apply multiple service policies to the VLAN, enter:

```
host1/Admin(config)# interface vlan50
host1/Admin(config-if)# mtu 1500
host1/Admin(config-if)# ip address 192.168.1.100 255.255.0.0
host1/Admin(config-if)# service-policy input DNS_INSPECT_L4POLICY
```

6. (Optional) Save your configuration changes to flash memory.

```
host1/Admin(config)# exit
host1/Admin# copy running-config startup-config
```
You can associate a maximum of 1024 instances of the same type of regular expression (regex) with a Layer 4 policy map. This limit applies to all Layer 7 policy-map types, including generic, HTTP, RADIUS, RDP, RTSP, and SIP. You configure regexes in the following:

- Match statements in Layer 7 class maps
- Inline match statements in Layer 7 policy maps
- Layer 7 hash predictors for server farms
- Layer 7 sticky expressions in sticky groups
- Header insertion and rewrite (including SSL URL rewrite) expressions in Layer 7 action lists

This section contains the following topics:

- Configuring an FTP Inspection Class Map
- Configuring a Layer 7 FTP Command Inspection Policy Map

### Configuring an FTP Inspection Class Map

This section contains the following topics:

- Creating an FTP Inspection Class Map
- Adding a Layer 7 FTP Inspection Class Map Description
- Defining FTP Match Request Methods

### Creating an FTP Inspection Class Map

You can define a class map to be used for the inspection of FTP request commands by using the `class-map type ftp inspect` command in configuration mode.

The syntax of this command is as follows:

```
class-map type ftp inspect match-any map_name
```

The keywords and arguments are as follows:
• **match-any**—Determines how the ACE inspects FTP request commands when multiple match criteria exist in a class map. Only one of the match criteria listed in the class map is satisfied to match the FTP command inspection class in the class map.

• **map_name**—Name assigned to the class map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. The class name is used for both the class map and to configure policy for the class in the policy map.

The CLI displays the class map FTP command inspection configuration mode. To classify the FTP request commands for inspection by the ACE, include one or more of the match request-method commands to configure the match criteria for the Layer 7 class map. See the “Defining FTP Match Request Methods” section.

For example, to specify FTP_INSPECT_L7CLASS as the name of a class map and identify that at least one FTP inspection command in the class map must be satisfied for the ACE to indicate a match, enter:

```
host1/Admin(config)# class-map type ftp inspect match-any FTP_INSPECT_L7CLASS
host1/Admin(config-cmap-ftp-insp)# match request-method cdup
host1/Admin(config-cmap-ftp-insp)# match request-method mkdir
host1/Admin(config-cmap-ftp-insp)# match request-method get
host1/Admin(config-cmap-ftp-insp)# match request-method put
```

To remove the FTP request inspection class map from the ACE, enter:

```
host1/Admin(config)#no class-map type ftp inspect match-any FTP_INSPECT_L7CLASS
```

### Adding a Layer 7 FTP Inspection Class Map Description

You can use the **description** command to provide a brief summary of the Layer 7 FTP inspection class map.

You must access the class map configuration mode to specify the **description** command.

The syntax of this command is as follows:

```
description text
```

Use the **text** argument to enter an unquoted text string with a maximum of 240 alphanumeric characters.
To add a description that the class map is to perform FTP command inspection, enter:

```plaintext
host1/Admin(config-cmap-ftp-insp)# description FTP command inspection of incoming traffic
```

To remove the description from the class map, enter:

```plaintext
host1/Admin(config-cmap-ftp-insp)# no description FTP command inspection of incoming traffic
```

### Defining FTP Match Request Methods

You can use the `match request-method` command to configure the class map to define FTP command inspection decisions by the ACE. The `match` command identifies the FTP commands that you want filtered by the ACE.

You must access the class map configuration mode to specify the `match request-method` command.

The syntax of this command is as follows:

```
match request-method ftp_commands
```

The `ftp_commands` argument is the FTP command in the class map to be subjected to FTP inspection by the ACE. The possible `ftp_commands` are `appe`, `cd`, `cdup`, `dele`, `get`, `help`, `mkd`, `put`, `rmd`, `rnfr`, `rnto`, `site`, `stou`, and `syst`.

You can specify multiple `match request-methods` commands within a class map.

For example, to specify FTP_INSPECT_L7CLASS as the name of a class map and identify that at least one FTP inspection command in the class map must be satisfied for the ACE to indicate a match, enter:

```plaintext
host1/Admin(config)# class-map type ftp inspect match-any FTP_INSPECT_L7CLASS
host1/Admin(config-cmap-ftp-insp)# match request-method cdup
host1/Admin(config-cmap-ftp-insp)# match request-method mkdir
host1/Admin(config-cmap-ftp-insp)# match request-method get
host1/Admin(config-cmap-ftp-insp)# match request-method stou
host1/Admin(config-cmap-ftp-insp)# match request-method put
```

Use the `no` form of the command to clear the FTP inspection request method from the class map:

```plaintext
host1/Admin(config-cmap-ftp-insp)# no match request-method cdup
```
Configuring a Layer 7 FTP Command Inspection Policy Map

This section outlines how to configure a Layer 7 FTP command inspection policy map. The Layer 7 policy map configures the applicable FTP command inspection actions executed on the network traffic that matches the classifications defined in a class map. You then associate the completed Layer 7 FTP command inspection policy with a Layer 3 and Layer 4 policy map to activate the operation on a VLAN interface (see the “Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions” section).

This section contains the following topics:

- Creating a Layer 7 FTP Command Inspection Policy Map
- Adding a Layer 7 FTP Inspection Policy Map Description
- Including Inline Match Statements in a Layer 7 FTP Command Inspection Policy Map
- Associating a Layer 7 FTP Command Inspection Traffic Class with the Traffic Policy
- Specifying the Layer 7 FTP Command Inspection Policy Actions

Creating a Layer 7 FTP Command Inspection Policy Map

You can use the `policy-map type inspect ftp` command in configuration mode to name the traffic policy and initiate FTP command inspection.

The syntax of this command is as follows:

```
policy-map type inspect ftp first-match map_name
```

The keywords and arguments are as follows:

- `ftp first-match`—Specifies a Layer 7 policy map that defines the inspection of FTP commands by the ACE. The `first-match` keyword defines the execution for the Layer 7 FTP command inspection policy map. The ACE executes only the action specified against the first-matching classification.
- `map_name`—Name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.
For example, to create a Layer 7 FTP command inspection policy map, enter:

```
host/Admin(config)# policy-map type inspect ftp first-match FTP_INSPECT_L7POLICY
host/Admin(config-pmap-ftp-ins)#
```

The CLI displays the policy map configuration mode.

To remove a Layer 7 command inspection policy map from the ACE, enter:

```
host1/Admin(config)# no policy-map type inspect ftp first-match FTP_INSPECT_L7POLICY
```

### Adding a Layer 7 FTP Inspection Policy Map Description

You can use the `description` command to provide a brief summary of the Layer 7 FTP inspection policy map.

You must access the policy map FTP inspection configuration mode to specify the `description` command.

The syntax of this command is as follows:

```
description text
```

Use the `text` argument to enter an unquoted text string with a maximum of 240 alphanumeric characters.

To add a description that the policy map is to perform FTP command inspection, enter:

```
host1/Admin(config-pmap-ftp-ins)# description FTP command inspection of incoming traffic
```

To remove the description from the policy map, enter:

```
host1/Admin(config-pmap-ftp-ins)# no description FTP command inspection of incoming traffic
```

### Including Inline Match Statements in a Layer 7 FTP Command Inspection Policy Map

You can include a single inline match criteria in the policy map without specifying a traffic class by entering an applicable Layer 7 `match` command. The inline Layer 7 policy map `match` commands function in the same way as the Layer 7
class map **match** commands. However, when you use an inline **match** command, you can specify an action for only a single match statement in the Layer 7 policy map.

**Note**

To specify actions for multiple match statements, use a class map as described in the “Associating a Layer 7 FTP Command Inspection Traffic Class with the Traffic Policy” section.

The syntax for this command is as follows:

```
match name match_statement [insert-before map_name]
```

The keywords, arguments, and options are as follows:

- **name**—Name assigned to the inline **match** command. Enter an unquoted text string with no spaces. The length of the inline match statement name plus the length of the policy map name with which it is associated cannot exceed a total maximum of 64 alphanumeric characters. For example, if the policy map name is L7_POLICY (nine characters), an inline match statement name under this policy cannot exceed 55 alphanumeric characters (64 - 9 = 55).

- **match_statement**—Inline match criteria to be used by the policy map. See below for details on the **match** commands associated with the Layer 7 FTP command inspection class map.

- **insert-before map_name**—(Optional) Places the inline **match** command ahead of an existing class map in the policy map configuration.

The syntax for the Layer 7 FTP inspection policy map inline **match** commands is as follows:

```
match name request-method {appe | cd | cdup | dele | get | help | mkd | put | rmd | rnfr | rnto | site | stou | syst}
```

See the “Defining FTP Match Request Methods” section for details about the inline **match** command.

For example, to add an inline **match** command to a Layer 7 FTP command policy map, enter:

```
host/Admin(config-pmap-ftp-ins-m)# match FTP_REQUEST_MATCH request-method mkdir
host/Admin(config-pmap-ftp-ins-m)#
```
To remove the inline `match` command from the Layer 7 FTP command policy map, enter:

```bash
host/Admin(config-pmap-ftp-ins)# no match FTP_REQUEST_MATCH
```

**Associating a Layer 7 FTP Command Inspection Traffic Class with the Traffic Policy**

You can associate a traffic class created with the `class-map` command to associate network traffic with the traffic policy by using the `class` command.

The syntax of this command is as follows:

```
class map_name
```

The `map_name` argument is the name of a previously defined traffic class, configured with the `class-map` command, to associate traffic to the traffic policy. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

The CLI displays the policy map class configuration mode.

For example, to specify an existing class map in the Layer 7 policy map, enter:

```bash
host1/Admin(config-pmap-ftp-ins)# class FTP_INSPECT_L7CLASS
host1/Admin(config-pmap-ftp-ins-c)#
```

To remove a class map from a Layer 7 policy map, enter:

```bash
host1/Admin(config-pmap-ftp-ins)# no class FTP_INSPECT_L7CLASS
```

**Specifying the Layer 7 FTP Command Inspection Policy Actions**

By default, the ACE allows all FTP commands to pass. To explicitly deny specific FTP commands, use one of the following commands as the action if the specified FTP traffic matches the classification. You apply the specified action against the single inline `match` command or the specified class map.

```
{deny | mask-reply}
```
The keywords are as follows:

- **deny**—Denies the FTP request commands against the single inline **match** command or specified in the class map by resetting the FTP session.

- **mask-reply**—Applies only to the FTP SYST command and its associated reply. The SYST command is used to find out the type of operating system at the FTP server. The **mask-reply** keyword instructs the ACE to mask the system’s reply to the FTP SYST command by filtering sensitive information from the command output.

For example, to specify the actions in the Layer 7 FTP inspection policy map, enter:

```bash
host1/Admin(config)# policy-map type inspect ftp first-match FTP_INSPECT_L7POLICY
host1/Admin(config-pmap-ftp-ins)# class FTP_INSPECT_L7CLASS
host1/Admin(config-pmap-ftp-ins-c)# mask-reply
```

To disable an action from the Layer 7 FTP inspection policy map, enter:

```bash
host1/Admin(config-pmap-ftp-ins-c)# no mask-reply
```

### Configuring a Layer 7 HTTP Deep Inspection Policy

You can configure HTTP inspection with or without a Layer 7 policy map and with or without URL logging. This section describes how to create a Layer 7 class map and policy map for HTTP deep packet inspection and, optionally, URL logging. The ACE performs a stateful deep packet inspection of the HTTP protocol and permits or restricts traffic based on the actions that you configure in the policy map. For information about configuring basic HTTP inspection at Layer 4, see the “Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy” section. The following security features are included as part of HTTP deep packet inspection as performed by the ACE:

- Regular expression matching on the name in an HTTP header, URL name, or content expressions in an HTTP entity-body
- Content, URL, and HTTP header length checks
- URL query parameter and secondary cookie filtering
- MIME-type message inspection
- Content and MIME type verification and filtering
• Transfer-encoding and method filtering
• Port 80 misuse detection and filtering

Note

You can associate a maximum of 1024 instances of the same type of regular expression (regex) with a Layer 4 policy map. This limit applies to all Layer 7 policy-map types, including generic, HTTP, RADIUS, RDP, RTSP, and SIP. You configure regexes in the following:

• Match statements in Layer 7 class maps
• Inline match statements in Layer 7 policy maps
• Layer 7 hash predictors for server farms
• Layer 7 sticky expressions in sticky groups
• Header insertion and rewrite (including SSL URL rewrite) expressions in Layer 7 action lists

This section contains the following topics:

• Configuring a Layer 7 HTTP Deep Inspection Class Map
• Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map

Configuring a Layer 7 HTTP Deep Inspection Class Map

This section contains the following topics:

• Creating an HTTP Deep Inspection Class Map
• Adding a Layer 7 HTTP Deep Packet Inspection Class Map Description
• Defining HTTP Content Match Criteria
• Defining the Length of the HTTP Content for Inspection
• Defining a Secondary Cookie for HTTP Inspection
• Defining an HTTP Header for Inspection
• Defining the HTTP Maximum Header Length for Inspection
• Defining a Header MIME-Type Messages for Inspection
• Defining an HTTP Traffic Restricted Category
Creating an HTTP Deep Inspection Class Map

You can create a Layer 7 class map for deep packet inspection of HTTP traffic by using the `class-map type http inspect` command in configuration mode.

The syntax of this command is as follows:

```
class-map type http inspect [match-all | match-any] map_name
```

The keywords, arguments, and options are as follows:

- **match-all | match-any**—(Optional) Determines how the ACE performs the deep packet inspection of HTTP traffic when multiple match criteria exist in a class map. The class map is considered a match if the `match` commands meet one of the following conditions:
  - **match-all**—(Default) Network traffic needs to satisfy all of the match criteria (implicit AND) to match the Layer 7 HTTP deep packet inspection class map. The `match-all` keyword is applicable only for match statements of different HTTP deep packet inspection types. For example, specifying a `match-all` condition for URL, HTTP header, and URL content statements in the same class map is valid. However, specifying a `match-all` condition for multiple HTTP headers with the same names or multiple URLs in the same class map is invalid.
  - **match-any**—Network traffic needs to satisfy only one of the match criteria (implicit OR) to match the Layer 7 HTTP deep packet inspection class map. The `match-any` keyword is applicable for match statements of different Layer 7 HTTP deep packet inspection type or multiple instances of the same type with different names. For example, the ACE allows you to specify a `match-any` condition for cookie, HTTP header, and URL content statements in the same class map, but it does not allow you to specify a `match-any` condition for URL length, HTTP header length, and content length statements in the same class map.

- **map_name**—Name assigned to the class map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.
The CLI displays the class map HTTP application protocol inspection configuration mode. To classify the HTTP application inspection of traffic for evaluation by the ACE, include one or more of the following commands to configure the match criteria for the Layer 7 class map:

- **match content**—See the “Defining HTTP Content Match Criteria” section.
- **match content length**—See the “Defining the Length of the HTTP Content for Inspection” section.
- **match cookie secondary**—See the “Defining a Secondary Cookie for HTTP Inspection” section.
- **match header**—See the “Defining an HTTP Header for Inspection” section.
- **match header length**—See the “Defining the HTTP Maximum Header Length for Inspection” section.
- **match header mime-type**—See the “Defining a Header MIME-Type Messages for Inspection” section.
- **match port-misuse**—See the “Defining an HTTP Traffic Restricted Category” section.
- **match request-method**—See the “Defining HTTP Request Methods and Extension Methods” section.
- **match transfer-encoding**—See the “Defining an HTTP Transfer Encoding Type”
- **match url**—See the “Defining an HTTP URL for Inspection” section.
- **match url length**—See the “Defining an HTTP Maximum URL Length for Inspection” section.

You may include multiple **match** commands in the class map.

For example, to specify HTTP_INSPECT_L7CLASS as the name of a class map and identify that at least one command in the Layer 7 HTTP application inspection class map must be satisfied for the ACE to indicate a match, enter:

```
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match header length request eq 200
host1/Admin(config-cmap-http-insp)# match header Host header-value .*mycompanyexample.com
host1/Admin(config-cmap-http-insp)# match url length eq 10000
host1/Admin(config-cmap-http-insp)# match url .*.gif
```
To remove the HTTP application inspection class map from the ACE, enter:

```
host1/Admin(config)#no class-map type http inspect match-any HTTP_INSPECT_L7CLASS
```

### Adding a Layer 7 HTTP Deep Packet Inspection Class Map Description

You can use the `description` command to provide a brief description of the Layer 7 HTTP deep packet inspection class map.

You must access the class map configuration mode to specify the `description` command.

The syntax of this command is as follows:

```
description text
```

Use the `text` argument to enter an unquoted text string with a maximum of 240 alphanumeric characters.

To add a description that the class map is to perform HTTP deep packet inspection, enter:

```
host1/Admin(config-cmap-http-insp)# description HTTP protocol deep inspection of incoming traffic
```

To remove the description from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no description
```

### Defining HTTP Content Match Criteria

You can use the `match content` command to configure the class map to define HTTP application inspection decisions based on content expressions contained within the HTTP entity-body.

You must access the class map configuration mode to specify the `match content` command.

The syntax of this command is as follows:

```
[line_number] match content expression [offset number]
```
Chapter 3  Configuring Application Protocol Inspection

Configuring a Layer 7 HTTP Deep Inspection Policy

The keywords, arguments, and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual match commands. Enter an integer from 2 to 1024 as the line number. You can enter no line_number to delete long match commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **expression**—Content expression contained within the HTTP entity-body. The range is from 1 to 255 alphanumeric characters. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

  **Note** When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([ ]) to match these symbols (for example, enter www[.]xyz[.]com instead of www.xyz.com). You can also use a backslash (\) to escape a dot (.) or a question mark (?).

- **offset number**—Provides an absolute offset where the content expression search string starts. The offset starts at the first byte of the message body, after the empty line (CR,LF,CR,LF) between the headers and the body of the message. The offset value is between 1 to 4000 bytes.

For example, to create a class map that specifies a content expression contained within the entity-body sent with an HTTP request, enter:

```
host1/Admin(config)#class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match content .*newp2psig
```

To clear the content expression checking match criteria from the class map, enter:

```
host1/Admin(config-cmap)# no match content .*newp2psig
```

**Defining the Length of the HTTP Content for Inspection**

You can use the match content length command to configure the class map to define application inspection decisions on HTTP traffic up to the configured maximum content parse length. Messages that meet the specified criteria will be either allowed or denied based on the Layer 7 HTTP deep packet inspection policy map action.
You must access the class map configuration mode to specify the `match content length` command.

The syntax of this command is as follows:

```
[line_number] match content length { eq bytes | gt bytes | lt bytes | range bytes1 bytes2 }
```

The keywords, arguments, and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **eq bytes**—Specifies a value for the content parse length in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with a content length equal to the specified value. Valid entries are from 1 to 65535 bytes.

- **gt bytes**—Specifies a minimum value for the content parse length in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with a content length greater than the specified value. Valid entries are from 1 to 65535 bytes.

- **lt bytes**—Specifies a maximum value for the content parse length in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with a content length size less than the specified value. Valid entries are from 1 to 65535 bytes.

- **range bytes1 bytes2**—Specifies a size range for the content parse length in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with a content length within this range. The range is from 1 to 65535 bytes.

For example, to create a class map that identifies the content length in an HTTP message that can be received by the ACE, enter:

```
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match content length eq 3495
```

To clear the HTTP content length match criteria from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no match content length eq 3495
```
Defining a Secondary Cookie for HTTP Inspection

You can use the `match cookie secondary` command in class map HTTP inspection configuration mode to configure a class map to define inspection decisions based on the name or prefix and value of a secondary cookie (URL query string). Normally, the ACE parses URLs up to, but not including, the question mark (?) in a URL string. This feature extends the URL parsing capabilities of the ACE to include the URL parameters beyond the question mark. The ACE also uses this command to match secondary cookies present in the HTTP content of POST requests. This command is available as either a match statement in a class map or an inline match statement (slightly different syntax) in a Layer 7 policy map. For details about inline match statements, see the “Including Inline Match Statements in a Layer 7 HTTP Deep Packet Inspection Policy Map” section.

The syntax of this command is as follows:

```
match cookie secondary [name cookie_name | prefix prefix_name] value expression
```

The keywords, options, and arguments are as follows:

- **name cookie_name**—Specifies the identifier of the secondary cookie to match. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

- **prefix prefix_name**—(Optional) Specifies the prefix of the secondary cookie to match. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

- **value expression**—(Optional) Specifies the regular expression of the secondary cookie to match. Enter an unquoted text string with no spaces and a maximum of 255 alphanumeric characters. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

**Note**

When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([ ]) to match these symbols (for example, enter www[.]xyz[.]com instead of www.xyz.com). You can also use a backslash (\) to escape a dot (.) or a question mark (?).
### Table 3-5 Special Characters for Matching String Expressions

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>One of any character.</td>
</tr>
<tr>
<td>.*</td>
<td>Zero or more of any character.</td>
</tr>
<tr>
<td>.</td>
<td>Period (escaped).</td>
</tr>
<tr>
<td>[charset]</td>
<td>Match any single character from the range.</td>
</tr>
<tr>
<td>[^charset]</td>
<td>Do not match any character in the range. All other characters represent themselves.</td>
</tr>
<tr>
<td>()</td>
<td>Expression grouping.</td>
</tr>
<tr>
<td>(expr1</td>
<td>expr2)</td>
</tr>
<tr>
<td>(expr)*</td>
<td>0 or more of expression.</td>
</tr>
<tr>
<td>(expr)+</td>
<td>1 or more of expression.</td>
</tr>
<tr>
<td>expr{m,n}</td>
<td>Repeat the expression between m and n times, where m and n have a range from 1 to 255.</td>
</tr>
<tr>
<td>expr{m}</td>
<td>Match the expression exactly m times. The range for m is from 1 to 255.</td>
</tr>
<tr>
<td>expr{m,}</td>
<td>Match the expression m or more times. The range for m is from 1 to 255.</td>
</tr>
<tr>
<td>\a</td>
<td>Alert (ASCII 7).</td>
</tr>
<tr>
<td>\b</td>
<td>Backspace (ASCII 8).</td>
</tr>
<tr>
<td>\f</td>
<td>Form-feed (ASCII 12).</td>
</tr>
<tr>
<td>\n</td>
<td>New line (ascii 10).</td>
</tr>
<tr>
<td>\r</td>
<td>Carriage return (ASCII 13).</td>
</tr>
<tr>
<td>\t</td>
<td>Tab (ASCII 9).</td>
</tr>
<tr>
<td>\v</td>
<td>Vertical tab (ASCII 11).</td>
</tr>
<tr>
<td>\0</td>
<td>Null (ASCII 0).</td>
</tr>
<tr>
<td>\</td>
<td>Backslash.</td>
</tr>
<tr>
<td>\x##</td>
<td>Any ASCII character as specified in two-digit hexadecimal notation.</td>
</tr>
</tbody>
</table>
The following configuration guidelines apply when you configure a secondary cookie match statement for HTTP inspection:

- Ensure that secondary cookie names do not overlap with other secondary cookie names in the same match-all class map. For example, the following configuration is not allowed because the two match statements have overlapping cookie names:

  ```
  host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
  host1/Admin(config-cmap-http-insp)# match cookie secondary prefix id value .*
  host1/Admin(config-cmap-http-insp)# match cookie secondary name identity value bob
  ```

- When you configure a secondary cookie value match across all secondary cookie names in a match-all class map, you cannot configure any other secondary cookie match in the same class map because a secondary cookie match on a value alone is equivalent to a wildcard match on a name. In the following example, the second match statement is not allowed:

  ```
  host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
  host1/Admin(config-cmap-http-insp)# match cookie secondary value bob
  host1/Admin(config-cmap-http-insp-m)# exit
  host1/Admin(config-cmap-http-insp)# match cookie secondary name identity value jane
  ```

For example, to match a secondary cookie called “matchme” with a regular expression value of .*abc123, enter:

  ```
  host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
  host1/Admin(config-cmap-http-insp)# match cookie secondary name matchme value .*abc123
  ```

For example, to match all cookie names starting with the letters “ab”, enter:

  ```
  host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
  host1/Admin(config-cmap-http-insp)# match cookie secondary prefix ab value .*
  ```

For example, to match a given regex in all secondary cookie values, enter:

  ```
  host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
  ```
host1/Admin(config-cmap-http-insp)# match cookie secondary value
.*machine-key

To remove a secondary cookie match statement from a class map, enter the no form of the command as follows:

host1/Admin(config)# class-map type http inspect match-any
HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# no match cookie secondary value
.*machine-key

Defining an HTTP Header for Inspection

You can use the match header command to configure the class map to define application inspection decisions based on the name and value in an HTTP header. The ACE performs regular expression matching against the received packet data from a particular connection based on the HTTP header expression.

You must access the class map configuration mode to specify the match header command.

The syntax of this command is as follows:

```
[line_number] match header {header_name | header_field} header-value expression
```

The keywords, arguments, and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual match commands. Enter an integer from 2 to 1024 as the line number. You can enter no line_number to delete long match commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **header_name**—Name of the HTTP header to match (for example, www.example1.com.) Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. Alternatively, you can enter a text string with spaces if you enclose the entire string in quotation marks (""). For a list of predefined header fields, see Table 3-6.

**Note** The header_name argument cannot include the colon in the name of the HTTP header; the ACE rejects the colon as an invalid token.
- header_field—Standard HTTP/1.1 header field. Valid selections include request-header fields, general-header fields, and entity-header field. Table 3-6 lists the supported HTTP/1.1 header fields.

**Table 3-6  HTTP/1.1 Header Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>Semicolon-separated list of representation schemes (content type metainformation values) that will be accepted in the response to the request.</td>
</tr>
<tr>
<td>Accept-Charset</td>
<td>Character sets that are acceptable for the response. This field allows clients capable of understanding more comprehensive or special-purpose character sets to signal that capability to a server that can representing documents in those character sets.</td>
</tr>
<tr>
<td>Accept-Encoding</td>
<td>Restricts the content encoding that a user will accept from the server.</td>
</tr>
<tr>
<td>Accept-Language</td>
<td>ISO code for the language in which the document is written. The language code is an ISO 3316 language code with an optional ISO639 country code to specify a national variant.</td>
</tr>
<tr>
<td>Authorization</td>
<td>Specifies that the user agent wants to authenticate itself with a server, usually after receiving a 401 response.</td>
</tr>
<tr>
<td>Cache-Control</td>
<td>Directives that must be obeyed by all caching mechanisms in the request/response chain. The directives specify behavior intended to prevent caches from adversely interfering with the request or response.</td>
</tr>
<tr>
<td>Connection</td>
<td>Connection options that the sender can specify.</td>
</tr>
<tr>
<td>Content-MD5</td>
<td>MD5 digest of the entity-body that provides an end-to-end integrity check. Only a client or an origin server can generate this header field.</td>
</tr>
<tr>
<td>Expect</td>
<td>Used by a client to inform the server about the behaviors that the client requires.</td>
</tr>
</tbody>
</table>
Table 3-6  HTTP/1.1 Header Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From</td>
<td>E-mail address of the person that controls the requesting user agent.</td>
</tr>
<tr>
<td>Host</td>
<td>Internet host and port number of the resource being requested, as obtained from the original URI given by the user or referring resource. The Host field value must represent the naming authority of the origin server or gateway given by the original URL.</td>
</tr>
<tr>
<td>If-Match</td>
<td>Used with a method to make it conditional. A client that has one or more entities previously obtained from the resource can verify that one of those entities is current by including a list of their associated entity tags in the If-Match header field. This feature allows efficient updates of cached information with a minimum amount of transaction overhead. It is also used, on updating requests, to prevent inadvertent modification of the wrong version of a resource. As a special case, the value “*” matches any current entity of the resource.</td>
</tr>
<tr>
<td>Pragma</td>
<td>Pragma directives understood by servers to whom the directives are relevant. The syntax is the same as for other multiple-value fields in HTTP. For example, the accept field is a comma-separated list of entries for which the optional parameters are separated by semicolons.</td>
</tr>
<tr>
<td>Referer</td>
<td>Address (URI) of the resource from which the URI in the request was obtained.</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>What (if any) type of transformation has been applied to the message body in order to safely transfer it between the sender and the recipient.</td>
</tr>
</tbody>
</table>
Table 3-6  HTTP/1.1 Header Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-Agent</td>
<td>Information about the user agent such as a software program that originates the request. This information is for statistical purposes, the tracing of protocol violations, and automated recognition of user agents so that you can customize responses to avoid particular user agent limitations.</td>
</tr>
<tr>
<td>Via</td>
<td>Used by gateways and proxies to indicate the intermediate protocols and recipients between the user agent and the server on requests, and between the origin server and the client on responses.</td>
</tr>
</tbody>
</table>

- **header-value expression**—Specifies the header value expression string to compare against the value in the specified field in the HTTP header. The range is from 1 to 255 alphanumeric characters. There are predefined header fields, such as Accept-Language, User-Agent, or Host. The ACE supports the use of regular expressions for matching. Expressions are stored in a header map in the form `header-name: expression`. Header expressions allow spaces, if the spaces are escaped or quoted. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

**Note**  
When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([ ]) to match these symbols (for example, enter `www[.]xyz[.]com` instead of `www.xyz.com`). You can also use a backslash (\) to escape a dot (.) or a question mark (?).

For example, to specify that the Layer 7 class map is to match and perform application inspection on HTTP headers, enter:

```
host1/Admin(config)# class-map type http inspect HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap)# match header Host header-value .mycompanyexample.com
```
For example, to specify regular expressions in a class map to emulate a wildcard search to match the header value expression string, enter:

```plaintext
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match header Host header-value .*myfirstcompanyexample.com
host1/Admin(config-cmap-http-insp)# match header Host header-value .*mysecondcompanyexample.com
```

To clear an HTTP header match criteria from the class map, enter:

```plaintext
host1/Admin(config-cmap-http-insp)# no match header Host header-value .*mysecondcompanyexample.com
```

### Defining the HTTP Maximum Header Length for Inspection

By default, the maximum header length for HTTP deep packet inspection is 2048 bytes. Use the `match header length` command to limit the HTTP traffic allowed through the ACE based on the length of the entity-body in the HTTP message. Messages are either allowed or denied based on the Layer 7 HTTP deep packet inspection policy map action.

You must access the class map configuration mode to specify the `match header length` command.

The syntax of this command is as follows:

```plaintext
[line_number] match header length {request | response} {eq bytes | gt bytes | lt bytes | range bytes1 bytes 2}
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.
- `request`—Specifies the size of the HTTP header request message that can be received by the ACE.
- `response`—Specifies the size of the HTTP header response message sent by the ACE.
• **eq bytes**—Specifies a value for the entity-body in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with an entity-body size equal to the specified value. Valid entries are from 1 to 65535 bytes.

• **gt bytes**—Specifies a minimum value for the entity-body in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with an entity-body size greater than the specified value. Valid entries are from 1 to 65535 bytes.

• **lt bytes**—Specifies a maximum value for the entity-body in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with an entity-body size less than the specified value. Valid entries are from 1 to 65535 bytes.

• **range bytes1 bytes2**—Specifies a size range for the entity-body in an HTTP message received by the ACE. Based on the policy map action, the ACE allows or denies messages with an entity-body size within this range. The range is from 1 to 65535 bytes.

For example, to specify that the class map is to match on HTTP traffic received with a length less than or equal to 3600 bytes in the entity-body of the HTTP message, enter:

```markdown
host1/Admin(config)# class-map type http inspect HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match header length request eq 3600
```

To clear the maximum HTTP header length match criteria from the class map, enter:

```markdown
host1/Admin(config-cmap-http-insp)# no match header length request eq 3600
```

**Defining a Header MIME-Type Messages for Inspection**

You can use the **match header mime-type** command to specify a subset of the Multipurpose Internet Mail Extension (MIME)-type messages that the ACE permits or denies based on the actions in the policy map. MIME-type validation extends the format of Internet mail to allow non-US-ASCII textual messages, nontextual messages, multipart message bodies, and non-US-ASCII information in message headers.
To define MIME-type messages in addition to what is supported under the `match header mime-type` command, use the `match header` command. For example, to define a match for a new MIME-type audio/myaudio, you could enter the following match statement: `match header Content-type header-value audio/myaudio`. See the “Defining an HTTP Header for Inspection” section for details.

The syntax of this command is as follows:

```
[line_number] match header mime-type mime_type
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter no `line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `mime_type`—Predefined list of mime-types, such as image/Jpeg, text/html, application/msword, and audio/mpeg. Choose whether only the mime-types included in this list are permitted through the ACE or whether all mime-types are acceptable. The default behavior is to allow all mime-types.

The supported mime-types are as follows:

- application/msexcel
- application/mspowerpoint
- application/msword
- application/octet-stream
- application/pdf
- application/postscript
- application/x-gzip
- application/x-java-archive
- application/x-java-vm
- application/x-messenger
- application/zip
- audio/*
- audio/basic
- audio/midi
- audio/mpeg
- audio/x-adpcm
- audio/x-aiff
- audio/x-ogg
- audio/x-wav
- image/*
- image/gif
- image/jpeg
- image/png
- image/tiff
- image/x-3ds
- image/x-bitmap
- image/x-niff
- image/x-portable-bitmap
- image/x-portable-greymap
- image/x-xpm
- text/*
- text/css
- text/html
- text/plain
- text/richtext
- text/sgml
- text/xmcd
- text/xml
- video/*
- video/flc
- video/mpeg
Follow these guidelines when using the `match header mime-type` command:

- You can specify multiple `match header mime-type` commands within a class map.
- Each `match header mime-type` command configures a single application type.

For example, to create a class map that specifies the MIME-type audio/midi and audio/mpeg messages permitted through the ACE, enter:

```
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match header mime-type audio/midi
host1/Admin(config-cmap-http-insp)# match header mime-type audio/mpeg
```

To deselect the specified MIME message match criteria from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no match header mime-type audio/midi
```

### Defining an HTTP Traffic Restricted Category

You can use the `match port-misuse` command to configure the class map to define application inspection compliance decisions that restrict certain HTTP traffic from passing through the ACE. This class map detects the misuse of port 80 (or any other port running HTTP) for tunneling protocols such as peer-to-peer (p2p) applications, tunneling applications, and instant messaging.

You must access the class map configuration mode to specify the `match port-misuse` command.

The syntax of this command is as follows:

```
[line_number] match port-misuse application_category
```

The keywords, arguments, and options are as follows:
- **line_number**—(Optional) Argument that assists you in editing or deleting individual **match** commands. Enter an integer from 2 to 1024 as the line number. You can enter **no line_number** to delete long **match** commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **application_category**—Restricted HTTP application category for the class map. The possible values for **application_category** are as follows:
  - **im**—Instant messaging application category. The ACE checks for the Yahoo Messenger instant messaging application.
  - **p2p**—Peer-to-peer application category. The applications checked include Kazaa and Gnutella.
  - **tunneling**—Tunneling application category. The applications checked include: HTTPort/HTTHost, GNU Httpunnel, and Firethru.

Follow these guidelines when using the **match port-misuse** command:
- You can specify multiple **match port-misuse** commands within a class map.
- Each **match port-misuse** command configures a single application type.
- The port misuse application inspection process requires a search of the entity-body of the HTTP message, which may degrade performance of the ACE.
- The ACE disables the **match port-misuse** command by default. If you do not configure a restricted HTTP application category, the default action by the ACE is to allow the applications without generating a log.

For example, to create a class map that identifies peer-to-peer applications as restricted HTTP traffic, enter:

```
host1/Admin(config)# class-map type http inspect HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match port-misuse p2p
```

To clear the HTTP restricted application category match criteria from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no match port-misuse p2p
```

**Defining HTTP Request Methods and Extension Methods**

By default, the ACE allows all request and extension methods. You can use the **match request-method** command to configure the class map to define application inspection compliance decisions based on the request methods defined in
RFC 2616 and by HTTP extension methods. If the HTTP request method or extension method compliance checks fails, the ACE denies or resets the specified HTTP traffic based on the policy map action.

You must access the class map configuration mode to specify the `match request-method` command.

The syntax of this command is as follows:

```
[line_number] match request-method { ext method | rfc method }
```

The keywords, arguments, and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **ext method**—Specifies an HTTP extension method. If the RFC request messages does not contain one of the RFC 2616 HTTP request methods, the ACE verifies if it is an extension method. The ACE supports the inspection of the following HTTP request extension methods: `bcopy`, `bdelete`, `bmove`, `bpropfind`, `bpropatch`, `copy`, `edit`, `getattr`, `getattrname`, `getprops`, `index`, `lock`, `mkcol`, `mkdir`, `move`, `notify`, `poll`, `propfind`, `proppatch`, `revadd`, `replabel`, `revlog`, `revnum`, `save`, `search`, `setattr`, `startrev`, `stoprev`, `subscribe`, `unsubscribe`, `unedits`, `unlock`, and `x-ms-emumatts`.

- **rfc method**—Specifies an RFC 2616 HTTP request method that you want to perform an RFC compliance check on. The ACE supports the inspection of the following RFC 2616 HTTP request methods: `connect`, `delete`, `get`, `head`, `options`, `post`, `put`, and `trace`.

Follow these guidelines when using the `match request-method` command:

- You can specify multiple `match request-method` commands within a class map.
- Each `match request-method` command configures a single request method.
- For unsupported HTTP request methods, include the `inspect http strict` command as an action in the Layer 3 and Layer 4 policy map.
- The ACE disables the `match request-method` command by default. If you do not configure a request method, the default action by the ACE is to allow the RFC 2616 HTTP request method without generating a log.
For example, to create a class map that identifies the connect, get, head, and index HTTP RFC 2616 protocols for HTTP application protocol inspection, enter:

```
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match request-method rfc connect
host1/Admin(config-cmap-http-insp)# match request-method rfc get
host1/Admin(config-cmap-http-insp)# match request-method rfc head
host1/Admin(config-cmap-http-insp)# match request-method ext index
```

To clear an RFC 2616 HTTP request method match criteria from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no match request-method rfc connect
```

### Defining an HTTP Transfer Encoding Type

You can use the `match transfer-encoding` command to configure the class map to define application inspection decisions that limit the HTTP transfer-encoding types that can pass through the ACE. The transfer-encoding general-header field indicates the type of transformation, if any, that has been applied to the HTTP message body to safely transfer it between the sender and the recipient. When an HTTP request message contains the configured transfer-encoding type, the ACE performs the configured action in the policy map.

You must access the class map configuration mode to specify the `match transfer-encoding` command.

The syntax of this command is as follows:

```
[line_number] match transfer-encoding coding_types
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.
- **coding_types**—HTTP transfer-encoding type for the class map. The possible values for `coding_types` are as follows:
  - **chunked**—Message body is transferred as a series of chunks.
  - **compress**—Encoding format produced by the common UNIX file compression program “compress.” This format is an adaptive Lempel-Ziv-Welch coding (LZW).
  - **deflate**—The .zlib format defined in RFC 1950 with the deflate compression mechanism described in RFC 1951.
  - **gzip**—Encoding format produced by the file compression program gzip (GNU zip) as described in RFC 1952. This format is a Lempel-Ziv coding (LZ77) with a 32-bit CRC.
  - **identity**—Default (identity) encoding, which does not require the use of transformation.

Follow these guidelines when using the **match transfer-encoding** command:

- You can specify multiple **match transfer-encoding** commands within a class map.
- Each **match transfer-encoding** command configures a single application type.
- The ACE disables the **match transfer-encoding** command by default.

For example, to create a class map that specifies a chunked HTTP transfer encoding type to limit the HTTP traffic that flows through the ACE, enter:

```plaintext
host1/Admin(config)# class-map type http inspect HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match transfer-encoding chunked
```

To clear the HTTP transfer-encoding match criteria from the class map, enter:

```plaintext
host1/Admin(config-cmap-http-insp)# no match transfer-encoding chunked
```

**Defining an HTTP URL for Inspection**

You can use the **match url** command to configure the class map to define application inspection decisions based on the URL name. HTTP performs regular expression matching against the received packet data from a particular connection based on the URL expression.

You must access the class map configuration mode to specify the **match url** command.
The syntax of this command is as follows:

```
[line_number] match url expression
```

The keywords, arguments, and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual match commands. Enter an integer from 2 to 1024 as the line number. You can enter no line_number to delete long match commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **expression**—URL, or portion of a URL, to match. The URL string range is from 1 to 255 characters. Include only the portion of the URL that follows www.hostname.domain in the match statement. For example, in the URL www.anydomain.com/latest/whatsnew.html, include only /latest/whatsnew.html. To match the www.anydomain.com portion, the URL string can take the form of a URL regular expression. The ACE supports the use of regular expressions for matching. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

**Note** When matching URLs, the period (.) does not have a literal meaning in regular expressions. Use either brackets ([]) or the backslash (\) character to match this symbol; for example, specify www[.]xyz[.]com instead of www.xyz.com.

For example, to specify that the Layer 7 class map is to match and perform application inspection on a specific URL, enter:

```
host1/Admin(config)# class-map type http inspect HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match url whatsnew/latest.*
```

For example, to use regular expressions to emulate a wildcard search to match on any .gif or .html file, enter:

```
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match url .*\.gif
host1/Admin(config-cmap-http-insp)# match url .*\.html
```

To clear a URL match criteria from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no match url .*\.gif
```
Defining an HTTP Maximum URL Length for Inspection

You can use the `match url length` command to limit the HTTP traffic allowed through the ACE by specifying the maximum length of a URL in a request message that can be received by the ACE. Messages will be either allowed or denied based on the Layer 7 HTTP deep packet inspection policy map action.

You must access the class map configuration mode to specify the `match url length` command.

The syntax of this command is as follows:

```
[line_number] match url length {eq bytes | gt bytes | lt bytes | range bytes1 bytes 2}
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `eq bytes`—Specifies a value for the HTTP URL length received by the ACE. Based on the policy map action, the ACE allows or denies messages with an HTTP URL length equal to the specified value. Valid entries are from 1 to 65535 bytes.

- `gt bytes`—Specifies a minimum value for the HTTP URL length received by the ACE. Based on the policy map action, the ACE allows or denies messages with an HTTP URL length greater than the specified value. Valid entries are from 1 to 65535 bytes.

- `lt bytes`—Specifies a maximum value for the HTTP URL length received by the ACE. Based on the policy map action, the ACE allows or denies messages with an HTTP URL length less than the specified value. Valid entries are from 1 to 65535 bytes.

- `range bytes1 bytes2`—Specifies a size range for the HTTP URL length received by the ACE. Based on the policy map action, the ACE allows or denies messages with a URL length within this range. The range is from 1 to 65535 bytes.
For example, to specify that a class map is to match on a URL with a length equal to 10000 bytes in the request message, enter:

```
host1/Admin(config)# class-map type http inspect HTTP_INSPECT_L7CLASS
host1/Admin(config-cmap-http-insp)# match url length eq 10000
```

To clear a URL length match criteria from the class map, enter:

```
host1/Admin(config-cmap-http-insp)# no match url length eq 10000
```

### Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map

This section describes how to configure a Layer 7 HTTP deep inspection policy map. The Layer 7 policy map configures the applicable HTTP deep packet inspection actions executed on the network traffic that match the classifications defined in a class map. You then associate the completed Layer 7 HTTP deep packet inspection policy with a Layer 3 and Layer 4 policy map to activate the operation on a VLAN interface (see the “Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions” section).

This section contains the following topics:

- Creating a Layer 7 HTTP Deep Packet Inspection Policy Map
- Adding a Layer 7 HTTP Deep Packet Inspection Policy Map Description
- Including Inline Match Statements in a Layer 7 HTTP Deep Packet Inspection Policy Map
- Associating a Layer 7 HTTP Inspection Traffic Class with the Traffic Policy
- Specifying the Layer 7 HTTP Deep Packet Policy Actions

### Creating a Layer 7 HTTP Deep Packet Inspection Policy Map

You can use the `policy-map type inspect http` command in configuration mode to name the traffic policy and initiate Layer 7 HTTP deep packet inspection.

The syntax of this command is as follows:

```
policy-map type inspect http all-match map_name
```
The keyword and arguments are as follows:

- **http all-match**—Specifies the policy map that initiates the deep packet inspection of the HTTP protocol by the ACE. The ACE attempts to match all specified conditions against the matching classification and executes the actions of all matching classes until it encounters a deny for a match request.

- **map_name**—Name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to create a Layer 7 HTTP deep packet inspection policy map, enter:

```
host/Admin(config)# policy-map type inspect http all-match HTTP_INSPECT_L7POLICY
host/Admin(config-pmap-ins-http)#
```

The CLI displays the policy map configuration mode.

To remove a Layer 7 HTTP deep packet inspection policy map from the ACE, enter:

```
host1/Admin(config)# no policy-map type inspect http all-match HTTP_INSPECT_L7POLICY
```

**Adding a Layer 7 HTTP Deep Packet Inspection Policy Map Description**

You can use the `description` command to provide a brief summary of the Layer 7 HTTP deep packet inspection policy map.

You must access the policy map configuration mode to specify the `description` command.

The syntax of this command is as follows:

```
description text
```

Use the `text` argument to enter an unquoted text string with a maximum of 240 alphanumeric characters.

To add a description that the policy map is to perform HTTP deep packet inspection, enter:

```
host1/Admin(config-pmap-ins-http)# description HTTP protocol deep inspection of incoming traffic
```

To remove the description from the policy map, enter:

```
host1/Admin(config-pmap-ins-http)# no description
```
Including Inline Match Statements in a Layer 7 HTTP Deep Packet Inspection Policy Map

You can include a single inline match criteria in the policy map without specifying a traffic class by entering an applicable Layer 7 match command. The inline Layer 7 policy map match commands function the same as with the Layer 7 class map match commands. However, when you use an inline match command, you can specify an action for only a single match statement in the Layer 7 policy map.

Note

To specify actions for multiple match statements, use a class map as described in the “Associating a Layer 7 HTTP Inspection Traffic Class with the Traffic Policy” section.

The syntax of this command is as follows:

```
match name match_statement [insert-before map_name]
```

The keywords, arguments, and options are as follows:

- **name**—Name assigned to the inline match command. Enter an unquoted text string with no spaces. The length of the inline match statement name plus the length of the policy map name with which it is associated cannot exceed a total maximum of 64 alphanumeric characters. For example, if the policy map name is L7_POLICY (nine characters), an inline match statement name under this policy cannot exceed 55 alphanumeric characters (64 - 9 = 55).

- **match_statement**—Inline match criteria to be used by the policy map. See below for details on the individual match commands associated with the Layer 7 HTTP deep inspection class map.

- **insert-before map_name**—(Optional) Places the inline match command ahead of an existing class map in the policy map configuration.

The syntax for the HTTP deep packet inspection policy map inline match commands is as follows:

```
match name content expression [offset number]
```

```
match name content length { eq bytes | gt bytes | lt bytes | range bytes1 bytes2 }
```

```
match name content-type-verification
```
match name cookie secondary [name cookie_name | prefix prefix_name] value expression

match name header {header_name \ header_field} header-value expression

match name header length {request | response} {eq bytes | gt bytes | lt bytes | range bytes1 bytes 2}

match name header mime-type mime_type

match name port-misuse application_category

match name request-method {ext method | rfc method}

match name strict-http

match name transfer-encoding coding_types

match name url expression

match name url length {eq bytes | gt bytes | lt bytes | range bytes1 bytes 2}

See the “Configuring a Layer 7 HTTP Deep Inspection Class Map” section for details on the individual inline match commands.

The match content-type-verification and match strict-http commands are available only as inline match commands under the Layer 7 policy-map type inspect http command. Because these two Layer 7 HTTP deep inspection match criteria cannot be combined with other match criteria, they appear as inline match commands for a policy map.

These two match commands perform the following HTTP deep inspection functions:

- **match content-type-verification**—Verifies the content MIME-type messages with the header MIME-type. This inline match command limits the MIME-types in HTTP messages allowed through the ACE. It verifies that the header MIME-type value is in the internal list of supported MIME-types, and the header MIME-type matches the actual content in the data or entity-body portion of the message. If they do not match, the ACE performs one of the specified Layer 7 policy map actions: permit or reset.

| Note | The MIME-type HTTP inspection process requires a search up to the configured maximum content parse length of the HTTP message, which may degrade performance of the ACE. |
• **match strict-http**—Enforces that the internal compliance checks verify that a message is compliant with the HTTP RFC standard, RFC 2616. If the HTTP message is not compliant, the ACE performs one of the specified Layer 7 policy map actions: **permit** or **reset**.

| Note | Strict HTTP parsing with the **reset** action is enabled by default. You can configure the **match strict-http** inline match statement with the **permit** action to override the default. |

For example, to add an inline **match** command to a Layer 7 HTTP deep inspection policy map, enter:

```
host/Admin(config-pmap-ins-http)# match L7httpinspect port-misuse p2p
```

### Associating a Layer 7 HTTP Inspection Traffic Class with the Traffic Policy

You can associate a traffic class created with the **class-map** command to associate network traffic with the traffic policy by using the **class** command.

The syntax of this command is as follows:

```
class map_name
```

The **map_name** argument is the name of a previously defined traffic class, configured with the **class-map** command, to associate traffic to the traffic policy. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

The CLI displays the policy map class configuration mode.

For example, to specify an existing class map in the Layer 7 policy map, enter:

```
host1/Admin(config-pmap-ins-http)# class HTTP_INSPECT_L7CLASS
host1/Admin(config-pmap-ins-http-c)#
```

To remove a class map from a Layer 7 policy map, enter:

```
host1/Admin(config-pmap-ins-http)# no class HTTP_INSPECT_L7CLASS
```

To manually insert a class map ahead of a previously specified class map, use the **insert-before** command. The ACE does not save sequence reordering through the **insert-before** command as part of the configuration.
The syntax of this command is as follows:

```plaintext
class map_name1 insert-before map_name2
```

The keywords and arguments are as follows:

- **class map_name1**— Specifies the name of a previously defined traffic class configured with the `class-map` command. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

- **insert-before map_name2**— Places the current class map ahead of an existing class map as specified by the `map_name2` argument. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to use the `insert-before` command to define the sequential order of two class maps in the policy map, enter:

```sh
host1/Admin(config-pmap-ins-http)# class HTTP_INSPECT_L7CLASSMAP2
insert-before HTTP_INSPECT_L7CLASS
```

To specify the `class-default` class map for the traffic policy, use the `class` `class-default` command. All traffic that fails to meet the other matching criteria in the named class map belongs to the default traffic class. If none of the specified classifications match, the ACE then matches the action specified under the `class` `class-default` command. The `class-default` class map has an implicit `match any` statement in it so that it matches all traffic.

**Note**

By default, all matches are applied to both HTTP request and response messages, but the `class class-default` command is applied only to HTTP requests.

For example, to use the `class class-default` command, enter:

```sh
host1/Admin(config-pmap-ins-http)# class class-default
host1/Admin(config-pmap-ins-http-c)#
```

The CLI displays the policy map class configuration mode.

**Specifying the Layer 7 HTTP Deep Packet Policy Actions**

The default behavior of the ACE is to permit HTTP traffic. For example, if a policy map explicitly permits the HTTP GET method, other methods such as PUT will also be permitted. Only an explicit deny can drop traffic.
Specify the **permit** or **reset** command to define the action that the ACE performs on the HTTP traffic depending on whether it matches the specified commands. You apply the specified command against the single inline **match** command or the specified class map.

The Layer 7 HTTP deep packet inspection policy commands are as follows:

```
{ permit | reset }
```

The keywords are as follows:

- **permit**—Allows the specified HTTP traffic to be received by the ACE if it passes the HTTP deep packet inspection match criteria specified in either the class map or an inline **match** command.
- **reset**—Denies the specified HTTP traffic by sending a TCP reset message to the client or server to close the connection.

For example, to specify the actions in the Layer 7 HTTP deep packet inspection policy map, enter:

```
host1/Admin(config)# policy-map type inspect http all-match HTTP_DEEPPROTOCOL_L7POLICY

host1/Admin(config-pmap-ins-http)# class http_check
host1/Admin(config-pmap-ins-http-c)# permit
```

Because the default is to permit all HTTP packets, you must remove the class map to disable this function. For example, enter:

```
host1/Admin(config-pmap-ins-http)# no class http_check
```

By default, HTTP inspection allows traffic that does not match any of the configured Layer 7 HTTP deep packet inspection matches. You can modify this behavior by including the **class class-default** command with the **reset** action to deny the specified Layer 7 HTTP traffic. In this case, if none of the class matches configured in the Layer 7 HTTP deep packet inspection policy map are hit, the **class-default** action will be taken by the ACE. For example, you can include a class map to allow the HTTP GET method and use the **class class-default** command to block all of the other requests.

---

**Note**

By default, all matches are applied to both HTTP request and response messages, but the **class class-default** command is applied only to HTTP requests.
Configuring a Layer 7 SCCP Inspection Policy

This section describes how to configure a Layer 7 SCCP inspection policy map. Throughout the CLI, SCCP is referred to as “skinny.” A Layer 7 class map is not required for this feature. The ACE uses the SCCP inspection policy to filter traffic based on the message ID and to perform user-configurable actions on that traffic.

You can associate a maximum of 1024 instances of the same type of regular expression (regex) with a Layer 4 policy map. This limit applies to all Layer 7 policy-map types, including generic, HTTP, RADIUS, RDP, RTSP, and SIP. You configure regexes in the following:

- Match statements in Layer 7 class maps
- Inline match statements in Layer 7 policy maps
- Layer 7 hash predictors for server farms
- Layer 7 sticky expressions in sticky groups
- Header insertion and rewrite (including SSL URL rewrite) expressions in Layer 7 action lists

This section contains the following topics:

- Creating a Layer 7 SCCP Inspection Policy Map
- Adding a Description to the Layer 7 SCCP Inspection Policy Map
- Including an Inline Match Statement in a Layer 7 SCCP Inspection Policy Map
- Specifying the Layer 7 SCCP Inspection Policy Map Action

Creating a Layer 7 SCCP Inspection Policy Map

You can create a Layer 7 SCCP inspection policy map by using the `policy-map type inspect skinny` command in configuration mode. The syntax of this command is as follows:

```
policy-map type inspect skinny name
```
The map_name argument is the name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to create a Layer 7 SCCP inspection policy map, enter:

```
host1/Admin(config)# policy-map type inspect skinny
SCCP_INSPECT_L7POLICY
host1/Admin(config-pmap-ins-skinny)#
```

To remove the Layer 7 SCCP inspection policy map from the configuration, enter:

```
host1/Admin(config)# no policy-map type inspect skinny
SCCP_INSPECT_L7POLICY
```

### Adding a Description to the Layer 7 SCCP Inspection Policy Map

You can add a description to a Layer 7 SCCP inspection policy map by using the description command in policy map inspection Skinny configuration mode. The syntax of this command is as follows:

```
description
```

For example, enter:

```
host1/Admin(config-pmap-ins-skinny)# description this is an SCCP inspection policy map
```

To remove the policy map description from the configuration, enter:

```
host1/Admin(config-pmap-ins-skinny)# no description
```

### Including an Inline Match Statement in a Layer 7 SCCP Inspection Policy Map

You can include a single inline match criteria in the policy map without specifying a traffic class by using the match message-id command in policy map inspection Skinny configuration mode. When you use an inline match command, you can specify an action for only a single match statement in the Layer 7 policy map.
The syntax of this command is as follows:

```
match name message-id {number1 {insert-before map_name} | range {number2 number3}}
```

The keywords and arguments are as follows:

- **name**—Name assigned to the inline `match` command. Enter an unquoted text string with no spaces. The length of the inline match statement name plus the length of the policy map name with which it is associated cannot exceed a total maximum of 64 alphanumeric characters. For example, if the policy map name is L7_POLICY (nine characters), an inline match statement name under this policy cannot exceed 55 alphanumeric characters (64 - 9 = 55).

- **message-id**—Specifies an SCCP StationMessageID.

- **number1**—Numerical identifier of the SCCP message. Enter an integer from 0 to 65535.

- **insert-before map_name**—(Optional) Places the inline `match` command ahead of an existing class map in the policy map configuration.

- **range {number2 number3}**—Specifies a range of SCCP message IDs. Enter an integer from 0 to 65535 for the lower and the upper limits of the range. The upper limit must be greater than or equal to the lower limit.

For example, to specify an inline `match` command for a Layer 7 SCCP inspection policy map, enter:

```
host1/Admin(config-pmap-ins-skinny)# match SCCP_MATCH message-id range 100 500
host1/Admin(config-pmap-ins-skinny-m)#
```

To remove the inline match statement from the policy map, enter:

```
host1/Admin(config-pmap-ins-skinny)# no match SCCP_MATCH message-id range 100 500
```

### Specifying the Layer 7 SCCP Inspection Policy Map Action

By default, the ACE allows all SCCP packets to pass. To explicitly drop SCCP traffic, use the `reset` command as the policy map action if the specified SCCP traffic matches the inline match statement. You apply the specified action against the single inline `match` command in policy map inspection Skinny match configuration mode.
The syntax of this command is as follows:

```
reset
```

The `reset` command causes the ACE to drop the SCCP traffic that matches the inline `match` command.

For example, to specify the ACE is to drop SCCP traffic that matches the `match message-id` inline command, enter:

```
host1/Admin(config)# policy-map type inspect sccp 
SCCP_INSPECT_L7POLICY
host1/Admin(config-pmap-ins-skinny)# match SCCP_MATCH message-id range 100 500
host1/Admin(config-pmap-ins-skinny-m)# reset
```

To disable the action in the Layer 7 SCCP inspection policy map, enter:

```
host1/Admin(config-pmap-ins-skinny-m)# no reset
```

## Configuring a Layer 7 SIP Inspection Policy

This section describes how to configure Layer 7 SIP inspection class maps and policy maps. The ACE uses class maps to filter SIP traffic based on a variety of parameters such as, the called party, the calling party, content type, SIP URI, and so on. The ACE uses policy maps to permit or deny that traffic, depending on the actions that you specify.

**Note**

You can associate a maximum of 1024 instances of the same type of regular expression (regex) with a a Layer 4 policy map. This limit applies to all Layer 7 policy-map types, including generic, HTTP, RADIUS, RDP, RTSP, and SIP. You configure regexes in the following:

- Match statements in Layer 7 class maps
- Inline match statements in Layer 7 policy maps
- Layer 7 hash predictors for server farms
- Layer 7 sticky expressions in sticky groups
• Header insertion and rewrite (including SSL URL rewrite) expressions in Layer 7 action lists

This section contains the following topics:
• Configuring a Layer 7 SIP Inspection Class Map
• Configuring a Layer 7 SIP Inspection Policy Map

**Configuring a Layer 7 SIP Inspection Class Map**

This section describes how to configure a Layer 7 class map for SIP application protocol inspection. It contains the following topics:
• Creating a Layer 7 SIP Inspection Class Map
• Adding a Layer 7 Class Map Description for SIP Inspection
• Defining the Called Party in the SIP To Header
• Defining the Calling Party in the SIP From Header
• Defining SIP Content Checks
• Defining the SIP Instant Messaging Subscriber
• Defining the Message Path Taken by SIP Messages
• Defining the SIP Request Methods
• Defining the SIP Party Registration Entities
• Defining SIP URI Checks

**Creating a Layer 7 SIP Inspection Class Map**

You can create a Layer 7 SIP inspection class map by using the `class-map type sip inspect` command in configuration mode.

The syntax of this command is as follows:

```
class-map type sip inspect [match-all | match-any] map_name
```
Chapter 3 Configuring Application Protocol Inspection

Configuring a Layer 7 SIP Inspection Policy

The keywords, arguments, and options are as follows:

- **match-all | match-any**—(Optional) Determines how the ACE performs the inspection of SIP traffic when multiple match criteria exist in a class map. The class map is considered a match if the **match** commands meet one of the following conditions:

  - **match-all**—(Default) Network traffic needs to satisfy all of the match criteria (implicit AND) to match the Layer 7 SIP inspection class map. The **match-all** keyword is applicable only for match statements of different SIP inspection types. For example, specifying a **match-all** condition for SIP URI, SIP header, and SIP content statements in the same class map is valid. However, specifying a **match-all** condition for multiple SIP headers with the same names or multiple URLs in the same class map is invalid.

  - **match-any**—Network traffic needs to satisfy only one of the match criteria (implicit OR) to match the Layer 7 SIP inspection class map. The **match-any** keyword is applicable only for match statements of the same Layer 7 SIP inspection type. For example, the ACE allows you to specify a **match-any** condition for SIP URI, SIP header, and SIP content statements in the same class map and allows you to specify a **match-any** condition for multiple URLs, multiple SIP headers, or multiple SIP content statements in the same class map as long as the statements are logical. For example, you could not have two **match uri sip length** statements in the same class map, but you could have one **match uri sip length** and one **match uri tel length** statement in one class map.

- **map_name**—Name assigned to the class map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

The CLI displays the class map SIP inspection configuration mode. To classify the SIP application inspection of traffic for evaluation by the ACE, include one or more of the following commands to configure the match criteria for the Layer 7 class map:

- **match called-party**—See the “Defining the Called Party in the SIP To Header” section.

- **match calling-party**—See the “Defining the Calling Party in the SIP From Header” section.

- **match content**—See the “Defining SIP Content Checks” section.

- **match im-subscriber**—See the “Defining the SIP Instant Messaging Subscriber” section.
Configuring a Layer 7 SIP Inspection Policy

- **match message-path**—See the “Defining the Message Path Taken by SIP Messages” section.
- **match request-method**—See the “Defining the SIP Request Methods” section.
- **match third-party-registration**—See the “Defining the SIP Party Registration Entities” section.
- **match uri**—See the “Defining SIP URI Checks” section.

You may include multiple `match` commands in the class map.

For example, to specify SIP_INSPECT_L7CLASS as the name of a class map and identify that all commands in the Layer 7 SIP application inspection class map must be satisfied for the ACE to indicate a match, enter:

```
host1/Admin(config)# class-map type sip inspect match-all SIP_INSPECT_L7CLASS
host1/Admin(config-cmap-sip-insp)# match calling-id .*ABC123
host1/Admin(config-cmap-sip-insp)# match im-subscriber JOHN_Q_PUBLIC
host1/Admin(config-cmap-sip-insp)# match content type sdp
```

To remove the SIP inspection class map from the ACE, enter:

```
host1/Admin(config)# no class-map type sip inspect match-any SIP_INSPECT_L7CLASS
```

### Adding a Layer 7 Class Map Description for SIP Inspection

You can add a Layer 7 Class map description by using the `description` command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

```
description text
```

The `text` argument is an unquoted text string with a maximum of 240 alphanumeric characters.

For example, enter:

```
host1/Admin(config-cmap-sip-insp)# description SIP inspection class map
```

To remove the description from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no description
```
Defining the Called Party in the SIP To Header

You can filter SIP traffic based on the called party (callee or destination) as specified in the URI of the SIP To header. The ACE does not include the display name or tag part of the field. To filter SIP traffic based on the called party, use the `match called-party` command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

```
[line_number] match called-party expression
```

The arguments and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `expression`—Regular expression that identifies the called party in the URI of the To header. Enter a regular expression from 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching. Expressions are stored in a header map in the form `header-name: expression`. Header expressions allow spaces if the spaces are escaped or quoted. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

**Note**

When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([ ]) to match these symbols (for example, enter `www[.]xyz[.]com` instead of `www.xyz.com`). You can also use a backslash (\) to escape a dot (.) or a question mark (?).

For example, to identify the called party in the SIP To header, enter:

```
host1/Config(config-cmap-sip-insp)# match called-party sip:some-user@somenetwork.com
```

To remove the `match` statement from the class map, enter:

```
host1/Config(config-cmap-sip-insp)# no match called-party sip:some-user@somenetwork.com
```
Defining the Calling Party in the SIP From Header

You can filter SIP traffic based on the calling party (caller or source) as specified in the URI of the SIP From header. The ACE does not include the display name or tag part of the field. To filter SIP traffic based on the calling party, use the \texttt{match calling-party} command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

\begin{verbatim}
[line_number] match calling-party expression
\end{verbatim}

The arguments are as follows:

- \textit{line_number}—(Optional) Argument that assists you in editing or deleting individual \texttt{match} commands. Enter an integer from 2 to 1024 as the line number. You can enter \texttt{no line_number} to delete long \texttt{match} commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- \textit{expression}—Regular expression that identifies the calling party in the URI of the SIP From header. Enter a regular expression from 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching. Expressions are stored in a header map in the form \texttt{header-name: expression}. Header expressions allow spaces if the spaces are escaped or quoted. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

\textbf{Note} When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([]) to match these symbols (for example, enter \texttt{www[.]xyz[.]com} instead of \texttt{www.xyz.com}). You can also use a backslash (\texttt{\}) to escape a dot (.) or a question mark (?).

For example, to identify the calling party in the SIP From header, enter:

\begin{verbatim}
host1/Admin(config-cmap-sip-insp)# match calling-party sip:this-user@thisnetwork.com;tag=745g8
\end{verbatim}

To remove the \texttt{match} statement from the class map, enter:

\begin{verbatim}
host1/Admin(config-cmap-sip-insp)# no match calling-party sip:this-user@thisnetwork.com;tag=745g8
\end{verbatim}
Defining SIP Content Checks

You can configure the ACE to perform SIP content checks based on the content length or the content type. By default, the ACE allows all content types. To define SIP content checks, use the `match content` command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

```
[line_number] match content {length gt number} | {type sdp | expression}
```

The keywords, arguments, and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **length**— Specifies the SIP message body length.

- **gt**—Specifies the greater than operator.

- **number**—Maximum size of a SIP message body that the ACE allows. Enter an integer from 0 to 65534 bytes. If the message body is greater than the configured value, the ACE performs the action that you configure in the policy map.

- **type**—Specifies a content type check.

- **sdp**—Specifies that the traffic must be of type Session Description Protocol (SDP) to match the class map.

- **expression**—Regular expression that identifies the content type in the SIP message body that is required to match the class map. Enter a regular expression from 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

Note: When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([[]]) to match these symbols (for example, enter `www[.]xyz[.]com` instead of `www.xyz.com`). You can also use a backslash (\) to escape a dot (.) or a question mark (?).
For example, to configure the ACE to drop SIP packets that have content with a length greater than 4000 bytes in length, enter:

```
host1/Admin(config)# class-map type sip inspect match-all
   SIP_INSP_CLASS
host1/Admin(config-cmap-sip-insp)# match content length gt 200

host1/Admin(config)# policy-map type inspect sip all-match
   SIP_INSP_POLICY
host1/Admin(config-pmap-ins-sip)# class SIP_INSP_CLASS
host1/Admin(config-pmap-ins-sip-c)# deny
```

To remove the `match` statement from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no match content length gt 200
```

### Defining the SIP Instant Messaging Subscriber

You can filter SIP traffic based on the IM subscriber by using the `match im-subscriber` command in class map SIP inspection configuration mode. The syntax of this command is as follows:

```
[line_number] match im-subscriber expression
```

The arguments and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `expression`—Regular expression that identifies the IM subscriber. Enter a regular expression from 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching. Expressions are stored in a header map in the form `header-name: expression`. Header expressions allow spaces, provided that the spaces are escaped or quoted. See Table 3-5 for a list of the supported characters that you can use in regular expressions.
Note  When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([ ]) to match these symbols (for example, enter www[.]xyz[.]com instead of www.xyz.com). You can also use a backslash (\) to escape a dot (.) or a question mark (?).

For example, enter:

```
host1/Admin(config-cmap-sip-insp)# match im-subscriber John_Q_Public
```

To remove the `match` statement from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no match im-subscriber John_Q_Public
```

**Defining the Message Path Taken by SIP Messages**

SIP inspection allows you to filter messages coming from or transiting through certain SIP proxy servers. The ACE maintains a list of unauthorized SIP proxy IP addresses or URIs in the form of regular expressions. The ACE checks this list against the VIA header field in each SIP packet. The default action is to drop SIP packets with VIA fields that match the regex list.

To filter SIP traffic based on the message path, use the `match message-path` command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

```
[line_number] match message-path expression
```

The arguments and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `expression`—Regular expression that identifies a SIP proxy server. Enter a regular expression from 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching. Expressions are stored in a
header map in the form header-name: expression. Header expressions allow spaces, provided that the spaces are escaped or quoted. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

**Note** When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([ ]) to match these symbols (for example, enter www[. ]xyz[. ]com instead of www.xyz.com). You can also use a backslash (\) to escape a dot (.) or a question mark (?).

For example, enter:

```
host1/Admin(config-cmap-sip-insp)# match message-path
192.168.12.3:5060
```

To remove the match statement from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no match message-path
192.168.12.3:5060
```

## Defining the SIP Request Methods

You can filter SIP traffic based on the request method by using the `match request-method` command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

```
[line_number] match request-method method_name
```

The arguments and options are as follows:

- **line_number**—(Optional) Argument that assists you in editing or deleting individual match commands. Enter an integer from 2 to 1024 as the line number. You can enter no line_number to delete long match commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- **method_name**—Supported SIP method using one of the following keywords:
  - ack
  - bye
  - cancel
Note Use the unknown keyword to permit or deny unknown or unsupported SIP methods.

For example, to filter SIP traffic based on the INVITE request method, enter:

```
host1/Admin(config-cmap-sip-insp)# match request-method invite
```

To remove the match statement from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no match request-method invite
```

**Defining the SIP Party Registration Entities**

SIP allows users to register other users on their behalf by sending REGISTER messages with different values in the From and To header fields. This process may pose a security threat if the REGISTER message is actually a DEREGISTER message. A malicious user could cause a Denial of Service (DoS) attack by deregistering all users on their behalf. To prevent this security threat, you can specify a list of privileged users who can register or unregister someone else on their behalf. The ACE maintains the list as a regex table. If you configure this policy, the ACE drops REGISTER messages with mismatched From and To headers and a From header value that does not match any of the privileged user IDs.
To filter SIP traffic based on third-party registrations or deregistrations, use the `match third-party-registration` command in class map SIP inspection configuration mode. The syntax of this command is as follows:

```
[line_number] match third-party-registration expression
```

The arguments and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `expression`—Privileged user that is authorized for third-party registrations. Enter a regular expression from 1 to 255 alphanumeric characters. The ACE supports the use of regular expressions for matching. Expressions are stored in a header map in the form `header-name: expression`. Header expressions allow spaces, provided that the spaces are escaped or quoted. See Table 3-5 for a list of the supported characters that you can use in regular expressions.

**Note**

When matching data strings, note that the period (.) and question mark (?) characters do not have a literal meaning in regular expressions. Use brackets ([]) to match these symbols (for example, enter `www[.]xyz[.]com` instead of `www.xyz.com`). You can also use a backslash (\) to escape a dot (.) or a question mark (?).

For example, to filter SIP traffic based on SIP registrations or deregistrations, enter:

```
host1/Admin(config-cmap-sip-insp)# match third-party-registration USER1
```

To remove the `match` statement from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no match third-party-registration USER1
```
Defining SIP URI Checks

You can configure the ACE to validate the length of SIP URIs or Tel URIs. A SIP URI is a user identifier that a calling party (source) uses to contact the called party (destination). A Tel URI is a telephone number that identifies the endpoint of a SIP connection. For more information about SIP URIs and Tel URIs, see RFC 2534 and RFC 3966.

To filter SIP traffic based on URIs, use the `match uri` command in class map SIP inspection configuration mode.

The syntax of this command is as follows:

```
[line_number] match uri {sip | tel} length gt value
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 1024 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.
- `sip`—Specifies the ACE validates the length of a SIP URI.
- `tel`—Specifications the ACE validates the length of a Tel URI.
- `length`—Specifies the length of the SIP or Tel URI.
- `gt`—Greater than operator.
- `value`—Maximum value for the length of the SIP URI or Tel URI in bytes. Enter an integer from 0 to 254 bytes.

For example, enter:

```
host1/Admin(config-cmap-sip-insp)# match uri sip length gt 100
```

To remove the `match` statement from the class map, enter:

```
host1/Admin(config-cmap-sip-insp)# no match uri sip length gt 100
```

Configuring a Layer 7 SIP Inspection Policy Map

This section describes how to configure a Layer 7 SIP inspection policy map. The Layer 7 policy map configures the applicable SIP inspection actions executed on the network traffic that matches the classifications defined in a class map. You
then associate the completed Layer 7 SIP inspection policy with a Layer 3 and Layer 4 policy map to activate the operation on a VLAN interface (see the “Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions” section).

This section contains the following topics:

- Configuring a Layer 7 SIP Inspection Policy Map
- Adding a Layer 7 SIP Inspection Policy Map Description
- Including Inline Match Statements in a Layer 7 SIP Inspection Policy Map
- Associating the Layer 7 SIP Inspection Class Map with the Policy Map
- Specifying the Layer 7 SIP Inspection Policy Map Actions

### Creating a Layer 7 SIP Policy Map

You can create a Layer 7 SIP policy map by using the `policy-map type inspect sip` command in configuration mode.

The syntax of this command is as follows:

```
policy-map type inspect sip all-match map_name
```

The keywords and arguments are as follows:

- `sip all-match`—Specifies the policy map that initiates the inspection of the SIP protocol packets by the ACE. The ACE attempts to match all specified conditions against the matching classification and executes the actions of all matching classes until it encounters a deny for a match request.

- `map_name`—Name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to create a Layer 7 SIP inspection policy map, enter:

```
host1/Admin(config)# policy-map type inspect sip all-match SIP_INSPECT_L7POLICY
host1/Admin(config-pmap-ins-sip)#
```

To remove the SIP inspection policy map from the configuration, enter:

```
host1/Admin(config)# no policy-map type inspect sip all-match SIP_INSPECT_L7POLICY
```
Adding a Layer 7 SIP Inspection Policy Map Description

You can configure a description for the Layer 7 SIP inspection policy map by using the `description` command in policy map inspection SIP configuration mode.

The syntax of this command is as follows:

```
description
```

For example, to add a description for a Layer 7 SIP inspection policy map, enter:

```
host1/Admin(config-pmap-ins-sip)# description layer 7 sip inspection policy
```

To remove the description from the policy map, enter:

```
host1/Admin(config-pmap-ins-sip)# no description
```

Including Inline Match Statements in a Layer 7 SIP Inspection Policy Map

You can include a single inline match criteria in the policy map without specifying a traffic class by using an applicable Layer 7 `match` command. The inline Layer 7 policy map `match` commands function the same as the Layer 7 class map `match` commands. However, when you use an inline `match` command, you can specify an action for only a single match statement in the Layer 7 policy map.

```
match name match_statement [insert-before map_name]
```

The keywords, arguments, and options are as follows:

- **name**—Name assigned to the inline `match` command. Enter an unquoted text string with no spaces. The length of the inline match statement name plus the length of the policy map name with which it is associated cannot exceed a total maximum of 64 alphanumeric characters. For example, if the policy map name is L7_POLICY (nine characters), an inline match statement name under this policy cannot exceed 55 alphanumeric characters (64 - 9 = 55).
match_statement—Inline match criteria to be used by the policy map. See the
details on the match commands associated with the Layer 7 SIP inspection
class map.

insert-before map_name—(Optional) Places the inline match command
ahead of an existing class map in the policy map configuration.

The syntax for the Layer 7 SIP inspection policy map inline match commands is
as follows:

match name called-party expression

match name calling-party expression

match name content {length gt number} | {type sdp | expression}

match name im-subscriber expression

match name message-path expression

match name request-method method_name

match name third-party-registration expression

match name uri {sip | tel} length gt value

See the “Configuring a Layer 7 SIP Inspection Class Map” section for details
about the inline match commands.

For example, to add an inline match command to a Layer 7 SIP inspection policy
map, enter:

host/Admin(config-pmap-ins-sip)# match sip_match called-party abc123.*

to remove the inline match command from the policy map, enter:

host/Admin(config-pmap-ins-sip)# no match sip_match called-party abc123.*

Associating the Layer 7 SIP Inspection Class Map with the Policy Map

You can associate the Layer 7 SIP inspection class map with the Layer 7 SIP
inspection policy map by using the class command in policy map inspection SIP
configuration mode.
The syntax of this command is as follows:

\texttt{class \textit{map\_name}}

The \textit{map\_name} argument is the identifier of an existing Layer 7 SIP inspection class map. Enter the name as an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to associate a Layer 7 SIP inspection class map with a Layer 7 SIP inspection policy map, enter:

```
host/Admin(config-pmap-ins-sip)# class SIP_INSPECT_L7CLASS
```

To dissociate the class map from the policy map, enter:

```
host/Admin(config-pmap-ins-sip)# no class SIP_INSPECT_L7CLASS
```

### Specifying the Layer 7 SIP Inspection Policy Map Actions

By default, the ACE allows all SIP packets to pass. To explicitly deny specific SIP commands, use one of the following commands as the action if the specified SIP traffic matches the classification. You apply the specified action against the single inline \texttt{match} command in policy map SIP inspection match configuration mode or against the specified class map in policy map SIP inspection class configuration mode.

\texttt{\{[drop \mid \texttt{permit} \mid \texttt{reset}] [log]\} | log}

The keywords and options are as follows:

- \texttt{drop}—Drops the SIP packet that matches the class map or the single inline \texttt{match} command.
- \texttt{permit}—(Default) Allows SIP traffic that matches the class map or the single inline \texttt{match} command to pass through the ACE.
- \texttt{reset}—Denies SIP traffic that matches the class map or the single inline \texttt{match} command and resets the connection using the TCP RESET message.
- \texttt{log}—Generates a log message for traffic that matches the class map or the single inline \texttt{match} command.
For example, to specify an action in the Layer 7 SIP inspection policy map for traffic that matches the associated Layer 7 SIP inspection class map, enter:

```
host1/Admin(config)# policy-map type inspect sip first-match SIP_INSPECT_L7POLICY
host1/Admin(config-pmap-ins-sip)# class SIP_INSPECT_L7CLASS
host1/Admin(config-pmap-ins-sip-c)# drop
```

To specify an action in a Layer 7 SIP inspection policy map for traffic that matches a single inline `match` command, enter:

```
host1/Admin(config)# policy-map type inspect sip first-match SIP_INSPECT_L7POLICY
host1/Admin(config-pmap-ins-sip)# match SIP_MATCH calling-party 123abc.*
host1/Admin(config-pmap-ins-sip-m)# drop
```

To disable an action in the Layer 7 SIP inspection policy map, enter:

```
host1/Admin(config-pmap-ins-sip-m)# no drop
```

## Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy

This section describes how to create a Layer 3 and Layer 4 class map and policy map to classify network traffic that passes through the ACE and to perform applicable application protocol inspection actions to that traffic. The Layer 3 and Layer 4 traffic policy defines the Layer 3 and Layer 4 HTTP deep packet inspection, FTP command inspection, or application protocol inspection policy actions. Application inspection involves the examination of protocols such as DNS, FTP, HTTP, ICMP, ILS, RTSP, SCCP, and SIP to verify the protocol behavior and identify unwanted or malicious traffic that passes through the ACE.

- **Configuration Guidelines for Inspection Traffic Policies**
- **Configuring a Layer 3 and Layer 4 Class Map**
- **Configuring a Layer 3 and Layer 4 Policy Map**
Configuration Guidelines for Inspection Traffic Policies

Because the version A4(1.0) ACE software has strict error checks for application protocol inspection configurations, be sure that your inspection configurations meet the guidelines in this section. The error checking process in the software denies misconfigurations in inspection classifications (class maps) and displays appropriate error messages. If such misconfigurations exist in your startup- or running-configuration file before you load the software, the standby ACE in a redundant configuration may boot up to the STANDBY_COLD state. For information about redundancy states, see the *Cisco Application Control Engine Module Administration Guide*.

If the class map for the inspection traffic is generic (match . . . any or class-default is configured) so that noninspection traffic is also matched, the ACE displays an error message and does not accept the inspection configuration. For example:

```
switch/Admin(config)# class-map match-all TCP_ANY
switch/Admin(config-cmap)# match port tcp any

switch/Admin(config)# policy-map multi-match FTP_POLICY
switch/Admin(config-pmap)# class TCP_ANY
switch/Admin(config-pmap-c)# inspect ftp
Error: This class doesn't have tcp protocol and a specific port
```

The following examples show some of the generic class-map match statements and an ACL that are not allowed in inspection configurations:

- match port tcp any
- match port udp any
- match port tcp range 0 65535
- match port udp range 0 65535
- match virtual-address 192.168.12.15 255.255.255.0 any
- match virtual-address 192.168.12.15 255.255.255.0 tcp any
- access-list acl1 line 10 extended permit ip any any

For application protocol inspection, the class map must have a specific protocol (related to the inspection type) configured and a specific port or range of port numbers.
For HTTP, FTP, RTSP, Skinny, and ILS protocol inspection, the class map must have TCP as the configured protocol and a specific port or range of ports. For example, enter the following commands:

```
host1/Admin(config)# class-map match-all L4_CLASS
host1/Admin(config-cmap)# match port tcp eq www
```

For SIP protocol inspection, the class map must have TCP or UDP as the configured protocol and a specific port or range of ports. For example, enter the following commands:

```
host1/Admin(config)# class-map match-all L4_CLASS
host1/Admin(config-cmap)# match port tcp eq 124
```

or

```
host1/Admin(config-cmap)# match port udp eq 135
```

For DNS inspection, the class map must have UDP as the configured protocol and a specific port or range of ports. For example, enter the following commands:

```
host1/Admin(config)# class-map match-all L4_CLASS
host1/Admin(config-cmap)# match port udp eq domain
```

For ICMP protocol inspection, the class map must have ICMP as the configured protocol. For example, enter the following commands:

```
host1/Admin(config)# access-list ACL1 extended permit icmp 192.168.12.15 255.255.255.0 192.168.16.25 255.255.255.0 echo

host1/Admin(config)# class-map match-all L4_CLASS
host1/Admin(config-cmap)# match access-list ACL1
```

## Configuring a Layer 3 and Layer 4 Class Map

You can create a Layer 3 and Layer 4 class map to classify network traffic that passes through the ACE to perform an applicable application protocol inspection policy by using the `class-map` command in configuration mode.

You can have multiple `match` commands in a single class map to specify the matching criteria. For example, you can configure class maps to define multiple access group or port commands in a group that you then associate with an
application protocol inspection policy. The **match-all** and **match-any** keywords determine how the ACE evaluates the operations for multiple match statements when multiple match criteria exist in a class map.

The syntax of this command is as follows:

```
class-map [match-all | match-any] map_name
```

The keywords, arguments, and options are as follows:

- **match-all | match-any**—(Optional) Determines how the ACE evaluates Layer 3 and Layer 4 network traffic when multiple match criteria exist in a class map. The class map is considered a match if the **match** commands meet one of the following conditions.
  - **match-all**—(Default) All of the match criteria listed in the class map are satisfied to match the network traffic class in the class map, typically, **match** commands of different types.
  - **match-any**—Only one of the match criteria listed in the class map is satisfied to match the network traffic class in the class map, typically, **match** commands of the same type.

- **map_name**—Name assigned to the class map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

The CLI displays the class map configuration mode. To classify network traffic that passes through the ACE for application protocol inspection, include one or more of the following commands to configure the match criteria for the class map:

- **description**—See the “Adding a Layer 3 and Layer 4 Class Map Description” section.
- **match access-list**—See the “Defining Access-List Match Criteria” section.
- **match port**—See the “Defining TCP/UDP Port Number or Port Range Match Criteria” section.

Follow these guidelines when creating a class map to define a Layer 3 and Layer 4 match classification:

- You may combine multiple **match access-list** and **match port** commands in a class map.
- The matched traffic depends on the individual **inspect** command specified in the policy map. See Table 3-1 for a summary of the application inspection protocols supported by the ACE with the IP protocol and port.
For example, to define a Layer 3 and Layer 4 class map, enter:

```
host1/Admin(config)# class-map match-all DNS_INSPECT_L4CLASS
host1/Admin(config-cmap)# description DNS application protocol inspection of incoming traffic
host1/Admin(config-cmap)# match port udp eq domain
```

To remove a Layer 3 and Layer 4 network traffic class map from the ACE, enter:

```
host1/Admin(config)# no class-map match-all DNS_INSPECT_L4CLASS
```

This section contains the following topics:

- Adding a Layer 3 and Layer 4 Class Map Description
- Defining Access-List Match Criteria
- Defining TCP/UDP Port Number or Port Range Match Criteria

### Adding a Layer 3 and Layer 4 Class Map Description

You can use the `description` command to provide a brief summary of the Layer 3 and Layer 4 class map. You must access the class map configuration mode to specify the `description` command.

The syntax of this command is as follows:

```
description text
```

The `text` argument is an unquoted text string with a maximum of 240 alphanumeric characters.

For example, to specify a description that the class map is to perform DNS application protocol inspection, enter:

```
host1/Admin(config)# class-map DNS_INSPECT_L4CLASS
host1/Admin(config-cmap)# description DNS application protocol inspection of incoming traffic
```

To remove the description from the class map, enter:

```
host1/Admin(config-cmap)# no description
```
Defining Access-List Match Criteria

You can use the `match access-list` command to configure the class map to filter Layer 3 and Layer 4 network traffic on a per-flow basis by using a predefined access control list. When a packet matches an entry in an access list, and if it is a `permit` entry, the ACE allows the matching result. If it is a `deny` entry, the ACE blocks the matching result. See Chapter 1, Configuring Security Access Control Lists, for details about the creating access control lists in the ACE.

For application protocol inspection, an access list must specify explicitly the IP addresses and ports in the ACL entries. Otherwise, the ACE displays an error message.

You must access the class map configuration mode to specify the `match access-list` command.

The syntax of this command is as follows:

```
[line_number] match access-list identifier
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 255 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `identifier`—Previously created access list identifier. Enter an unquoted text string with a maximum of 64 characters.

You can enter multiple `match access-list` commands within a single class map. You may combine multiple `match access-list` and `match port` commands in a class map.

For example, to specify that the class map is to match on access control list INBOUND_ACL1, enter:

```
host1/Admin(config)# class-map match-any DNS_INSPECT_L4CLASS
host1/Admin(config-cmap)# match access-list INBOUND_ACL1
```

To clear the access control list match criteria from the class map, enter:

```
host1/Admin(config-cmap)# no match access-list inboundacl1
```
Defining TCP/UDP Port Number or Port Range Match Criteria

You can use the `match port` command to specify a TCP or UDP port number or port range as the Layer 3 and Layer 4 network traffic matching criteria.

You must access the class map configuration mode to specify the `match port` command.

The syntax of this command is as follows:

```
[line_number] match port {tcp | udp} {any | eq {port_number} | range port1 port2}
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. Enter an integer from 2 to 255 as the line number. You can enter `no line_number` to delete long `match` commands instead of entering the entire line. The line numbers do not dictate a priority or sequence for the match statements.

- `tcp | udp`—Specifies the protocol, TCP or UDP, as follows:
  - `any`—Specifies the wildcard value for the TCP or UDP port number. If you use `any` in place of either the `eq` or `range` values, packets from any incoming port will match.
  - `eq port_number`—Specifies that the TCP or UDP port number must match the specified value. Enter an integer from 0 to 65535. A value of 0 instructs the ACE to include all ports. Alternatively, you can enter the name of a well-known TCP port as listed in Table 3-7 or a well-known UDP port as listed in Table 3-8.
  - `range port1 port2`—Specifies a port range to use for the TCP or UDP port. Valid port ranges are from 0 to 65535. A value of 0 instructs the ACE to match all ports.

<table>
<thead>
<tr>
<th>Table 3-7 Well-Known TCP Ports and Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
</tr>
<tr>
<td>domain</td>
</tr>
<tr>
<td>ftp</td>
</tr>
<tr>
<td>ftp-data</td>
</tr>
<tr>
<td>http</td>
</tr>
</tbody>
</table>
### Table 3-7  **Well-Known TCP Ports and Keywords (continued)**

<table>
<thead>
<tr>
<th>Port</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>53</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>https</td>
<td>443</td>
<td>HTTP over SSL protocol</td>
</tr>
<tr>
<td>irc</td>
<td>194</td>
<td>Internet Relay Chat protocol</td>
</tr>
<tr>
<td>matip-a</td>
<td>350</td>
<td>Matip Type A protocol</td>
</tr>
<tr>
<td>nntp</td>
<td>119</td>
<td>Network News Transport Protocol</td>
</tr>
<tr>
<td>pop2</td>
<td>109</td>
<td>Post Office Protocol v2</td>
</tr>
<tr>
<td>pop3</td>
<td>110</td>
<td>Post Office Protocol v3</td>
</tr>
<tr>
<td>rtsp</td>
<td>554</td>
<td>Real Time Streaming Protocol</td>
</tr>
<tr>
<td>sip</td>
<td>5060</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>skinny</td>
<td>2000</td>
<td>Cisco Skinny Client Control Protocol (SCCP)</td>
</tr>
<tr>
<td>smtp</td>
<td>25</td>
<td>Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>sunrpc</td>
<td>111</td>
<td>Sun Remote Procedure Call (RPC)</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>Telnet protocol</td>
</tr>
<tr>
<td>www</td>
<td>80</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>xot</td>
<td>1998</td>
<td>X25 over TCP</td>
</tr>
</tbody>
</table>

### Table 3-8  **Well-Known UDP Port Numbers and Keywords**

<table>
<thead>
<tr>
<th>Key Word</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>53</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>sip</td>
<td>5060</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>wsp</td>
<td>9200</td>
<td>Connectionless Wireless Session Protocol (WSP)</td>
</tr>
<tr>
<td>wsp-wtls</td>
<td>9202</td>
<td>Secure Connectionless WSP</td>
</tr>
<tr>
<td>wsp-wtp</td>
<td>9201</td>
<td>Connection-based WSP</td>
</tr>
<tr>
<td>wsp-wtp-wtls</td>
<td>9203</td>
<td>Secure Connection-based WSP</td>
</tr>
</tbody>
</table>
You can enter multiple `match port` commands within a single class map. You may combine multiple `match access-list` and `match port` commands in a class map.

For example, to specify that the class map is to match on TCP port number 23 (Telnet client), enter:

```
host1/Admin(config)# class-map DNS_INSPECT_L4CLASS
host1/Admin(config-cmap)# match port tcp eq 23
```

To clear the TCP or UDP port number match criteria from the class map, enter:

```
host1/Admin(config-cmap)# no match port tcp eq 23
```

### Configuring a Layer 3 and Layer 4 Policy Map

This section describes how to configure a Layer 3 and Layer 4 policy that defines an HTTP deep packet inspection, FTP command inspection, or application protocol inspection traffic policy.

This section contains the following topics:

- Creating a Layer 3 and Layer 4 Policy Map
- Adding a Layer 3 and Layer 4 Policy Map Description
- Specifying a Layer 3 and Layer 4 Traffic Class with the Traffic Policy
- Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions

### Creating a Layer 3 and Layer 4 Policy Map

You can use the `policy-map multi-match` configuration command to configure a Layer 3 and Layer 4 policy map that defines the application inspection policies. The ACE attempts to match multiple classes within the Layer 3 and Layer 4 policy map but can match only one class within each of the sets of traffic classes. If a classification matches more than one class map, then the ACE executes all of the corresponding actions. However, for a specific feature, the ACE executes only the first matching classification action.

The syntax of this command is as follows:

```
policy-map multi-match map_name
```
Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy

The `map_name` argument is the name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to create a Layer 3 and Layer 4 network traffic policy map, enter:

```
host1/Admin(config)# policy-map multi-match HTTP_INSPECT_L4POLICY
```

The CLI displays the policy map configuration mode.

To remove a Layer 3 and Layer 4 policy map from the ACE, enter:

```
host1/Admin(config)# no policy-map multi-match HTTP_INSPECT_L4POLICY
```

Adding a Layer 3 and Layer 4 Policy Map Description

You can use the `description` command to provide a brief summary of the Layer 3 and Layer 4 policy map. You must access the policy map configuration mode to specify the `description` command.

The syntax of this command is as follows:

```
description text
```

The `text` argument is an unquoted text string with a maximum of 240 alphanumeric characters.

For example, to specify a description that the policy map is to perform DNS application protocol inspection, enter:

```
host1/Admin(config-pmap)# description DNS application protocol inspection of incoming traffic
```

To remove the description from the policy map, enter:

```
host1/Admin(config-pmap)# no description
```

Specifying a Layer 3 and Layer 4 Traffic Class with the Traffic Policy

You can specify a traffic class created with the `class-map` command to associate network traffic with the traffic policy by using the `class` command.

The syntax of this command is as follows:

```
class map_name
```
The `map_name` argument is the name of a previously defined traffic class, configured with the `class-map` command, to associate traffic to the traffic policy. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

The CLI displays the policy map class configuration mode.

For example, to specify an existing class map within the Layer 3 and Layer 4 policy map, enter:

```bash
host1/Admin(config-pmap)# class HTTP_INSPECT_L4CLASS
host1/Admin(config-pmap-c)#
```

To remove a class map from a Layer 3 and Layer 4 policy map, enter:

```bash
host1/Admin(config-pmap)# no class HTTP_INSPECT_L4CLASS
```

To manually insert a class map ahead of a previously specified class map, use the `insert-before` command. The ACE does not save sequence reordering through the `insert-before` command as part of the configuration.

The syntax of this command is as follows:

```
   class map_name1 insert-before map_name2
```

The keywords and arguments are as follows:

- **class map_name1**—Specifies the name of a previously defined traffic class configured with the `class-map` command. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

- **insert-before map_name2**—Places the current class map ahead of an existing class map as specified by the `map_name2` argument. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to use the `insert-before` command to define the sequential order of two class maps in the policy map, enter:

```bash
host1/Admin(config-pmap-c)# 5 class FTP_INSPECT_L4CLASS insert-before HTTP_INSPECT_L4CLASS
```

To specify the `class-default` class map for the Layer 3 and Layer 4 traffic policy, use the `class class-default` command. All network traffic that fails to meet the other matching criteria in the named class map belongs to the default traffic class. If none of the specified classifications match, the ACE then matches the action specified under the `class class-default` command. The `class-default` class map has an implicit `match any` statement in it so that it matches all traffic.
For example, to use the `class class-default` command, enter:

```
host1/Admin(config-pmap)# class class-default
host1/Admin(config-pmap-c)#
```

The CLI displays the policy map class configuration mode.

### Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions

You can use the `inspect` command in policy map class configuration mode to define the Layer 3 and Layer 4 HTTP deep packet inspection, FTP command inspection, or application protocol inspection policy actions. Application inspection involves the examination of protocols such as DNS, FTP, HTTP, ICMP, ILS, RTSP, SCCP, and SIP to verify the protocol behavior and identify unwanted or malicious traffic that passes through the ACE.

If you intend to perform the optional Layer 7 application inspection of network traffic, first create a Layer 7 policy as follows:

- To perform the deep packet inspection of Layer 7 HTTP application traffic by the ACE, first create a Layer 7 policy using the `policy-map type inspect http` command (see the “Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map” section). You nest the Layer 7 HTTP inspection policy by using the Layer 3 and Layer 4 `inspect http` command.

- To perform the request inspection of FTP commands, first create a Layer 7 policy by using the `policy-map type inspect ftp` command (see the “Configuring a Layer 7 FTP Command Inspection Policy Map” section). You nest the Layer 7 FTP inspection policy by using the Layer 3 and Layer 4 `inspect ftp` command.

You associate the Layer 7 policy map within the appropriate Layer 3 and Layer 4 policy map to provide an entry point for the traffic classification. Layer 7 policy maps are considered to be child policies and can only be associated within a Layer 3 and Layer 4 policy map. Only a Layer 3 and Layer 4 policy map can be applied to a VLAN interface or applied globally to all VLAN interfaces in the same context; a Layer 7 policy map cannot be directly applied on an interface.

---

**Note**

If you do not specify a Layer 7 HTTP or FTP policy map, the ACE performs a general set of Layer 3 and Layer 4 HTTP or FTP protocol fixup actions. For example, the ACE performs strict HTTP. For a complete list of the Layer 4 HTTP inspection operations and checks that the ACE performs, see the “HTTP Deep Packet Inspection” section.
The syntax of this command is as follows:

- `inspect dns [maximum-length bytes]`
- `inspect ftp [strict policy name1 | sec-param conn_parammap_name1]`
- `inspect http [policy name4 | url-logging]`
- `inspect icmp [error]`
- `inspect ils`
- `inspect rtsp [sec-param conn_parammap_name3]`
- `inspect sip [sec-param conn_parammap_name4] [policy name5]`
- `inspect skinny [sec-param conn_parammap_name5] [policy name6]`

The keywords, arguments, and options are as follows:

- **dns**—Enables DNS query inspection. DNS requires an application inspection so that DNS queries will not be subject to the generic UDP handling based on activity timeouts. Instead, the UDP connections associated with DNS queries and responses are torn down as soon as a reply to a DNS query has been received. The ACE performs the reassembly of DNS packets to verify that the packet length is less than the configured maximum length.

- **maximum-length bytes**—(Optional) Sets the maximum length of a DNS reply. Valid entries are from 512 to 65536 bytes. There is no default. If you do not set a maximum-length value, the ACE does not check the size of the reply from the DNS server.

- **ftp**—Enables FTP inspection. The ACE inspects FTP packets, translates addresses and ports embedded in the payload, and opens up a secondary channel for data.

  - **strict**—(Optional) Checks for protocol RFC compliance and prevents web browsers from sending embedded commands in FTP requests. The `strict` keyword prevents an FTP client from determining valid usernames that are supported on an FTP server. When an FTP server replies to the USER command, the ACE intercepts the 530 reply code from the FTP server and replaces it with the 331 reply code. Specifying an FTP inspection policy allows selective command filtering and also prevent the display of the FTP server system type to the FTP client. The ACE intercepts the FTP server 215 reply code and message to the SYST command and replaces the text following the reply code with asterisks.
- `sec-param conn_parammap_name1`—(Optional) Specifies the name of a previously created connection parameter map used to define parameters for FTP inspection.

**Note** If you do not specify a Layer 7 policy map, the ACE performs a general set of Layer 3 and Layer 4 FTP protocol fixup actions.

- `policy name1`—(Optional) Specifies the name assigned to a previously created Layer 7 FTP command inspection policy map to implement the inspection of Layer 7 FTP commands by the ACE. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. Use the `inspect ftp` command in policy map class configuration mode to define the FTP command request inspection policy.

- `http`—Enables enhanced HTTP inspection on HTTP traffic. By default, the ACE allows all request methods.

- `policy name4`—(Optional) Specifies the name assigned to a previously created Layer 7 HTTP application inspection policy map to implement the deep packet inspection of Layer 7 HTTP application traffic by the ACE. The inspection checks are based on configured parameters in an existing Layer 7 policy map and internal RFC compliance checks performed by the ACE. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

**Note** If you do not specify a Layer 7 policy map, the ACE performs a general set of Layer 3 and Layer 4 HTTP fixup actions and internal RFC compliance checks. For details, see the “HTTP Deep Packet Inspection” section.

- `url-logging`—(Optional) Enables the monitoring of Layer 3 and Layer 4 traffic. This function logs every URL request that is sent in the specified class of traffic, including the source or destination IP address and the URL that is accessed.

- `icmp`—Enables ICMP payload inspection. ICMP inspection allows ICMP traffic to have a “session” so it can be inspected similarly to TCP and UDP traffic.
- **error**—(Optional) Performs a NAT of ICMP error messages. The ACE creates translation sessions for intermediate or endpoint nodes that send ICMP error messages based on the NAT configuration. The ACE overwrites the packet with the translated IP addresses.

- **ils**— Enables Internet Locator Service (ILS) protocol inspection.

- **rtsp**— Enables RTSP packet inspection. RTSP is used by RealAudio, RealNetworks, Apple QuickTime 4, RealPlayer, and Cisco IP/TV connections. The ACE monitors Setup and Response (200 OK) messages in the control channel established using TCP port 554 (no UDP support).

- **sec-param conn_parammap_name3**—(Optional) Specifies the name of a previously created connection parameter map used to define parameters for RTSP inspection.

- **sip**— Enables Session Initiation Protocol (SIP) inspection. SIP is used for call handling sessions and instant messaging. The ACE inspects signaling messages for media connection addresses, media ports, and embryonic connections. The ACE also performs Network Address Translations (NATs) on IP addresses that are embedded in the user-data portion of the packet.
  - **sec-param conn_parammap_name4**—(Optional) Specifies the name of a previously created connection parameter map used to define parameters for SIP inspection.
  - **policy name5**—(Optional) Specifies the name of a previously created Layer 7 SIP application inspection policy map to implement packet inspection of Layer 7 SIP application traffic by the ACE. The inspection checks are based on configured parameters in an existing Layer 7 policy map and internal RFC compliance checks performed by the ACE. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

  **Note** If you do not specify a Layer 7 policy map, the ACE performs a general set of Layer 3 and Layer 4 SIP protocol fixup actions and internal RFC compliance checks.

- **skinny**— Enables Cisco Skinny Client Control Protocol (SCCP) inspection. The SCCP is a Cisco proprietary protocol that is used between Cisco CallManager and Cisco VoIP phones. The ACE performs a NAT on embedded IP addresses and port numbers in SCCP packet data.
– **policy name**6—(Optional) Specifies the name of a previously created deep packet inspection of Layer 7 SCCP application traffic by the ACE. The inspection checks are based on configured parameters in an existing Layer 7 policy map and internal RFC compliance checks performed by the ACE. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

**Note**

If you do not specify a Layer 7 policy map, the ACE performs a general set of Layer 3 and Layer 4 SCCP protocol fixup actions and internal RFC compliance checks.

– **sec-param conn_parammap_name**5—(Optional) Specifies the name of a previously created connection parameter map used to define parameters for SCCP inspection.

For example, to specify the **inspect http** command as an action for an HTTP application protocol inspection policy map, enter:

```
host1/Admin(config)# policy-map multi-match HTTP_INSPECT_L4POLICY
host1/Admin(config-pmap)# class HTTP_INSPECT_L4CLASS
host1/Admin(config-pmap-c)# inspect http policy
HTTP_DEEPINSPECT_L7POLICY
```

For example, to specify the **inspect dns** command as an action for a DNS application protocol inspection policy map, enter:

```
host1/Admin(config)# policy-map multi-match DNS_INSPECT_L4POLICY
host1/Admin(config-pmap)# class DNS_INSPECT_L4CLASS
host1/Admin(config-pmap-c)# inspect dns 1000
```

For example, to specify the **inspect ftp** command as an action for an FTP command inspection policy map, enter:

```
host1/Admin(config)# policy-map multi-match FTP_INSPECT_L4POLICY
host1/Admin(config-pmap)# class FTP_INSPECT_L7CLASS
host1/Admin(config-pmap-c)# inspect ftp strict policy
FTP_INSPECT_L7POLICY
host1/Admin(config-pmap-c)# exit
host1/Admin(config)#
```

To disable an application protocol inspection action from a policy map, enter:

```
host1/Admin(config-pmap-c)# no inspect dns 1000
```
Configuring a DNS Parameter Map

You can use a parameter map to apply actions to a Layer 3 and Layer 4 DNS inspection policy map. You reference this parameter map in the `appl-parameter` command in policy map class configuration mode. See the “Associating a DNS Parameter Map with a Layer 3 and Layer 4 Policy Map” section.

You can configure DNS actions for DNS packet inspection by using the `parameter-map type dns` command in configuration mode. The syntax of this command is as follows:

```
parameter-map type dns name
```

The `name` argument is the identifier assigned to the parameter map. Enter an unquoted text string with no spaces and a maximum of 32 alphanumeric characters.

For example, to create a parameter map called DNS_PARAMMAP, enter the following command:

```
host1/Admin(config)# parameter-map type dns DNS_PARAMMAP
host1/Admin(config-parammap-dns)#
```

To remove a DNS parameter map from the configuration, enter the following command:

```
host1/Admin(config)# no parameter-map type dns DNS_PARAMMAP
```

This section contains the following subsections:

- Defining a Description of the DNS Parameter Map
- Configuring a DNS Query Timeout
- Associating a DNS Parameter Map with a Layer 3 and Layer 4 Policy Map

Defining a Description of the DNS Parameter Map

You can provide a brief summary of the DNS parameter map by using the `description` command in DNS parameter map configuration mode. The syntax of this command is as follows:

```
description text
```
For the text argument, enter an unquoted text string with a maximum of 240 alphanumeric characters including spaces.

For example, to specify a description of a DNS parameter map, enter the following command:

```
host1/Admin#parameter-map type dns DNS_PARAMMAP
host1/Admin(config-parammap-dns)# description dns parameter map
```

To remove the description from the DNS parameter map, enter:

```
host1/Admin(config-parammap-dns)# no description
```

### Configuring a DNS Query Timeout

When you enable DNS inspection using the inspect dns command as a Layer 4 policy-map action (see the “Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions” section), the ACE stores DNS queries that it receives from clients in a hash table. When it receives a response from the DNS server, the ACE forwards the server response to the client if it finds a matching query in the table and then deletes the entry in the table. Queries, for which the ACE does not receive a response, remain in the table until they time out. The ACE may not receive an answer for a DNS query because the server is down, the query was spoofed, and so on.

If the underlying UDP connection times out, the ACE removes all DNS query hash entries using that UDP connection in 2 seconds. You can configure the UDP inactivity timeout using a connection parameter map. For details, see Chapter 4, Configuring TCP/IP Normalization and IP Reassembly Parameters.

If the ACE continues to receive DNS queries on the same UDP connection, the UDP connection does not time out. In this case, the queries without answers will time out in 10 seconds. To change this time-out value, use the timeout query command in DNS parameter map configuration mode. The syntax of this command is as follows:

```
timeout query number
```

The number argument specifies the length of time in seconds that the ACE keeps the query entries without answers in the hash table before timing them out. Enter an integer from 2 to 120 seconds. The default is 10 seconds.
For example, to time out DNS queries with no responses after 20 seconds, enter the following commands:

```
host1/Admin(config)# parameter-map type dns DNS_PARAMMAP
host1/Admin(config-parammap-dns)# timeout query 20
```

To reset the query timeout value to the default of 10 seconds, enter the following commands:

```
host1/Admin(config)# parameter-map type dns DNS_PARAMMAP
host1/Admin(config-parammap-dns)# no timeout query 20
```

### Associating a DNS Parameter Map with a Layer 3 and Layer 4 Policy Map

You can associate a DNS parameter map with a Layer 3 and Layer 4 policy map by using the `appl-parameter dns advanced-options` command in policy map class configuration mode.

The syntax of this command is as follows:

```
appl-parameter dns advanced-options name
```

The `name` argument is the name of an existing DNS parameter map. Parameter maps aggregate DNS traffic-related actions together. Enter the name of an existing DNS parameter map as an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. For details about configuring a DNS parameter map, see the “Configuring a DNS Parameter Map” section.

For example, to associate the DNS_PARAMMAP DNS parameter map with the L4_POLICY Layer 3 and Layer 4 policy map, enter the following commands:

```
host1/Admin(config)# policy-map multi-match L4_POLICY
host1/Admin(config-pmap)# class L4_CLASS
host1/Admin(config-pmap-c)# appl-parameter dns advanced-options DNS_PARAMMAP
```

To disassociate the DNS parameter map from the Layer 4 policy map, enter the following commands:

```
host1/Admin(config)# policy-map multi-match L4_POLICY
host1/Admin(config-pmap)# class L4_CLASS
host1/Admin(config-pmap-c)# no appl-parameter dns advanced-options DNS_PARAMMAP
```
Configuring an HTTP Parameter Map

You can use a parameter map to combine related actions for use in a Layer 3 and Layer 4 HTTP deep packet inspection policy map. You reference this parameter map in the `appl-parameter` command in policy map class configuration mode. See the “Associating an HTTP Parameter Map with a Layer 3 and Layer 4 Policy Map” section.

You can configure advanced HTTP behavior for HTTP deep packet inspection by using the `parameter-map type http` command in configuration mode. The syntax of this command is as follows:

```
parameter-map type http name
```

The `name` argument is the identifier assigned to the parameter map. Enter an unquoted text string with no spaces and a maximum of 32 alphanumeric characters.

This section contains the following topics to define the advanced HTTP parameter map:

- Disabling Case-Sensitivity Matching
- Setting the Maximum Number of Bytes to Parse in HTTP Headers
- Setting the Maximum Number of Bytes to Parse in HTTP Content
- Associating an HTTP Parameter Map with a Layer 3 and Layer 4 Policy Map

Disabling Case-Sensitivity Matching

By default, the ACE CLI is case sensitive. To disable case-sensitivity matching for HTTP only, use the `case-insensitive` command in HTTP parameter map configuration mode. With case-insensitive matching enabled, the CLI does not distinguish between uppercase and lowercase letters. When case sensitivity is disabled, it applies to the following:

- HTTP header names and values
- URL strings
- HTTP content inspection

The syntax of this command is as follows:

```
case-insensitive
```
For example, to disable case sensitivity, enter:

```
host1/Admin(config-parammap-http)# case-insensitive
```

To reenable case-sensitive matching after it has been disabled, enter:

```
host1/Admin(config-parammap-http)# no case-insensitive
```

### Setting the Maximum Number of Bytes to Parse in HTTP Headers

You can set the maximum number of bytes to parse in HTTP headers by using the `set header-maxparse-length` command in HTTP parameter-map configuration mode. The syntax of this command is as follows:

```
set header-maxparse-length bytes
```

The `bytes` argument is the maximum number of bytes to parse for HTTP headers. Enter an integer from 1 to 65535. The default is 2048 bytes.

For example, to set the HTTP header maximum parse length to 8192, enter:

```
host1/Admin(config-parammap-http)# set header-maxparse-length 8192
```

To reset the HTTP header maximum parse length to the default of 2048 bytes, enter:

```
host1/Admin(config-parammap-http)# no set-header maxparse-length
```

### Setting the Maximum Number of Bytes to Parse in HTTP Content

You can set the maximum number of bytes to parse in HTTP content by using the `set content-maxparse-length` command in HTTP parameter map configuration mode. The syntax of this command is as follows:

```
set content-maxparse-length bytes
```

The `bytes` argument is the maximum number of bytes to parse in HTTP content. Enter an integer from 1 to 65535. The default is 4096 bytes.

For example, to set the maximum parse length to 8192, enter:

```
host1/Admin(config-parammap-http)# set content-maxparse-length 8192
```
To reset the maximum parse length to the default of 4096 bytes, enter:

```
host1/Admin(config-parammap-http)# no set content-maxparse-length
```

## Configuring an HTTP Parameter Map with a Layer 3 and Layer 4 Policy Map

You can associate an HTTP parameter map with a Layer 3 and Layer 4 policy map by using the `appl-parameter http advanced-options` command in policy map class configuration mode.

The syntax of this command is as follows:

```
appl-parameter http advanced-options name
```

The `name` argument is the name of an existing HTTP parameter map. Parameter maps aggregate HTTP traffic-related actions together. Enter the name of an existing HTTP parameter map as an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. For details about configuring an HTTP parameter map, see the “Configuring an HTTP Parameter Map” section.

For example, to specify the `appl-parameter http advanced-options` command as an action for the HTTP deep packet inspection policy map, enter:

```
host1/Admin(config)# policy-map multi-match HTTP_INSPECT_L4POLICY
host1/Admin(config-pmap)# class HTTP_INSPECT_L4CLASS
host1/Admin(config-pmap-c)# appl-parameter http advanced-options HTTP_PARAM_MAP1
```

To dissociate the HTTP parameter map as an action from the HTTP deep packet inspection policy map, enter:

```
host1/Admin(config-pmap-c)# no appl-parameter http advanced-options HTTP_PARAM_MAP1
```
Configuring an SCCP Parameter Map

You can use a parameter map to combine related actions for use in a Layer 3 and Layer 4 Skinny Client Control Protocol (SCCP) application protocol inspection policy map. You reference this parameter map in the `appl-parameter` command in policy map class configuration mode. See the “Associating an SCCP Parameter Map with a Layer 3 and Layer 4 Policy Map” section.

This section contains the following topics:

- SCCP Inspection Configuration Considerations
- Creating an SCCP Parameter Map
- Defining a Description of the SCCP Parameter Map
- Enabling Registration Enforcement
- Setting the Maximum Message ID
- Setting the Minimum and Maximum SCCP Prefix Length
- Associating an SCCP Parameter Map with a Layer 3 and Layer 4 Policy Map

**SCCP Inspection Configuration Considerations**

Be aware of the following considerations when you configure SCCP inspection on the ACE:

- If the ACE resides between the Cisco CallManager (CCM) and the IP phones, then explicit security ACLs are required to permit the TFTP traffic between the CCM and the phones because the ACE does not support TFTP fixup.

- If the IP address of an internal CCM is configured for NAT or PAT to a different IP address or port, registrations for external Cisco IP phones fail because the ACE does not support NAT or PAT of the file content transferred over TFTP. Although the ACE supports NAT of TFTP messages, it does not open a secure port for TFTP. In addition, the ACE cannot translate the CCM IP address and port that are embedded in the Cisco IP phone configuration files. The configuration files are transferred using TFTP during phone registration.
Creating an SCCP Parameter Map

You can configure SCCP packet inspection by using the `parameter-map type skinny` command in configuration mode. The syntax of this command is as follows:

```
parameter-map type skinny name
```

The `name` argument is the identifier assigned to the parameter map. Enter an unquoted text string with no spaces and a maximum of 32 alphanumeric characters.

For example, enter:

```
host1/Admin(config)# parameter-map type skinny SCCP_PARAMMAP
host1/Admin(config-parammap-skinny)#
```

To remove the parameter map from the configuration, enter:

```
host1/Admin(config)# no parameter-map type skinny SCCP_PARAMMAP
```

Defining a Description of the SCCP Parameter Map

You can provide a brief summary of the SCCP parameter map by using the `description` command in parameter map Skinny configuration mode. The syntax of this command is as follows:

```
description text
```

For the `text` argument, enter an unquoted text string with a maximum of 240 alphanumeric characters including spaces.

For example, to specify a description of an SCCP parameter map, enter the following command:

```
host1/Admin(config)# parameter-map type skinny SCCP_PARAMMAP
host1/Admin(config-parammap-skinny)# description sccp parameter map
```

To remove the description from the SCCP parameter map, enter:

```
host1/Admin(config-parammap-sccp)# no description
```
Enabling Registration Enforcement

You can configure the ACE to allow only registered Skinny clients to make calls. To accomplish this task, the ACE maintains the state of each Skinny client. After a client registers with CCM, the ACE opens a secure port (pinhole) to allow that client to make a call. By default, this feature is disabled.

To enable registration enforcement, use the `enforce-registration` command in parameter map Skinny configuration mode. The syntax of this command is as follows:

```
enforce-registration
```

For example, to enable registration enforcement for Skinny clients, enter:

```
host1/Admin(config-parammap-skinny)# enforce-registration
```

To disable registration enforcement, enter:

```
host1/Admin(config-parammap-skinny)# no enforce-registration
```

Setting the Maximum Message ID

You can set the maximum SCCP StationMessageID that the ACE allows by using the `message-id max` command in parameter map Skinny configuration mode. The syntax of this command is as follows:

```
message-id max number
```

The `number` argument is the largest value for the station message ID in hexadecimal that the ACE accepts. Enter a hexadecimal value from 0 to 4000. If a packet arrives with a station message ID greater than the maximum configured value or greater than the default value, the ACE drops the packet and generates a syslog message.

For example, to set the maximum SCCP message ID to 0x3000, enter:

```
host1/Admin(config-parammap-skinny)# message-id max 3000
```

To reset the maximum message ID to the default of 0x181, enter:

```
host1/Admin(config-parammap-skinny)# no message-id max 3000
```
Setting the Minimum and Maximum SCCP Prefix Length

By default, the ACE drops SCCP messages that have an SCCP prefix length that is less than the message ID. You can configure the ACE to check for a specific minimum prefix length. You can also configure the ACE to check for a maximum prefix length, but this check is disabled by default. The ACE drops any Skinny message packets that fail these checks and generates a syslog message.

To set the minimum and maximum SCCP prefix length, use the `sccp-prefix-len` command in parameter map Skinny configuration mode. The syntax of this command is as follows:

```
sccp-prefix-len { max number | min number }
```

The keywords and arguments are as follows:

- **max number**—Enables the check of the maximum SCCP prefix length. Enter an integer from 4 to 4000 bytes. The default is 4 bytes.
- **min number**—Specifies the minimum SCCP prefix length. Enter an integer from 4 to 4000 bytes.

For example, to set the minimum SCCP prefix length, enter:

```
host1/Admin(config-parammap-skinny)# sccp-prefix-len min 4
```

To reset the minimum SCCP prefix length to the default behavior, enter:

```
host1/Admin(config-parammap-skinny)# no sccp-prefix-len min 4
```

Associating an SCCP Parameter Map with a Layer 3 and Layer 4 Policy Map

You can associate an SCCP parameter map with a Layer 3 and Layer 4 policy map by using the `appl-parameter skinny advanced-options` command in policy map class configuration mode.

The syntax of this command is as follows:

```
appl-parameter skinny advanced-options name
```
The **name** argument is the name of an existing SCCP parameter map. Parameter maps aggregate SCCP traffic-related actions together. Enter the name of an existing SCCP parameter map as an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. For details about configuring an SCCP parameter map, see the “Configuring an SCCP Parameter Map” section.

For example, to specify the **appl-parameter skinny advanced-options** command as an action for the SCCP deep packet inspection policy map, enter:

```bash
host1/Admin(config)# policy-map multi-match SCCP_INSPECT_L4POLICY
host1/Admin(config-pmap)# class SCCP_INSPECT_L4CLASS
host1/Admin(config-pmap-c)# appl-parameter skinny advanced-options SCCP_PARAM_MAP1
```

To dissociate the SCCP parameter map as an action from the SCCP packet inspection policy map, enter:

```bash
host1/Admin(config-pmap-c)# no appl-parameter skinny advanced-options SCCP_PARAM_MAP1
```

## Configuring a SIP Parameter Map

You can use a parameter map to combine related actions for use in a Layer 3 and Layer 4 SIP deep packet inspection policy map. You reference this parameter map in the **appl-parameter** command in policy map class configuration mode. See the “Associating a SIP Parameter Map with a Layer 3 and Layer 4 Policy Map” section.

This section contains the following topics:

- SIP Inspection Configuration Considerations
- Creating a SIP Parameter Map
- Defining a Description of the SIP Parameter Map
- Configuring a Timeout for a SIP Media Secure Port
- Enabling Instant Messaging
- Enabling Maximum Forward Field Validation
- Configuring User Agent Software Version Options
- Enabling Strict Header Validation
• Enabling Non-SIP URI Detection in SIP Messages
• Enabling Logging of All Received and Transmitted SIP Packets
• Associating a SIP Parameter Map with a Layer 3 and Layer 4 Policy Map

SIP Inspection Configuration Considerations

Be aware of the following considerations when you configure SIP inspection on the ACE:

• If the IP address in the owner field (o=) is different from the IP address in the connection field (c=) of the Session Description Protocol (SDP) portion of a SIP packet, the ACE may not translate the IP address correctly. This incorrect IP address translation is caused by a limitation of the SIP protocol, which does not provide a port value in the owner field (o=).

• If a remote endpoint attempts to register with a SIP proxy server on a network protected by the ACE, the registration fails under the following conditions:
  – PAT is configured on the remote endpoint.
  – The SIP registration server is on the outside network.
  – The port value is missing in the contact field of the REGISTER message that the endpoint sends to the proxy server.

Creating a SIP Parameter Map

You can configure advanced SIP behavior for SIP deep packet inspection by using the `parameter-map type sip` command in configuration mode.

The syntax of this command is as follows:

```
parameter-map type sip name
```

The `name` argument is the identifier assigned to the parameter map. Enter an unquoted text string with no spaces and a maximum of 32 alphanumeric characters.

For example, enter:

```
host1/Admin(config)# parameter-map type sip SIP_PARAMMAP
host1/Admin(config-parammap-sip)#
```
To remove the parameter map from the configuration, enter:

```
host1/Admin(config)# no parameter-map type sip SIP_PARAMMAP
```

## Defining a Description of the SIP Parameter Map

You can provide a brief summary of the SIP parameter map by using the `description` command in parameter map SIP configuration mode. The syntax of this command is as follows:

```
description text
```

For the `text` argument, enter an unquoted text string with a maximum of 240 alphanumeric characters including spaces.

For example, to specify a description of an SIP parameter map, enter the following command:

```
host1/Admin(config)# parameter-map type sip SIP_PARAMMAP
host1/Admin(config-parammap-sip)# description sip parameter map
```

To remove the description from the SIP parameter map, enter:

```
host1/Admin(config-parammap-sip)# no description
```

## Configuring a Timeout for a SIP Media Secure Port

The ACE opens a temporary secure port (pinhole) to stream media to a SIP client. To prevent a hacker from exploiting this port, set a timeout for SIP media by using the `timeout` command in parameter map SIP configuration mode.

The syntax of this command is as follows:

```
timeout sip-media number
```

The `number` argument is the timeout in seconds for the media port. Enter an integer from 1 to 65535 seconds. The default is 5 seconds. Be sure to provide a timeout value that is large enough for streaming media applications to complete.
For example, to specify a secure streaming media port timeout value of 1 hour, enter:

```bash
host1/Admin(config)# parameter-map type sip SIP_PARAMMAP
host1/Admin(config-parammap-sip)# timeout sip-media 3600
```

To return the streaming media port timeout value to the default of 5 seconds, enter:

```bash
host1/Admin(config-parammap-sip)# no timeout sip-media 3600
```

### Enabling Instant Messaging

You can enable instant messaging (IM) over SIP after it has been disabled by using the `im` command in parameter map SIP configuration mode. The ACE supports SIMPLE RFC for IM. By default, IM is disabled.

The syntax of this command is as follows:

```bash
im
```

For example, to enable instant messaging, enter:

```bash
host1/Admin(config)# parameter-map type sip SIP_PARAMMAP
host1/Admin(config-parammap-sip)# im
```

To disable instant messaging, enter:

```bash
host1/Admin(config-parammap-sip)# no im
```

**Note**

Disabling IM results in the ACE dropping all messages that belong to the IM as specified by SIMPLE RFC extensions.

### Enabling Maximum Forward Field Validation

The Max-Forwards header field limits the number of hops that a SIP request can take on the way to its destination. This header field contains an integer that is decremented by one at each hop. If the Max-Forwards value reaches zero before the request reaches its destination, the request is rejected with a 483 Too Many Hops error response. You can instruct the ACE to validate the Max-Forwards header field value and to take appropriate action if the validation fails.
To instruct the ACE to validate the value of the Max-Forwards header field, use the `max-forward-validation` command in parameter map configuration mode.

The syntax of this command is as follows:

```
max-forward-validation { log } | { { drop | reset } [ log ] }
```

The keywords and options are as follows:

- **log**—Specifies that the ACE log a max forward validation event.
- **drop**—Specifies that the ACE drop the SIP message.
- **reset**—Specifies that the ACE reset the SIP connection.

For example, to enable Max-Forwards header field validation, enter:

```
host1/Admin(config-parammap-sip)# max-forward-validation drop log
```

To disable maximum forward field validation, enter:

```
host1/Admin(config-parammap-sip)# no max-forward-validation
```

### Configuring User Agent Software Version Options

If the software version of a user agent (UA) were exposed, the UA may be more vulnerable to attacks from hackers who exploit the security holes present in that particular version of software. To protect the UA from such attacks, the ACE allows you to log or mask the UA software version.

To configure the UA software version options, use the `software-version` command in parameter map SIP configuration mode.

The syntax of this command is as follows:

```
software-version { log } | { mask [ log ] }
```

The keywords are as follows:

- **log**—Specifies that the ACE log the UA software version.
- **mask**—Specifies that the ACE mask the UA software version.

For example, to configure the ACE to mask the UA software version, enter:

```
host1/Admin(config-parammap-sip)# software-version mask
```
To return the ACE behavior to the default of not checking the software version, enter:

```
host1/Admin(config-parammap-sip)# no software-version mask
```

## Enabling Strict Header Validation

The ACE checks SIP packet headers for the presence of the following mandatory SIP header fields:

- From
- To
- Call-ID
- CSeq

When one of these fields is not present in the header, the ACE drops the SIP message and generates a syslog message for the reason of the dropped message.

However, by default, the ACE does not check SIP header packets for the VIA and Max-Forwards header fields. If these fields are not present, the ACE does not drop the message.

You can enable strict header validation on the ACE to check for the presence of these fields. When validation is enabled and either of the VIA and Max-Forwards header fields is missing in a SIP packet, the ACE considers that packet invalid.

The ACE also checks for forbidden header fields, according to RFC 3261.

To enable strict header validation and the action that you want the ACE to perform if a SIP header does not meet the validation requirements, use the `strict-header-validation` command in parameter map SIP configuration mode.

The syntax of this command is as follows:

```
strict-header-validation {log} | {{drop | reset} [log]}
```

The keywords and options are as follows:

- **log**—Specifies that the ACE log the header validation event.
- **drop**—Specifies that the ACE drop the SIP message.
Caution

In some cases, the use of the **drop** option can lead to problems with some phones that do not use the mandatory headers in the request. For example, when a call is made and then cancelled, the phone receives a 487 Request Terminated cancel status request and transmits an ACK. However, for the Cisco IP Phone 7960, the transmitted ACK does not contain the Max-Forwards header, which is a mandatory header for ACK. The ACE drops this packet, which can result in operational issues with the phone.

- **reset**—Specifies that the ACE reset the connection.

For example, to enable strict header validation to instruct the ACE to drop the connection if the packet header does not meet the header validation requirements, and to log the event, enter:

```
host1/Admin(config-parammap-sip)# strict-header-validation drop log
```

To disable strict header validation, enter:

```
host1/Admin(config-parammap-sip)# no strict-header-validation drop
```

### Enabling Non-SIP URI Detection in SIP Messages

You can enable detection of non-SIP URIs in SIP messages by using the **uri-non-sip** command in parameter map SIP configuration mode.

The syntax of this command is as follows:

```
uri-non-sip {log} | {mask [log]}
```

The keywords and options are as follows:

- **log**—Specifies that the ACE log the non-SIP URI.
- **mask**—Specifies that the ACE mask the non-SIP URI.

For example, to enable the detection of non-SIP URIs in SIP messages and to log the event, enter:

```
host1/Admin(config-parammap-sip)# uri-non-sip log
```

To disable the detection of non-SIP URIs in SIP messages, enter:

```
host1/Admin(config-parammap-sip)# no uri-non-sip log
```
Enabling Logging of All Received and Transmitted SIP Packets

When you configure the ACE to load balance SIP requests among multiple servers involved in the call setup and teardown of multipoint telepresence sessions that occur on SIP over TCP or UDP, the SIP call flows through the ACE multiple times for proper setup and teardown.

The ACE handles all SIP messages and allows all headers sent in the SIP packet including proprietary headers. In case of failover for established SIP sessions over UDP, the ACE continues to process SIP packets.

By default, the ACE logs dropped packets due to internal errors through the ACE-4-607005 syslog message. However, it does not log all received and transmitted SIP packets. To enable the ACE to log these packets, use the `logging all` command in parameter map SIP configuration mode. The syntax of this command is as follows:

```
logging all
```

For example, to enable the logging of all SIP received and transmitted packets, enter:

```
host1/Admin(config-parammap-sip)# logging all
```

When this command is enabled, the ACE logs all received and transmitted SIP packets using the syslog messages ACE-6-607002 and ACE-6-607004, respectively, which contain the following information from the packet:

- Source and destination IP addresses and associated port numbers
- SIP Call-ID
- SIP message type
- Brief description of the reason for the log generation

For information on sending these messages to a remote server through the use of the `logging host` global configuration mode command, and the associated system log messages, see the *Cisco Application Control Engine Module System Message Guide*.

To disable the logging of all SIP received and transmitted packets, enter:

```
host1/Admin(config-parammap-sip)# no logging all
```
Applying a Service Policy

You can use the `service-policy` command to do the following tasks:

- Apply a previously created policy map.
- Attach the traffic policy to a specific VLAN interface or globally to all VLAN interfaces in the same context.
- Specify that the traffic policy is to be attached to the input direction of an interface.
The `service-policy` command is available at both the interface configuration mode and at the configuration mode. Specifying a policy map in the interface configuration mode applies the policy map to a specific VLAN interface. Specifying a policy map in the configuration mode applies the policy to all of the VLAN interfaces associated with a context.

The syntax of this command is as follows:

```
   service-policy input policy_name
```

The keywords and arguments are as follows:

- `input`—Specifies that the traffic policy is to be attached to the input direction of a VLAN interface. The traffic policy evaluates all traffic received by that interface.

- `policy_name`—Name of a previously defined policy map, configured with a previously created `policy-map` command. The name can be a maximum of 64 alphanumeric characters.

For example, to specify a VLAN interface and apply multiple service policies to a VLAN, enter:

```
host1/Admin(config)# interface vlan 50
host1/Admin(config-if)# ip address 172.16.1.100 255.255.255.0
host1/Admin(config-if)# service-policy input FTP_INSPECT_L4POLICY
host1/Admin(config-if)# service-policy input HTTP_INSPECT_L4POLICY
host1/Admin(config-if)# service-policy input DNS_INSPECT_L4POLICY
```

For example, to globally apply multiple service policies to all of the VLANs associated with a context, enter:

```
host1/Admin(config)# service-policy input FTP_INSPECT_L4POLICY
host1/Admin(config)# service-policy input HTTP_INSPECT_L4POLICY
host1/Admin(config)# service-policy input DNS_INSPECT_L4POLICY
```

To detach a traffic policy from a VLAN interface, enter:

```
host1/Admin(config-if)# no service-policy input DNS_INSPECT_L4POLICY
```

To globally detach a traffic policy from all VLANs associated with a context, enter:

```
host1/Admin(config)# no service-policy input DNS_INSPECT_L4POLICY
```

When you detach a traffic policy either individually from the last VLAN interface on which you applied the service policy or globally from all VLAN interfaces in the same context, the ACE automatically resets the associated service policy.
Statistics. The ACE performs this action to provide a new starting point for the service policy statistics the next time that you attach a traffic policy to a specific VLAN interface or globally to all VLAN interfaces in the same context.

Follow these guidelines when creating a service policy:

- Policy maps, applied globally in a context, are internally applied on all interfaces existing in the context.
- A policy activated on a VLAN interface overwrites any specified global policies for overlapping classification and actions.
- The ACE allows only one policy of a specific feature type to be activated on a given interface.

Examples of Application Protocol Inspection Configurations

The following examples each illustrate a running-configuration for performing:

- Layer 7 deep packet inspection of the HTTP protocol
- Layer 7 FTP command inspection
- Layer 3 and Layer 4 DNS application protocol inspection

The application protocol inspection configurations appear in bold in each example.

Layer 7 HTTP Protocol Deep Packet Inspection

In the following HTTP protocol deep packet inspection configuration, the ACE does the following:

- Includes an ACL that allows the ACE to receive any HTTP traffic through the VLAN.
- Filters on content to allow only HTTL headers that contain the “html” expression.
• Filters a subset of the HTTP traffic using a content filtering rule that permits the following packet types:
  – With an HTTP header length greater than 400 bytes
  – Without the string “BAD” included in the URL

access-list ACL1 extended permit tcp any any eq http

rserver host SERVER1
  ip address 192.168.252.245
  inservice
rserver host SERVER2
  ip address 192.168.252.246
  inservice
rserver host SERVER3
  ip address 192.168.252.247
  inservice

serverfarm host SFARM1
  probe HTTP
  rserver SERVER1
    inservice
  rserver SERVER2
    inservice
  rserver SERVER3
    inservice

class-map match-all L4_FILTERHTTP_CLASS
  2 match access-list ACL1

class-map type http inspect match-all L7_FILTERHTML1_CLASS
  2 match header Accept header-value "html"
  3 match header length request gt 400

class-map type http inspect match-all L7_FILTERHTML2_CLASS
  2 match url BAD

policy-map type loadbalance first-match L7_HTTP-LB-HTTP_POLICY
  class class-default
    serverfarm SFARM1

policy-map type inspect http all-match L7_FILTERHTTP_POLICY
  class L7_FILTERHTTPCLASS
    permit
  class L7_FILTERHTML1_CLASS
    reset

policy-map multi-match L4_FILTER_POLICY
  class L4_FILTERHTTP_CLASS
    inspect http policy L7_FILTERHTTP_POLICY

interface vlan 50
  access-group input ACL1
Layer 7 FTP Command Inspection

In the following FTP command inspection configuration, the ACE does the following:

- Masks the responses from the SYST and USER commands
- Denies selected FTP commands from executing
- Allows the remaining FTP commands to execute

access-list ACL1 line 10 extended permit ip any any

rserver host SERVER1
  ip address 192.168.252.245
  inservice
rserver host SERVER2
  ip address 192.168.252.246
  inservice
rserver host SERVER3
  ip address 192.168.252.247
  inservice

serverfarm host SFARM1
  probe FTP
  rserver SERVER1
    inservice
  rserver SERVER2
    inservice
  rserver SERVER3
    inservice

class-map type ftp inspect match-any L7_FTP-MAX-DENY_CLASS
  2 match request-method appe
  3 match request-method cdup
  4 match request-method get
  5 match request-method help
  6 match request-method mkd
  7 match request-method rmd
  8 match request-method rnfr
  9 match request-method rnto
 10 match request-method site
 11 match request-method stou
 12 match request-method cwd
Examples of Application Protocol Inspection Configurations

```plaintext
class-map type ftp inspect match-any L7_FTP-MAX-DENY2_CLASS
  2 match request-method syst
  3 match request-method user
class-map match-all L4_FTP-VIP_CLASS
  2 match virtual-address 192.168.120.119 tcp range 3333 4444
policy-map type loadbalance first-match L7_FTP-LB-SF-FTP_POLICY
  class class-default
  serverfarm SFARM1
policy-map type inspect ftp first-match L7_FTP-INSPSF-FTP_POLICY
  class L7_FTP-MAX-DENY_CLASS
      deny
  class L7_FTP-MAX-DENY2_CLASS
      mask-reply
policy-map multi-match L4_VIP_POLICY
  class L4_FTP-VIP_CLASS
      loadbalance vip inservice
      loadbalance policy L7_FTP-LB-SF-FTP_POLICY
      inspect ftp strict policy L7_FTP-INSPSF-FTP_POLICY

interface vlan 29
  ip address 172.16.0.1 255.255.255.0
  fragment chain 20
  fragment min-mtu 68
  nat-pool 1 192.168.120.71 192.168.120.71 netmask 255.255.255.0 pat
  no shutdown
interface vlan 120
  description Upstream VLAN_120 - Clients and VIPs
  ip address 192.168.120.1 255.255.255.0
  fragment chain 20
  fragment min-mtu 68
  access-group input ACL1
  nat-pool 1 192.168.120.70 192.168.120.70 netmask 255.255.255.0 pat
  service-policy input L4_VIP_POLICY
  no shutdown
  ip route 10.1.0.0 255.255.255.0 192.168.120.254
  ip route 172.16.0.0 255.252.0.0 172.16.0.253
```

Layer 3 and Layer 4 Application Protocol Inspection for DNS Inspection

In the following application protocol inspection configuration, the ACE performs DNS query inspection using a Layer 3 and Layer 4 policy map. DNS requires application inspection so that DNS queries will not be subject to the generic UDP
Viewing Application Protocol Inspection Statistics and Service Policy Information

The ACE CLI provides a comprehensive set of show commands that display application protocol inspection statistics and service policy configuration information. This section contains the following topics:

- Displaying HTTP Protocol Inspection Statistics
- Displaying Service Policy Configuration Information

Displaying HTTP Protocol Inspection Statistics

You can display FTP, HTTP, or RTSP protocol inspection statistics by using the show stats inspect command. The syntax of this command is as follows:

```
show stats inspect [ftp | http | rtsp]
```

For example, enter:

```
host1/Admin# show stats inspect http
```

```
+-------------------------------+-----------------------------+
+ HTTP Inspect statistics      +-------------------------------+
+-------------------------------+-----------------------------+
```

Use the `clear stats inspect` command to clear the HTTP protocol inspection statistics.

Table 3-9 describes the fields in the `show stats inspect` command output.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Request/Response</td>
<td>Total number of FTP, HTTP, or RTSP packet requests or responses processed by the ACE.</td>
</tr>
<tr>
<td>Total Allow Decisions</td>
<td>Total number of FTP, HTTP, or RTSP packets inspected and allowed by the ACE.</td>
</tr>
<tr>
<td>Total Drop Decisions</td>
<td>Total number of FTP, HTTP, or RTSP okpackets inspected and denied by the ACE.</td>
</tr>
<tr>
<td>Total Logging Decisions</td>
<td>Total number of syslog messages generated to track the action taken by the ACE on the matching HTTP traffic. Logging is enabled as an action in the associated HTTP inspection policy map. This field is displayed only for the <strong>http</strong> keyword.</td>
</tr>
</tbody>
</table>

### Displaying Service Policy Configuration Information

You can display service policy statistics by using the `show service-policy` command in Exec mode. The statistics that appear in the output are dependent on the configuration of the associated Layer 3 and Layer 4 policy map. The `show service-policy` command displays the following information:

- VLAN to which the policy is applied
- Class map associated with the policy
- Status of any load-balancing operations
The syntax of this command is as follows:

```
show service-policy policy_name [detail]
```

The keywords, arguments, and options are as follows:

- `policy_name`—Identifier of an existing policy map that is currently in service (applied to an interface) as an unquoted text string with a maximum of 64 alphanumeric characters.

- `detail`—(Optional) Displays a more detailed listing of policy map statistics and status information.

**Note**

The ACE updates the counters that the `show service-policy` command displays after the applicable connections are closed.

For example, to display service policy statistics for the HTTP_INSPECT_L4POLICY policy map, enter:

```
host1/Admin# show service-policy HTTP_INSPECT_L4POLICY
Status     : ACTIVE
Description: HTTP protocol deep inspection of incoming traffic
-----------------------------------------
Interface: vlan 40
  service-policy: HTTP_INSPECT_L4POLICY
    class: HTTP_INSPECT_L4CLASS
    inspect http:
      curr conns       : 0         , hit count        : 0
      dropped conns    : 0
      client pkt count : 0         , client byte count: 0
      server pkt count : 0         , server byte count: 0
    L4 policy stats:
      TotalReq/Resp: 0           TotalAllowed: 0
      TotalDropped  : 0          TotalLogged   : 0
    L7 policy: HTTP_INSPECT_L7POLICY, url logging: disabled
    L7 policy stats: Total number of L7 rules 1
    L7 class/match HTTP_INSPECT_L7CLASS: reset
      TotalInspected     : 0           TotalMatched: 0
      TotalDroppedOnError: 0          TotalLogged : 0
```

For example, to display service policy statistics for the FTP_INSPECT_L4POLICY policy map, enter:

```
host1/Admin# show service-policy FTP_INSPECT_L4POLICY
Status     : ACTIVE
Description: FTP command inspection of incoming traffic
```
For example, to display service policy statistics for the APP_INSPECT_L4POLICY policy map, enter:

```
host1/Admin# show service-policy APP_INSPECT_L4POLICY
Status     : ACTIVE

Context Global Policy:
  service-policy: APP_INSPECT_L4POLICY
    class: APP_INSPECT_L4CLASS
      inspect dns:
        max length: 0
        curr conns : 0 , hit count : 0
        dropped conns : 0
        client pkt count : 0 , client byte count: 0
        server pkt count : 0 , server byte count: 0

To clear the service policy statistics, use the clear service-policy command. The syntax of this command is as follows:

```
clear service-policy policy_name
```

The `policy_name` argument, is the identifier of an existing policy map that is currently in service (applied to an interface).

For example, to clear the statistics for the policy map HTTP_INSPECT_L4POLICY that is currently in service, enter:

```
host1/Admin# clear service-policy HTTP_INSPECT_L4POLICY
```
Table 3-10 describes the fields in the `show service-policy detail` command output for an application protocol inspection policy map.

### Table 3-10 Field Descriptions for the `show service-policy detail` Command Output

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Status of the policy map as applied in a service policy to a VLAN interface: Active or Inactive.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description about the policy map.</td>
</tr>
<tr>
<td>Context Global</td>
<td>Indicates whether the service policy has been applied globally in configuration mode to all VLAN interfaces for the context.</td>
</tr>
<tr>
<td>Policy</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td>VLAN identifier of the interface associated with the service policy.</td>
</tr>
<tr>
<td>Service-Policy</td>
<td>Identifier of the policy map.</td>
</tr>
<tr>
<td>Class</td>
<td>Identifier of the class map associated with the policy map.</td>
</tr>
<tr>
<td>Inspect DNS</td>
<td>DNS application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect HTTP</td>
<td>HTTP application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect FTP</td>
<td>FTP application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect ICMP</td>
<td>ICMP application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect ILS</td>
<td>ILS application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect RTSP</td>
<td>RTSP application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect SIP</td>
<td>SIP application protocol inspection statistics.</td>
</tr>
<tr>
<td>Inspect Skinny</td>
<td>SCCP application protocol inspection statistics.</td>
</tr>
<tr>
<td>Max Length</td>
<td>Maximum length of a DNS reply.</td>
</tr>
<tr>
<td>Strict FTP</td>
<td>Status of the strict FTP function for FTP application protocol inspection: Enabled or Disabled.</td>
</tr>
<tr>
<td>URL Logging</td>
<td>Status of the URL logging function for HTTP application protocol inspection: Enabled or Disabled.</td>
</tr>
<tr>
<td>ICMP Error</td>
<td>Status of the ICMP error function for ICMP application protocol inspection: Enabled or Disabled.</td>
</tr>
<tr>
<td>Curr Conns</td>
<td>Number of active connections.</td>
</tr>
</tbody>
</table>
### Field Descriptions for the show service-policy detail Command Output (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hit Count</td>
<td>Number of connections that the ACE.</td>
</tr>
<tr>
<td>Dropped Conns</td>
<td>Number of connections that the ACE discarded.</td>
</tr>
<tr>
<td>Client Pkt Count</td>
<td>Number of packets received from clients.</td>
</tr>
<tr>
<td>Client Byte Count</td>
<td>Number of bytes received from clients.</td>
</tr>
<tr>
<td>Server Pkt Count</td>
<td>Number of packets received from servers.</td>
</tr>
<tr>
<td>Server Byte Count</td>
<td>Number of bytes received from servers.</td>
</tr>
<tr>
<td>L4 Policy Stats</td>
<td></td>
</tr>
<tr>
<td>TotalReq/Resp</td>
<td>Total number of requests and responses for the policy map.</td>
</tr>
<tr>
<td>Total Allowed</td>
<td>Total number of packets received and allowed.</td>
</tr>
<tr>
<td>Total Dropped</td>
<td>Total number of packets received and discarded.</td>
</tr>
<tr>
<td>Total Logged</td>
<td>Total number of errors logged.</td>
</tr>
<tr>
<td>L7 Policy</td>
<td>Identifier of the policy map associated with the service policy.</td>
</tr>
<tr>
<td>L7 Policy Stats</td>
<td>Current status of the Layer 7 policy map including the total number of Layer 7 rules.</td>
</tr>
<tr>
<td>L7 Class/Match</td>
<td>Identifier of the Layer 7 HTTP deep packet inspection class map and the associated policy map match actions.</td>
</tr>
<tr>
<td>Total Inspected</td>
<td>Total number of packets inspected.</td>
</tr>
<tr>
<td>Total Matched</td>
<td>Total number of packets matched.</td>
</tr>
<tr>
<td>Total Reply Masked</td>
<td>Total number of masked system replies to the FTP SYST command. (Applicable only to the FTP SYST command and its associated reply.)</td>
</tr>
</tbody>
</table>
### Table 3-10 Field Descriptions for the show service-policy detail Command Output (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Dropped On Error</td>
<td>Total number of packets dropped due to an error in the match.</td>
</tr>
<tr>
<td>TotalLogged</td>
<td>Total number of errors logged.</td>
</tr>
</tbody>
</table>
CHAPTER 4

Configuring TCP/IP Normalization and IP Reassembly Parameters

This chapter describes how to configure TCP/IP normalization and termination parameters to protect your Cisco Application Control Engine (ACE) module and the data center from attacks. It also describes IP fragmentation and reassembly parameters. The chapter contains the following major sections:

- TCP Normalization Overview
- IP Normalization Overview
- TCP/IP Normalization and Termination Configuration Quick Start
- Configuring a Connection Parameter Map for TCP/IP Normalization and Termination
- Configuring a Traffic Policy for TCP/IP Normalization and Termination
- Configuring Interface Normalization Parameters
- Configuring IP Fragment Reassembly Parameters
- Configuring the Switch Mode Feature
- Example of a TCP/IP Normalization and IP Reassembly Configuration
- Displaying Configurations and Statistics for TCP/IP and UDP Connections, IP Reassembly, and SYN Cookie
- Clearing TCP/IP and UDP Connections and Statistics
TCP Normalization Overview

This section describes how the ACE uses TCP normalization to protect itself and the data center from a variety of network-based attacks.

TCP normalization is a Layer 4 feature that consists of a series of checks that the ACE performs at various stages of a flow, from the initial connection setup to the closing of a connection. You can control many of the segment checks by configuring one or more advanced TCP connection settings. The ACE uses these TCP connection settings to decide which checks to perform and whether to discard a TCP segment based on the results of the checks. The ACE discards segments that appear to be abnormal or malformed.

With TCP normalization, the ACE checks for segments that have invalid or suspect conditions (for example, a SYN sent to the client from the server or a SYNACK sent to the server from the client) and takes actions based on the configured parameter settings. The ACE uses TCP normalization to block certain types of network attacks (for example, insertion attacks and evasion attacks). Insertion attacks occur when the inspection module accepts a packet that the end system rejects. Evasion attacks occur when the inspection module rejects a packet while the end system accepts it.

The ACE always discards segments when the following conditions exist:

- Bad segment checksum
- Bad TCP header or payload length
- Suspect TCP flags (for example, NULL, SYN/FIN, or FIN/URG)

TCP normalization is enabled by default. If you are migrating to, or replacing legacy products with, the ACE, disable normalization using the `no normalization` command in interface configuration mode until you are sure that everything is working properly. Then, reenable normalization using the `normalization` command in interface configuration mode.

To configure TCP normalization on the ACE, you assemble various TCP commands into a parameter map. After you create the connection parameter map, you associate it with a multi-match policy map, and activate the traffic policy globally across all interfaces in the context using a service policy. For details about configuring traffic policies, see the “Configuring a Traffic Policy for TCP/IP Normalization and Termination” section.
IP Normalization Overview

In addition to TCP normalization, the ACE uses a Layer 3 feature called IP normalization to protect itself and the data center from a variety of attacks. IP normalization performs the following series of checks on IP packets:

- General security checks
- ICMP security checks
- Fragmentation security checks
- IP fragment reassembly
- IP fragmentation if a packet exceeds the outbound maximum transmission unit (MTU)

If a packet fails one of these checks, the ACE takes action (including discarding a packet) depending on the IP parameters that you configure.

Because IP normalization is always enabled on the ACE, if you have a Layer 2 connected server that sends traffic with a source MAC address that is not the one advertised by the ARP reply to received traffic, the ACE drops this traffic.

To configure the type of service (ToS) for IP traffic, use the `set ip tos` command in a connection parameter map.

To configure interface-related IP normalization parameters, see the “Configuring Interface Normalization Parameters” section.

TCP/IP Normalization and Termination Configuration Quick Start

Table 4-1 provides a quick overview of the steps required to configure TCP normalization. Each step includes the CLI command or a reference to the procedure required to complete the task. For a complete description of each feature and all the options associated with the CLI commands, see the sections following Table 4-1.
Table 4-1  TCP/IP Normalization and Termination Configuration Quick Start

Task and Command Example

1. If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, change to the correct context.

host1/Admin# changeto C1
host1/C1#

The rest of the examples in this table use the C1 user context, unless otherwise specified. For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.

2. Enter configuration mode.

host1/C1# config
host1/C1(config)#

3. Create a connection parameter map to group together TCP/IP normalization and termination parameters.

host1/C1(config)# parameter-map type connection TCPIP_PARAM_MAP
host1/C1(config-parammap-conn)#

4. Configure TCP/IP normalization parameters in the connection parameter map as required. For example, enter:

host1/C1(config-parammap-conn)# set timeout inactivity 2400
host1/C1(config-parammap-conn)# set ip tos 20
host1/C1(config-parammap-conn)# exit
host1/C1(config)#

5. Create a Layer 3 and Layer 4 TCP class map, and then configure match criteria as required.

host1/C1(config)# class-map match-any TCP_CLASS
host1/C1(config-cmap)# match destination-address 172.27.16.7
host1/C1(config-cmap)# match port tcp eq 21
host1/C1(config-cmap)# exit

6. Create a Layer 3 and Layer 4 policy map and associate the class map with it.

host1/C1(config)# policy-map multi-match TCPIP_POLICY
host1/C1(config-pmap)# class TCP_CLASS
host1/C1(config-pmap-c)# exit
host1/C1(config-pmap)# exit
Chapter 4  Configuring TCP/IP Normalization and IP Reassembly Parameters

Configuring a Connection Parameter Map for TCP/IP Normalization and Termination

Table 4-1  TCP/IP Normalization and Termination Configuration Quick Start (continued)

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Associate the connection parameter map as an action in the TCP/IP policy map.</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)# connection advanced-options</td>
</tr>
<tr>
<td>TCPIP_PARAM_MAP</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)# exit</td>
</tr>
<tr>
<td>host1/C1(config-pmap)# exit</td>
</tr>
<tr>
<td>8. Apply the policy map globally across all interfaces in the context using a service policy.</td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 50</td>
</tr>
<tr>
<td>host1/C1(config-if)# service-policy input TCPIP_POLICY</td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
</tr>
<tr>
<td>9. Configure additional IP normalization parameters in interface configuration mode.</td>
</tr>
<tr>
<td>host1/C1(config-if)# ip ttl 15</td>
</tr>
<tr>
<td>host1/C1(config-if)# ip options clear</td>
</tr>
<tr>
<td>host1/C1(config-if)# ip df allow</td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
</tr>
<tr>
<td>host1/C1(config)# exit</td>
</tr>
<tr>
<td>10. (Optional) Save your configuration changes to flash memory.</td>
</tr>
<tr>
<td>host1/C1# copy running-config startup-config</td>
</tr>
<tr>
<td>11. Display the TCP/IP normalization configuration information.</td>
</tr>
<tr>
<td>host1/C1# show running-config policy-map</td>
</tr>
<tr>
<td>host1/C1# show running-config parameter-map</td>
</tr>
<tr>
<td>host1/C1# show running-config interface</td>
</tr>
<tr>
<td>host1/C1# show service-policy name</td>
</tr>
</tbody>
</table>

Configuring a Connection Parameter Map for TCP/IP Normalization and Termination

You can configure a parameter map to group TCP/IP connection-related commands that apply to normalization and termination. After you configure the parameter map, associate it with a specific action statement in a policy map using
the connection tcp advanced-options command. For details about associating a parameter map with a policy map, see the “Associating a Connection Parameter Map with a Policy Map” section. This section contains the following topics:

- Creating a Connection Parameter Map for TCP/IP, UDP, and ICMP
- Defining a Description of the Connection Parameter Map
- Configuring Rate Limits for a Policy Map
- Setting the Maximum Receive or Transmit Buffer Share
- Setting a Range for the Maximum Segment Size
- Configuring ACE Behavior for a Segment That Exceeds the Maximum Segment Size
- Setting the Maximum Number of TCP SYN Retries
- Enabling Nagle’s Algorithm
- Enabling Random TCP Sequence Numbers
- Configuring How the ACE Handles Reserved Bits
- Configuring the Timeout for an Embryonic Connection
- Configuring the Timeout for a Half-Closed Connection
- Configuring the Connection Inactivity Timeout
- Setting How the ACE Applies TCP Optimizations to Packets
- Setting the Window Scale Factor
- Enabling the TCP Slow Start Algorithm
- Setting the ACK Delay Timer
- Configuring How the ACE Handles TCP SYN Segments that Contain Data
- Configuring How the ACE Handles TCP Options
- Setting the Urgent Pointer Policy
- Setting the Type of Service
Creating a Connection Parameter Map for TCP/IP, UDP, and ICMP

You can create a connection parameter map for TCP/IP, UDP, and ICMP by using the `parameter-map type connection` command in configuration mode. The syntax of this command is as follows:

```
parameter-map type connection map_name
```

The `map_name` argument is a unique name as an unquoted text string with no spaces with a maximum of 64 alphanumeric characters.

For example, to create a connection parameter map, enter:

```
host1/C1(config)# parameter-map type connection TCPIP_PARAM_MAP
host1/C1(config-parammap-conn)#
```

To remove the connection parameter map from the configuration, enter:

```
host1/C1(config)# no parameter-map type connection TCPIP_PARAM_MAP
```

Use one or more of the commands in the sections that follow to define the connection parameter map.

To limit the maximum number of ACE connections, create a resource class and then use the following commands:

- Through-the-ACE connections—`limit-resource conc-connections`
- To-the-ACE connections—`limit-resource mgmt-connections`

Make sure that you assign the current context to the resource class. For details about resource classes, see the *Cisco Application Control Engine Module Virtualization Configuration Guide*.

**Note**

If you configure switch mode and you configure any connection parameter-map commands (for example, `set tcp buffer-share`, `rate-limit`, `exceed-mss`, `nagle`, `random-sequence-number`, `reserved-bits`, `set tcp wan-optimization`, `timeout inactivity`, `slowstart`, and so on) either locally on a specific interface or globally on all interfaces, switch mode will override these commands for certain types of traffic. This behavior applies only to non-VIP, non-inspection, non-NATed, and non-management traffic. The ACE continues to apply local, global, and VIP-specific connection parameter maps to load-balanced (VIP), inspected, NATed, and management traffic. For details about switch mode, see the “Configuring the Switch Mode Feature” section.
Defining a Description of the Connection Parameter Map

You can provide a brief summary of the connection parameter map by using the `description` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
description text
```

For the `text` argument, enter an unquoted text string with a maximum of 240 alphanumeric characters including spaces.

For example, to specify a description of a connection parameter map, enter the following command:

```
host1/Admin#parameter-map type conn TCPIP_PARAM_MAP
host1/Admin(config-parammap-conn)# description TCPIP parameter map
```

To remove the description from the connection parameter map, enter:

```
host1/Admin(config-parammap-conn)# no description
```

Configuring Rate Limits for a Policy Map

The ACE allows you to limit the connection rate and the bandwidth rate of a policy map. The connection rate is the number of connections per second that match the policy. The bandwidth rate is the number of bytes per second that match the policy. The ACE applies these rate limits to each class map that you associate with the policy at the virtual server level.

When the connection-rate limit or the bandwidth-rate limit is reached, the ACE blocks any further traffic that matches that policy until the connection rate or bandwidth rate drops below the configured limit. By default, the ACE does not limit the connection rate or the bandwidth rate of a policy.

You can also limit the connection rate and the bandwidth rate of a real server in a server farm. For details, see the *Cisco Application Control Engine Module Server Load-Balancing Configuration Guide*. 
To limit the connection rate or the bandwidth rate of a policy, use the `rate-limit` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
rate-limit {connection number1 | bandwidth number2}
```

The keywords and arguments are as follows:

- **connection number1**—Specifies the connection-rate limit for a policy in connections per second. Enter an integer from 0 to 350000. There is no default value.
- **bandwidth number2**—Specifies the bandwidth-rate limit for a policy in bytes per second. Enter an integer from 0 to 300000000. There is no default value.

For example, to limit the connection rate of a policy to 100000 connections per second, enter:

```
host1/Admin(config)# parameter-map type connection RATE-LIMIT
host1/Admin(config-parammap-conn)# rate-limit connection 100000
```

To return the behavior of the ACE to the default of not limiting the policy connection rate, enter:

```
host1/Admin(config-parammap-conn)# no rate-limit connection 100000
```

For example, to limit the policy bandwidth rate to 5000000 bytes per second, enter:

```
host1/Admin(config)# parameter-map type connection RATE-LIMIT
host1/Admin(config-parammap-conn)# rate-limit bandwidth 5000000
```

To return the behavior of the ACE to the default of not limiting the policy bandwidth rate, enter:

```
host1/Admin(config-parammap-conn)# no rate-limit bandwidth 5000000
```

### Setting the Maximum Receive or Transmit Buffer Share

To improve throughput and overall performance, the ACE checks the number of buffered bytes on each TCP and UDP connection against the configured buffer setting before accepting new receive or transmit data. By default, the maximum size of the receive or transmit buffer for each TCP or UDP connection is 32768 bytes. For large bandwidth and delay network connections, you may want to increase the default buffer size to improve your network performance. To set the
maximum receive or transmit buffer size for each TCP and UDP connection, use the `set tcp buffer-share` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set tcp buffer-share number
```

The `number` argument is the maximum size of the receive or transmit buffer in bytes for each TCP and UDP connection. Enter an integer from 8192 to 262143 bytes. The default is 32768 bytes.

⚠️ **Caution**
If you are using the ACE to terminate SSL traffic, do not decrease the buffer share value below the default value of 32 KB. With a buffer share value of less than 32 KB, SSL connections are significantly slower.

✏️ **Note**
If you set the `set content-maxparse-length` or `set header-maxparse-length` command in HTTP parameter-map configuration mode to a value that is greater than 32 KB, you must configure the `set tcp buffer-share` command in a connection parameter map to a value that is greater than either of the other command’s value. If you do not, even if you configure `length-exceed continue` command, the ACE may not completely parse a content string or a header packet that is greater than 32 KB. The reason is that the default value of the `set tcp buffer-share` command buffer size (32 KB) will not accommodate the larger content string size.

For example, enter:

```
host1/C1(config-parammap-conn)# set tcp buffer-share 16384
```

To reset the buffer limit to the default value of 32768 bytes, enter:

```
host1/C1(config-parammap-conn)# no set tcp buffer-share
```

### Setting a Range for the Maximum Segment Size

The maximum segment size (MSS) is the largest amount of TCP data that the ACE accepts in one segment. To prevent the transmission of many smaller segments that waste bandwidth or very large segments that may require fragmentation, you
can set the minimum and maximum acceptable sizes of the MSS. To set the MSS, use the `set tcp mss` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set tcp mss min number1 max number2
```

The keywords and arguments are as follows:

- **min number1**—Specifies the smallest segment size that the ACE will accept. Enter an integer from 0 to 65535 bytes. The default is 0 bytes. The `min number` value must be less than or equal to the `max number` value. A value of 0 instructs the ACE to not perform a minimum MSS check on the incoming segment.

- **max number2**—Specifies the largest segment size that the ACE will accept. Enter an integer from 0 to 65535 bytes. The default is 1460 bytes. The `max number` value must be greater than or equal to the `min number` value. A value of 0 instructs the ACE to not perform a maximum MSS check on the incoming segment.

⚠️ **Caution**

If you configure a Layer 7 policy map and set the maximum transmit unit (MTU) of the ACE server-side VLAN lower than the client maximum segment size (MSS), ensure that the maximum value of the MSS that you set for the ACE using the `set tcp mss max` command is at least 40 bytes (the size of the TCP header plus options) less than the MTU of the ACE server-side VLAN. Otherwise, the ACE may discard incoming packets from the server.

Both the host and the server can set the MSS when they first establish a connection. If either maximum exceeds the value that you set with the `set tcp mss max` command, the ACE overrides the maximum value and inserts the value that you set. If either maximum is less than the value that you set with the `set tcp mss min` command, the ACE overrides the maximum and inserts the minimum value that you set (the minimum value is actually the smallest maximum allowed). For example, if you set a maximum size of 1200 bytes and a minimum size of 400 bytes, when a host requests a maximum size of 1300 bytes, the ACE alters the packet to request 1200 bytes (the maximum). If another host requests a maximum value of 300 bytes, the ACE alters the packet to request 400 bytes (the minimum).

If the host or server does not request an MSS, the ACE assumes that the RFC 793 default value of 536 bytes is in effect.
For example, to set the minimum acceptable MSS size to 768 bytes, and the maximum acceptable MSS size to 1500, enter:

```
host1/C1(config-parammap-conn)# set tcp mss min 768 max 1500
```

To reset the minimum MSS to the default value of 0 bytes and the maximum MSS to the default value of 1460, enter:

```
host1/C1(config-parammap-conn)# no set tcp mss
```

### Configuring ACE Behavior for a Segment That Exceeds the Maximum Segment Size

You can configure the ACE behavior for a segment that exceeds the configured maximum segment size (MSS) by using the `exceed-mss` command in connection parameter map configuration mode. The syntax of this command is as follows:

```
exceed-mss {allow | drop}
```

The keywords are as follows:

- **allow**—Permits segments that exceed the configured MSS
- **drop**—(Default) Discards segments that exceed the configured MSS

For example, to configure the ACE to allow segments that exceed the MSS, enter:

```
host1/C1(config-parammap-conn)# exceed-mss allow
```

To reset the ACE behavior to the default of discarding segments that exceed the MSS set by a peer, enter:

```
host1/C1(config-parammap-conn)# no exceed-mss allow
```

### Setting the Maximum Number of TCP SYN Retries

You can set the maximum number of attempts that the ACE makes to transmit a TCP segment when initiating a Layer 7 connection by using the `set tcp syn-retry` command in connection parameter map configuration mode. The syntax of this command is as follows:

```
set tcp syn-retry number
```
The number argument is the number of SYN retries. Enter an integer from 1 to 15. The default is 4.

For example, to set the maximum TCP SYN retries to 3, enter:

```
host1/C1(config-parammap-conn)# set tcp syn-retry 3
```

To reset the TCP SYN retries to the default value of 4, enter:

```
host1/C1(config-parammap-conn)# no set tcp syn-retry
```

**Enabling Nagle’s Algorithm**

Nagle’s algorithm instructs a sender to buffer any data to be sent until all outstanding data has been acknowledged or until there is a full segment of data to send. The algorithm automatically concatenates a number of small buffer messages transmitted over the TCP connection. This process increases the throughput by decreasing the number of segments that need to be sent over the network. However, the interaction between the Nagle algorithm and the TCP delay acknowledgment may increase latency in your TCP connection. You should disable the Nagle algorithm when you observe an unacceptable delay in a TCP connection.

You can enable Nagle’s algorithm by using the `nagle` command in parameter map connection configuration mode. By default, this command is disabled. The syntax of this command is as follows:

```
nagle
```

For example, enter:

```
host1/C1(config-parammap-conn)# nagle
```

To disable Nagle’s algorithm, enter:

```
host1/C1(config-parammap-conn)# no nagle
```

**Enabling Random TCP Sequence Numbers**

Randomizing TCP sequence numbers adds a measure of security to TCP connections by making it more difficult for a hacker to guess or predict the next sequence number in a TCP connection. This feature is enabled by default and you
cannot disable this feature for Layer 7 flows. To enable TCP sequence number randomization after it has been disabled for Layer 4 flows, use the `random-sequence-number` command in parameter map connection configuration mode.

The syntax of this command is as follows:

```
    random-sequence-number
```

For example, to enable the use of random sequence numbers if you have disabled the feature, enter:

```
host1/C1(config-parammap-conn)# random-sequence-number
```

To disable sequence number randomization for Layer 4 flows, enter:

```
host1/C1(config-parammap-conn)# no random-sequence-number
```

**Note**
The `no random-sequence-number` command has no effect on Layer 7 flows.

## Configuring How the ACE Handles Reserved Bits

You can configure how an ACE handles segments with the reserved bits set in the TCP header by using the `reserved-bits` command in parameter map connection configuration mode. The six reserved bits in the TCP header are for future use and usually have a value of 0. The syntax of this command is as follows:

```
    reserved-bits {allow | clear | drop}
```

The keywords are as follows:

- **allow**—(Default) Permits segments with the reserved bits set in the TCP header
- **clear**—Clears the reserved bits in the TCP header and allows the segment
- **drop**—Discards segments with reserved bits set in the TCP header

For example, to configure the ACE to clear the reserved bits set in the TCP header of segments, enter:

```
host1/C1(config-parammap-conn)# reserved-bits clear
```
To reset the ACE behavior to the default of allowing reserved bits set in the TCP header of a segment, enter:

```
host1/C1(config-parammap-conn)# no reserved-bits clear
```

## Configuring the Timeout for an Embryonic Connection

Occasionally, the TCP three-way handshake for a connection may not complete for some reason. This type of connection is called an *embryonic* connection. To configure a timeout for embryonic connections, use the `set tcp timeout embryonic` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set tcp timeout embryonic seconds
```

The `seconds` argument is an integer from 0 to 4294967295 seconds. The default is 5 seconds. A value of 0 specifies that the ACE does not time out an embryonic connection.

**Note**

This command affects only Layer 4 flows and not Layer 7 flows.

For example, enter:

```
host1/C1(config-parammap-conn)# set tcp timeout embryonic 24
```

To reset the TCP embryonic connection timeout to the default value of 5 seconds, enter:

```
host1/C1(config-parammap-conn)# no set tcp timeout embryonic
```

## Configuring the Timeout for a Half-Closed Connection

A half-closed connection is a connection in which the client (or server) sends a FIN and the server (or client) ACKs the FIN without sending a FIN itself. The timer starts once this condition has occurred. To configure a timeout for a half-closed connection, use the `set tcp timeout half-closed` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set tcp timeout half-closed seconds
```
The `seconds` argument is an integer from 0 to 4294967295 seconds. The default is 3600 seconds (1 hour). A value of 0 specifies that the ACE does not time out a half-closed TCP connection.

For example, enter:

```
host1/C1(config-parammap-conn)# set tcp timeout half-closed 2400
```

To reset the TCP half-closed connection timeout to the default value of 3600 seconds, enter:

```
host1/C1(config-parammap-conn)# no set tcp timeout half-closed
```

### Setting the TCP Reassembly Timeout

The ACE uses the TCP reassembly timeout to stop reassembling TCP packets that have remained idle for the specified timeout period. To set the TCP reassembly timeout, use the `set tcp reassembly-timeout` command in parameter map connection configuration mode. The syntax of this command is:

```
set tcp reassembly-timeout seconds
```

The `seconds` argument is the time period after which the ACE stops reassembling TCP packets. Enter an integer from 1 to 255. The default is 60 seconds.

For example, to set the TCP reassembly timeout to five minutes, enter the following command:

```
host1/Admin(config-parammap-conn)# set tcp reassembly-timeout 300
```

To reset the timeout to the default value of 60 seconds, enter the following command:

```
host1/Admin(config-parammap-conn)# no set tcp reassembly-timeout
```
Chapter 4 Configuring TCP/IP Normalization and IP Reassembly Parameters

Configuring a Connection Parameter Map for TCP/IP Normalization and Termination

Configuring the Connection Inactivity Timeout

The ACE uses the connection inactivity timer to disconnect established HTTP/SSL, ICMP, TCP/IP, and UDP connections that have remained idle for the duration of the specified timeout period. To configure the connection inactivity timer, use the `set timeout inactivity` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set timeout inactivity seconds
```

The `seconds` argument is the time period after which the ACE disconnects idle established connections. Enter an integer from 0 to 3217203 seconds. The defaults are as follows:

- HTTP/SSL—300 seconds
- ICMP—2 seconds
- TCP—3600 seconds (1 hour)
- UDP—10 seconds

A value of 0 specifies that the ACE does not time out a TCP connection. The ACE rounds up the value that you enter to the nearest 30-second interval.

For example, to set the connection inactivity timeout to 2400 seconds (40 minutes), enter:

```
host1/C1(config-parammap-conn)# set timeout inactivity 2400
```

To reset the connection inactivity timeout to the default values, enter:

```
host1/C1(config-parammap-conn)# no set timeout inactivity
```

Setting How the ACE Applies TCP Optimizations to Packets

You can control how the ACE applies TCP optimizations to packets on a connection associated with a Layer 7 policy map using a round-trip time (RTT) value by using the `set tcp wan-optimization rtt` command in parameter map connection configuration mode.
TCP optimizations include the following connection parameter-map configuration mode operations:

- Nagle optimization algorithm (see the “Enabling Nagle’s Algorithm” section)
- Slow-start connection behavior (see the “Enabling the TCP Slow Start Algorithm” section)
- Acknowledgement (ACK) delay timer (see the “Setting the ACK Delay Timer” section)
- Window-scale factor (see the “Setting the Window Scale Factor” section)
- Retry settings (see the “Setting the Maximum Number of TCP SYN Retries” section)

The syntax of this command is as follows:

```
set tcp wan-optimization rtt number
```

The `number` argument is the round-trip time (RTT) in milliseconds and controls how the ACE applies TCP optimizations as follows:

- For a value of 0, the ACE applies TCP optimizations to packets for the life of a connection.
- For a value of 65535 (the default), the ACE performs normal operations (no optimizations) for the life of a connection.
- For values from 1 to 65534, the ACE applies TCP optimizations to packets based on the client RTT to the ACE as follows:
  - If the actual client RTT is less than the configured RTT, the ACE performs normal operations for the life of the connection.
  - If the actual client RTT is greater than or equal to the configured RTT, the ACE performs TCP optimizations on the packets for the life of a connection.

For example, to specify the ACE applies TCP optimizations to packets for the life of a connection, enter:

```
host1/C1(config-parammap-conn)# set tcp wan-optimization rtt 0
```

To restore the ACE behavior to the default of not optimizing TCP connections, enter:

```
host1/C1(config-parammap-conn)# no set tcp wan-optimization rtt 0
```
Setting the Window Scale Factor

The TCP window scaling feature adds support for the Window Scaling option in RFC 1323. We recommend that you increase the window size to improve TCP performance in network paths with large bandwidth, long-delay characteristics. This type of network is called a long fat network (LFN).

The window scaling extension expands the definition of the TCP window to 32 bits and then uses a scale factor to carry this 32-bit value in the 16-bit window field of the TCP header. You can increase the window size to a maximum scale factor of 14. Typical applications use a scale factor of 3 when deployed in LFNs.

For Layer 7 connections (where the ACE terminates the connection), the ACE forwards the original window scale factor from the client to the server. The ACE sends the window scale factor that you configure in the SYN-ACK to the client. The ACE window scale factor must match the server window scale factor or the `tcp-options` command must be set to clear the window scale option. Otherwise, unexpected results may occur. For more information about the `tcp-options` command, see the “Configuring How the ACE Handles TCP Options” section.

For Secure Sockets Layer (SSL) connections or for configurations where the WAN optimization RTT is set to zero (see the “Setting How the ACE Applies TCP Optimizations to Packets” section), window scale mismatches between the ACE and a server are allowed. For all other connections, the ACE window scale factor must match the server window scale factor.

To set the TCP window scale factor, use the `set tcp window-scale` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set tcp window-scale number
```

The `number` argument is an integer from 0 to 14. The default is 0.

For example, to set the TCP window scale factor to 3, enter:

```
host1/C1(config-parammap-conn)# set tcp window-scale factor 3
```

To reset to the default value of 0, enter:

```
host1/C1(config-parammap-conn)# no set tcp window-scale
```
Enabling the TCP Slow Start Algorithm

The slow start algorithm is a congestion avoidance method in which TCP increases its window size as ACK handshakes arrive. It operates by observing that the rate at which new segments should be injected into the network is the rate at which the acknowledgments are returned by the host at the other end of the connection. This feature is disabled by default. For details about the TCP slow start algorithm, see RFC 2581 and RFC 3782.

To enable the slow start algorithm, use the `slowstart` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
slowstart
```

For example, to enable the slow start feature, enter:

```
host1/C1(config-parammap-conn)# slowstart
```

To disable the slow start algorithm, enter:

```
host1/C1(config-parammap-conn)# no slowstart
```

Setting the ACK Delay Timer

You can configure the ACE to delay sending the ACK from a client to a server. Some applications require delaying the ACK for best performance. Delaying the ACK can also help reduce congestion by sending one ACK for multiple segments rather than ACKing each segment individually. To configure an ACK delay, use the `set tcp ack-delay` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set ack-delay number
```

The `number` argument is an integer from 0 to 400 ms. The default is 200 ms.

For example, to delay sending an ACK from a client to a server for 400 ms, enter:

```
host1/C1(config-parammap-conn)# set ack-delay 400
```

To reset the ACK delay timer to the default value of 200 ms, enter:

```
host1/C1(config-parammap-conn)# no set ack-delay
```
Configuring How the ACE Handles TCP SYN Segments that Contain Data

Occasionally, the ACE may receive a TCP SYN segment that contains data. You can configure the ACE to either discard the segment or flag the segment for data processing. To set the ACE behavior for SYN segments with data, use the `syn-data` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
syn-data {allow | drop}
```

The keywords are as follows:

- **allow**—(Default) Permits the SYN segments that contain data and marks them for data processing
- **drop**—Discards the SYN segments that contain data

For example, to discard SYN segments that contain data, enter:

```
host1/C1(config-parammap-conn)# syn-data drop
```

To reset the ACE behavior to the default of allowing SYN segments that contain data, enter:

```
host1/C1(config-parammap-conn)# no syn-data drop
```

Configuring How the ACE Handles TCP Options

The ACE permits you to allow or clear the following explicitly supported TCP options specified in a SYN segment:

- Selective Acknowledgement (SACK)
- Time stamp
- Window Scale

You can also specify a range of TCP option numbers for those TCP options not explicitly supported by the ACE. To configure TCP options, use the `tcp-options` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
tcp-options {range number1 number2 {allow | drop}} | {selective-ack | timestamp | window-scale} {allow | clear | drop}
```
The order of precedence for the actions in this command is as follows:

1. Drop
2. Clear
3. Allow

The keywords, options, and variables are as follows:

- **range**—Specifies the TCP options not explicitly supported by the ACE using a range of option numbers. This command enables you to allow or discard segments associated with the TCP options specified in the option range.
  
  - `number1`—Lower limit of the TCP option range. Enter either 6 or 7 or an integer from 9 to 255. See Table 4-2.
  
  - `number2`—Upper limit of the TCP option range. Enter either 6 or 7 or an integer from 9 to 255. See Table 4-2.

- **allow**—Allows any segment with the specified option set.

- **drop**—Used with the range or window-scale option only. Causes the ACE to discard any segment with the specified option set.

- **selective-ack**—Allows the ACE to inform the sender about all segments that it received. The sender needs to retransmit the lost segments only, rather than wait for a cumulative acknowledgement or retransmit segments unnecessarily. Selective ACK (SACK) can reduce the number of retransmitted segments and increase the throughput under some circumstances.

- **timestamp**—Measures the round-trip time (RTT) of a TCP segment between two nodes on a network. Time stamps are always sent and echoed in both directions.

- **window-scale**—Allows the ACE to use a window scale factor that essentially increases the size of the TCP send and receive buffers. The sender specifies a window scale factor in a SYN segment that determines the send and receive window size for the duration of the connection.

- **clear**—Default for the explicitly supported options. Clears the specified option from any segment that has it set and allows the segment.

Table 4-2 lists the TCP options available for the `tcp-options range` command.
### Table 4-2  TCP Options for the tcp options range Command

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>6</td>
<td>Echo (obsoleted by option 8)</td>
<td>[RFC1072]</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>Echo reply (obsoleted by option 8)</td>
<td>[RFC1072]</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>Partial order connection permitted</td>
<td>[RFC1693]</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>Partial order service profile</td>
<td>[RFC1693]</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>CC</td>
<td>[RFC1644]</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>CC.NEW</td>
<td>[RFC1644]</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>CC.ECHO</td>
<td>[RFC1644]</td>
</tr>
<tr>
<td>14</td>
<td>3</td>
<td>TCP alternate checksum request</td>
<td>[RFC1146]</td>
</tr>
<tr>
<td>15</td>
<td>N</td>
<td>TCP alternate checksum data</td>
<td>[RFC1146]</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>Skeeter</td>
<td>[Knowles]</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>Bubba</td>
<td>[Knowles]</td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>Trailer checksum option</td>
<td>[Subbu &amp; Monroe]</td>
</tr>
<tr>
<td>19</td>
<td>18</td>
<td>MD5 signature option</td>
<td>[RFC2385]</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>SCPS capabilities</td>
<td>[Scott]</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>Selective negative acknowledgements (SNACK)</td>
<td>[Scott]</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>Record boundaries</td>
<td>[Scott]</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>Corruption experienced</td>
<td>[Scott]</td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>SNAP</td>
<td>[Sukonnik]</td>
</tr>
<tr>
<td>25</td>
<td></td>
<td>Unassigned (released 12/18/00)</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td></td>
<td>TCP compression filter</td>
<td>[Bellovin]</td>
</tr>
</tbody>
</table>
Table 4-3 lists the TCP options explicitly supported by this command.

Table 4-3  TCP Options Explicitly Supported by the ACE

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-</td>
<td>End of option list</td>
<td>[RFC793]</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>No operation</td>
<td>[RFC793]</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>WSOPT—Window Scale</td>
<td>[RFC1323]</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>Selective acknowledgement (SACK) permitted</td>
<td>[RFC2018]</td>
</tr>
<tr>
<td>5</td>
<td>N</td>
<td>SACK</td>
<td>[RFC2018]</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>Time stamp option (TSOPT)</td>
<td>[RFC1323]</td>
</tr>
</tbody>
</table>

You can specify this command multiple times to configure different options and actions. If you specify the same option with different actions, the ACE uses the order of precedence described earlier in this section to decide which action to use.

For example, to allow a segment with the SACK option set, enter:

```bash
host1/C1(config-parammap-conn)# tcp-options selective-ack allow
```

To reset the ACE behavior to the default of clearing the SACK option and allowing the segment, enter:

```bash
host1/C1(config-parammap-conn)# no tcp-options selective-ack
```

You can specify a range of options for each action. If you specify overlapping option ranges with different actions, the ACE uses the order of precedence described earlier in this section to decide which action to perform for the specified options.

For example, enter:

```bash
host1/C1(config-parammap-conn)# tcp-options range 6 7 allow
host1/C1(config-parammap-conn)# tcp-options range 19 26 drop
```

To remove the TCP option ranges from the configuration, enter:

```bash
host1/C1(config-parammap-conn)# no tcp-options range 6 7 allow
host1/C1(config-parammap-conn)# no tcp-options range 19 26 drop
```
Setting the Urgent Pointer Policy

If the Urgent control bit (flag) is set in the TCP header, it indicates that the Urgent Pointer is valid. The Urgent Pointer contains an offset that indicates the location of the segment that follows the urgent data in the payload. Urgent data is data that should be processed as soon as possible, even before normal data is processed. The ACE permits you to allow or clear the Urgent flag. If you clear the Urgent flag, you invalidate the Urgent Pointer.

The ACE clears the Urgent flag for any traffic above Layer 4. If you have enabled TCP server connection reuse (see the Cisco Application Control Engine Module Server Load-Balancing Configuration Guide, Chapter 2, Configuring Traffic Policies for Server Load Balancing), the ACE does not pass the Urgent flag value to the server.

To set the Urgent Pointer policy, use the `urgent-flag` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
urgent-flag {allow | clear}
```

The keywords are as follows:

- **allow**—(Default) Permits the status of the Urgent flag. If the Urgent flag is set, the offset in the Urgent Pointer that indicates the location of the urgent data is valid. If the Urgent flag is not set, the offset in the Urgent Pointer is invalid.
- **clear**—Sets the Urgent flag to 0, which invalidates the offset in the Urgent Pointer and allows the segment.

For example, to clear the Urgent flag and allow the segment, enter:

```
host1/C1(config-parammap-conn)# urgent-flag clear
```

To reset the ACE behavior to the default of allowing the Urgent flag, enter:

```
host1/C1(config-parammap-conn)# no urgent-flag
```
Setting the Type of Service

The type of service (ToS) for an IP packet determines how the network handles the packet and balances its precedence, throughput, delay, reliability, and cost. This information resides in the IP header. To set the ToS for packets in a particular traffic class, use the `set ip tos` command in parameter map connection configuration mode. The syntax of this command is as follows:

```
set ip tos number
```

Use the `number` argument to replace a packet’s ToS byte value with the specified value. Enter an integer from 0 to 255. For details about the ToS byte, see RFCs 791, 1122, 1349, and 3168.

For example, to set a packet’s ToS byte value to 20, enter:

```
host1/C1(config-parammap)# set ip tos 20
```

To reset the ACE behavior to the default of not rewriting the ToS byte value of an incoming packet, enter:

```
host1/C1(config-parammap)# no set ip tos 20
```

Configuring a Traffic Policy for TCP/IP Normalization and Termination

This section describes how to configure a traffic policy for TCP/IP normalization and termination and contains the following topics:

- Configuring a Layer 4 Class Map
- Configuring a Layer 3 and Layer 4 Policy Map
- Associating a Connection Parameter Map with a Policy Map
- Associating a Layer 3 and Layer 4 Policy Map with a Service Policy

Configuring a Layer 4 Class Map

You can use a Layer 4 class map to classify network traffic for TCP/IP normalization and termination. To match the traffic class, the network traffic must satisfy the match criteria that you specify in the class map.
To configure a class map for TCP/IP normalization and termination, use the `class-map` command in configuration mode. For details about configuring a class map, see the *Cisco Application Control Engine Module Administration Guide*. The syntax of this command is as follows:

```
class-map [match-all | match-any] name
```

The keywords, arguments, and options are as follows:

- **match-all | match-any**—(Optional) Determines how the ACE evaluates Layer 4 network traffic when multiple match criteria exist in a class map. The class map is considered a match if the `match` commands meet one of the following conditions.
  - **match-all**—(Default) To match the traffic class, network traffic must satisfy all the match criteria listed in the class map, typically, `match` commands of different types.
  - **match-any**—To match the traffic class, network traffic must match only one of the match criteria listed in the class map, typically, `match` commands of the same type.

- **name**—Identifier of the class map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters. The class name is used for both the class map and to configure policy for the class in the policy map.

For example, enter:

```
host1/C1(config)# class-map match-any TCP_CLASS
host1/C1(config-cmap)#
```

To remove the class map from the configuration, enter:

```
host1/C1(config)# no class-map match-any TCP_CLASS
```

This section contains the following topics:

- Defining a Class Map Description
- Specifying IP Address Match Criteria
- Defining a TCP or UDP Port Number or Port Range Match Criteria
Defining a Class Map Description

You can use the `description` command in class-map configuration mode to provide a brief description of the Layer 4 class map. The syntax of this command is as follows:

```
description text
```

The `text` argument is an unquoted text string with a maximum of 256 alphanumeric characters.

The following example specifies a description that the class map is to filter network traffic to the server:

```
host1/C1(config)# class-map TCP_CLASS
host1/C1(config-cmap)# description filter tcp connections
```

To remove the description from the class map, enter:

```
host1/C1(config-cmap)# no description filter tcp connections
```

Continue with the following section to enter match criteria as required using the `match` command in class-map configuration mode.

Specifying IP Address Match Criteria

You can specify a source address, destination address, or VIP address as the Layer 3 network traffic match criteria by using the `match` command in class-map configuration mode. The syntax of this command is as follows:

```
[line_number] match {source-address | destination-address | virtual-address} ip_address netmask
```

The keywords, arguments, and options are as follows:

- `line_number`—(Optional) Argument that assists you in editing or deleting individual `match` commands. For example, you can enter `no line_number` to delete long `match` commands instead of entering the entire line.
- `source-address`—Specifies the source IP address as the match criteria.
- `destination-address`—Specifies the destination IP address as the match criteria.
- `virtual-address`—Specifies the virtual IP (VIP) address as the match criteria.
• **ip_address**—IP address of the source, destination, or VIP. Enter an IP address in dotted-decimal notation (for example, 192.168.12.15). You can also specify 0.0.0.0 as a wildcard that will match any IP address.

• **netmask**—(Optional) Subnet mask for the IP address. Enter a subnet mask in dotted-decimal notation (for example, 255.255.255.0). The default subnet mask is 255.255.255.255. You can also specify 0.0.0.0 as a wildcard that will match any netmask.

There can be multiple **match address** commands within a single class map. Also, you can combine other **match** commands in the same class map.

The following example specifies that the network traffic must match destination IP address 172.27.16.7:

```
host1/C1(config)# class-map match-any IP_CLASS
host1/C1(config-cmap)# match destination-address 172.27.16.7
```

To remove the destination IP address match criteria from the class map, enter:

```
host1/C1(config-cmap)# no match destination-address 172.27.16.7
```

### Defining a TCP or UDP Port Number or Port Range Match Criteria

You can specify a TCP or UDP port number or port range as the Layer 4 network traffic match criteria by using the **match port** command in class-map configuration mode. The syntax of this command is as follows:

```
[line_number] match port \{ tcp | udp \{ eq port1 \| range port2 port3 \}
```

The keywords, arguments, and options are as follows:

• **line_number**—(Optional) Argument that assists you in editing or deleting individual **match** commands. For example, you can enter **no line_number** to delete long **match** commands instead of entering the entire line.

• **tcp**—Specifies TCP.

• **udp**—Specifies UDP.

• **eq port1**—Specifies that the TCP or UDP port number of the network traffic must match the specified value. Enter an integer from 0 to 65535. A value of 0 instructs the ACE to match any port. Alternatively, you can enter a protocol keyword that corresponds to a TCP or UDP port number. See Table 4-4 for a list of supported well-known TCP port names and numbers. See Table 4-5 for a list of supported well-known UDP port names and numbers.
Table 4-4  Well-Known TCP Port Numbers and Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>53</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>ftp</td>
<td>21</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>ftp-data</td>
<td>20</td>
<td>File Transfer Protocol Data</td>
</tr>
<tr>
<td>h323</td>
<td>1720</td>
<td>H.323 Call Signaling Protocol</td>
</tr>
<tr>
<td>http</td>
<td>80</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>https</td>
<td>443</td>
<td>HTTP over TLS/SSL</td>
</tr>
<tr>
<td>irc</td>
<td>194</td>
<td>Internet Relay Chat</td>
</tr>
<tr>
<td>matip-a</td>
<td>350</td>
<td>Mapping of Airline Traffic over Internet Protocol (MATIP) Type A</td>
</tr>
<tr>
<td>nntp</td>
<td>119</td>
<td>Network News Transport Protocol</td>
</tr>
<tr>
<td>pop2</td>
<td>109</td>
<td>Post Office Protocol v2</td>
</tr>
<tr>
<td>pop3</td>
<td>110</td>
<td>Post Office Protocol v3</td>
</tr>
<tr>
<td>rtsp</td>
<td>554</td>
<td>Real Time Streaming Protocol</td>
</tr>
<tr>
<td>sip</td>
<td>5060</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>skinny</td>
<td>2000</td>
<td>Skinny Client Control Protocol (SCCP)</td>
</tr>
<tr>
<td>smtp</td>
<td>25</td>
<td>Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>Telnet</td>
</tr>
<tr>
<td>www</td>
<td>80</td>
<td>World Wide Web</td>
</tr>
</tbody>
</table>

Table 4-5  Well-Known UDP Port Numbers and Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>53</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>ras</td>
<td>1719</td>
<td>H.323 RAS protocol</td>
</tr>
<tr>
<td>sip</td>
<td>5060</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>wsp</td>
<td>9200</td>
<td>Connectionless Wireless Session Protocol (WSP)</td>
</tr>
</tbody>
</table>
### Configuring a Traffic Policy for TCP/IP Normalization and Termination

- **range port2 port3**—Specifies a port range to use for the TCP or UDP port. Enter an integer from 0 to 65535. A value of 0 instructs the ACE to match any port.

You can have multiple `match port` commands within a single class map. Also, you can combine other `match` commands with the `match port` command in the same class map.

The following example specifies that the network traffic must match on TCP port number 23 (Telnet client):

```
host1/C1(config)# class-map TCP_CLASS
host1/C1(config-cmap)# match port tcp eq 23
```

To remove the TCP port number match criterion from the class map, enter:

```
host1/C1(config-cmap)# no match port tcp eq 23
```

### Configuring a Layer 3 and Layer 4 Policy Map

You can configure a Layer 4 traffic policy for TCP normalization, termination, and reuse by using the `policy-map` command in configuration mode. The ACE attempts to match multiple classes within a Layer 4 policy map, but can match only one class within each feature. If a classification matches more than one class map, then the ACE executes all the corresponding actions. However, for a specific feature, the ACE executes only the first matching classification action. For more information about policy maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
policy-map multi-match name
```

The `name` argument is the identifier of the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.
For example, enter:

```
host1/C1(config)# policy-map multi-match TCP_POLICY
host1/C1(config-pmap)#
```

To remove a policy map from the configuration, enter:

```
host1/C1(config)# no policy-map multi-match TCP_POLICY
```

**Associating a Layer 3 and Layer 4 Class Map with a Policy Map**

You can associate a Layer 4 class map with a Layer 4 policy map by using the `class` command in policy-map configuration mode. The syntax of this command is as follows:

```
class {name1 | class-default} [insert-before name2]
```

The keywords, arguments, and options are as follows:

- **name1**—Name of a previously defined traffic class configured with the `class-map` command. Enter an unquoted text string with no spaces and a maximum of 32 alphanumeric characters.

- **class-default**—Specifies a reserved, well-known class map created by the ACE. You cannot delete or modify this class. All traffic that fails to meet the other matching criteria in the named class map belongs to the default traffic class. If none of the specified classifications match the traffic, then the ACE performs the action specified under the `class class-default` command. The `class-default` class map has an implicit `match any` statement in it that enables it to match all traffic.

  **Note** When you configure a connection parameter map with a class-default class map in a Layer 3 policy map, it is applied to all other Layer 3 class maps in the same Layer 3 policy map.

- **insert-before name2**—(Optional) Places the current class map ahead of an existing class map specified by the `name2` argument in the policy-map configuration. The ACE does not save the sequence reordering as part of the configuration.
The following example shows how to use the `insert-before` command to define the sequential order of two class maps in the policy map:

```
(config-pmap)# 10 class TCP_CLASS insert-before IP_CLASS
(config-pmap-c)#
```

To remove a class map from a Layer 4 policy map, enter:

```
(config-pmap)# no 10 class TCP_CLASS
```

### Associating a Connection Parameter Map with a Policy Map

You can associate a connection parameter map with a policy map by using the `connection advanced-options` command in policy-map class configuration mode. For details about configuring a connection parameter map, see the “Configuring a Connection Parameter Map for TCP/IP Normalization and Termination” section. The syntax of this command is as follows:

```
connection advanced-options name
```

The `name` argument is a unique identifier of an existing parameter map, specified as an unquoted text string with a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config-pmap-c)# connection advanced-options TCP_PARAM_MAP
```

To dissociate the TCP parameter map from a policy map, enter:

```
host1/C1(config-pmap-c)# no connection advanced-options TCP_PARAM_MAP
```

### Associating a Layer 3 and Layer 4 Policy Map with a Service Policy

After you configure a Layer 4 policy map with a class map, a connection parameter map, and connection parameters, you must associate the policy map with a service policy to activate it. To associate a policy map with a service policy, use the `service-policy` command in configuration mode. The syntax of this command is as follows:

```
service-policy input name
```
The keywords and arguments are as follows:

- **input**—Specifies that the service policy is to be applied to the incoming traffic
- **name**—Identifier of the policy map that you want to associate with the service policy

For example, enter:

```
host1/C1(config)# service-policy input TCP_POLICY
```

To dissociate a policy map from a service policy, enter:

```
host1/C1(config)# no service-policy input TCP_POLICY
```

## Configuring Interface Normalization Parameters

This section describes how to configure TCP/IP normalization parameters in interface configuration mode. It contains the following topics:

- Disabling TCP Normalization on an Interface
- Enabling Normalization Send Reset on an Interface
- Disabling the ICMP Security Checks on an Interface
- Configuring SYN-Cookie Denial-of-Service Protection
- Configuring How the ACE Handles the Don’t Fragment Bit
- Configuring How the ACE Handles IP Options
- Setting the IP Packet TTL
- Configuring Unicast Reverse-Path Forwarding

### Disabling TCP Normalization on an Interface

By default, TCP normalization is enabled. To disable TCP normalization on an interface, use the `no normalization` command in interface configuration mode. Disabling TCP normalization affects only Layer 4 traffic. TCP normalization is always enabled for Layer 7 traffic.
Use this command when you encounter the following two types of asymmetric flows, which would otherwise be blocked by the normalization checks that the ACE performs:

- ACE only sees the client-to-server traffic. For example, for a TCP connection, the ACE sees the SYN from the client, but not the SYN-ACK from the server. In this case, apply the `no normalization` command to the client-side VLAN.

- ACE only sees the server-to-client traffic. For example, for a TCP connection, the ACE receives a SYN-ACK from the server without having received the SYN from the client. In this case, apply the `no normalization` command to the server-side VLAN.

**Note**

With TCP normalization disabled, the ACE still sets up flows for the asymmetric traffic described above and makes entries in the connection table. Note that the ACE does not check the TCP flags and TCP state of the connection. If a connection is in the half-closed state and a new SYN arrives, the connection is still used but the states do not change. Once the connection is closed properly, the extra ACK from the server goes through as a routed connection and the address is not masked to originate from the VIP.

With TCP normalization enabled, when the ACE receives the final ACK, the ACE removes the entry from the connection table. Even if FIN/ACK retransmission occurs, the ACE drops this packet due to TCP normalization feature. This means that the client cannot receive the final ACK and keeps the LAST_ACK state until half-close timeout occurs by the client.

**Caution**

Disabling TCP normalization may expose your ACE and your data center to potential security risks. TCP normalization helps protect the ACE and the data center from attackers by enforcing strict security policies that are designed to examine traffic for malformed or malicious segments.

The syntax of this command is as follows:

```
no normalization
```

For example, to disable TCP normalization on interface VLAN 100, enter:

```
host1/C1(config)# interface vlan 100
host1/C1(config-if)# no normalization
```
To reenable TCP normalization, enter:

```
host1/C1(config-if)# normalization
```

## Enabling Normalization Send Reset on an Interface

By default, the ACE silently drops any non-SYN packet when there is no flow. To enable sending a RST to the peer so it can reset its TCP connections for any non-SYN packets that are a connection miss, use the `normalization send-reset` command. This feature is disabled by default. Use the `no` form of this command to disable the normalization send-reset function.

The syntax of this command is as follows:

```
normalization send-reset
```

Prior to enabling the `normalization send-reset` command, ensure the following:

- TCP normalization is enabled through the `normalization` command. By default, TCP normalization is enabled. Normalization must be enabled to use the `normalization send-reset` command.
- Switch mode feature is disabled (see the “Configuring the Switch Mode Feature” section).

For example, to enable sending a RST to the peer so it can reset its TCP connections for any non-SYN packets, enter:

```
host1/Admin(config)# interface vlan 200
host1/Admin(config-if)# normalization send-reset
```

To disable the normalization send-reset function, enter:

```
host1/Admin(config-if)# no normalization send-reset
```

## Disabling the ICMP Security Checks on an Interface

The ACE provides several ICMP security checks by matching ICMP reply packets with request packets and using mismatched packets to detect attacks. Also, the ACE forwards ICMP error packets only if a connection record exists pertaining to the flow for which the error packet was received. By default, the ACE ICMP security checks are enabled.
To disable the ICMP security checks, use the `no icmp-guard` command in interface mode. Use this command as part of an overall strategy to operate the ACE as a pure server load balancer. For details, see Chapter 1, Overview, in the Cisco Application Control Engine Module Server Load-Balancing Configuration Guide.

The syntax of this command is as follows:

```
no icmp-guard
```

**Caution**
Disabling the ACE ICMP security checks may expose your ACE and your data center to potential security risks. After you enter the `no icmp-guard` command, the ACE no longer performs NAT translations on the ICMP header and payload in error packets, which potentially can reveal real host IP addresses to attackers.

For example, to disable ICMP security checks on interface VLAN 100, enter:

```
host1/C1(config)# interface vlan 100
host1/C1(config-if)# no icmp-guard
```

To reenable ICMP security checks, enter:

```
host1/C1(config-if)# icmp-guard
```

---

## Configuring SYN-Cookie Denial-of-Service Protection

This section describes the SYN cookie feature that the ACE uses to protect itself and devices in the data center from Denial of Service (DoS) attacks. It covers the following topics:

- Overview of SYN Cookie DoS Protection
- Configuration and Operational Considerations
- Configuring SYN Cookie DoS Protection on an Interface
Overview of SYN Cookie DoS Protection

Occasionally, a TCP three-way handshake (SYN, SYN-ACK, ACK) may not complete for some reason. These incomplete or half-open connections are known as *embryonic* connections. Such occurrences are normal if the frequency is low. However, a large number of embryonic connections could indicate a DoS attack (SYN flood attack) by a hacker.

A SYN flood attack is characterized by a large number of SYNs sent to a server or other host from one or more hosts with source IP addresses that are invalid and unreachable. Such an attack causes half-open connections on the target host that must time out before the host can service other connection requests. When multiple hosts in different networks are used to attack a server or other host, the attack is known as a Distributed Denial of Service (DDoS). The goal of the attacker is to overwhelm the target host, consume its resources, and cause it to deny service to legitimate connection requests.

The ACE allows you to protect it and the servers and other hosts in the data center from SYN flood attacks by configuring SYN-cookie-based DoS protection for TCP connections. You configure an embryonic connection threshold, beyond which the ACE applies SYN cookie protection.

When the configured embryonic connection threshold is reached, the ACE intercepts the next SYN packet from a client. The ACE responds to the SYN with a SYN-ACK using a sequence number that is the actual SYN cookie value. The SYN cookie consists of the following:

- A 32-bit timer that increases every second. Because there are two cookie tables (current and shadow), the client has a maximum of two seconds to respond correctly to the SYN cookie-generated SYN-ACK.
- An encoding of the client MSS, which the ACE forwards to the server.
- An ACE-selected secret that is a randomly generated number.

Normally, if the SYN queue fills up, the ACE drops additional connection requests. If the SYN queue fills up on the ACE with SYN cookies enabled, the ACE continues to service a client request normally by sending a SYN-ACK to the requesting client as if the SYN queue was actually larger. The ACE uses the calculated SYN cookie value as the sequence number \( n \) and discards the SYN queue entry.

When it receives an ACK (sequence number = \( n+1 \)) from the client, the ACE verifies the validity of the secret and the SYN cookie value for a recent value of the SYN cookie timer. If the secret or the sequence number is not valid, the ACE drops the packet. If the secret and the sequence number are valid, the ACE
rebuilds the SYN queue entry based on the encoded MSS and the ACK from the client. At this point, the connection process proceeds normally; the ACE sends the newly built SYN to the server and establishes the back-end TCP connection.

When you configure SYN cookie protection, the ACE calculates the internal embryonic connection threshold value for each network processor (NP) as \( \text{configured\_threshold} \div 4 \) (fractions are not disregarded). For example, if you configure the threshold as 6, the ACE applies the threshold to each NP in a round-robin fashion in the order shown, which results in the following threshold distribution:

- NP1=2
- NP2=2
- NP3=1
- NP4=1

Because of this internal division of the threshold value, you may occasionally observe that SYN cookie protection is applied before the embryonic connection count reaches the configured threshold value. For example, suppose that you configure a threshold value of 4. Because the threshold value is divided by four internally for each NP, the internally calculated threshold is 1. After one incomplete connection attempt (SYN) is sent to an NP, the ACE activates SYN cookie protection and intercepts the second SYN going to that same NP.

**Configuration and Operational Considerations**

When you use the SYN cookie feature, be aware of the following considerations:

- If the server drops the SYN that the ACE sent to it, the ACE retries the connection up to three times by default. The retry value is configurable. For more information, see the “Setting the Maximum Number of TCP SYN Retries” section.
- When you enable the SYN cookie feature on an interface, and there is a configured ACL applied to that interface, once the SYN cookie threshold is reached the ACL will not block traffic on the interface until after the three-way handshake completes.
- SYN cookie supports only the MSS TCP option. The ACE ignores all other TCP options, even if there are problems with those other options.
- The ACE returns an MSS of 536 to the client, which is the RFC-specified default.
Chapter 4  Configuring TCP/IP Normalization and IP Reassembly Parameters

Configuring Interface Normalization Parameters

- After the client-side three-way handshake completes, the ACE will use the minimum and maximum MSS that are specified in a parameter map when it sends a SYN to the server.

- If SYN cookie is enabled, you cannot disable TCP normalization. Conversely, if TCP normalization is disabled, you cannot enable SYN cookie.

- The ACE does not generate any syslogs for SYN cookie, even if the number of embryonic connections exceeds the configured threshold, which may indicate a SYN-flood attack.

- If you are configuring the SYN cookie feature on a bridged VLAN with non-loadbalanced flows, you must configure static routes for non-loadbalanced destinations that do not reside in the same subnet as the bridge-group virtual interface (BVI).

For example, assuming the following configuration:
- BVI IP address is 192.168.1.1
- Gateway1 IP address 192.168.1.2 to reach external network 172.16.1.0
- Gateway2 IP address 192.168.1.3 to reach external network 172.31.1.0

Configure the following static routes:
- ip route 172.16.1.0 255.255.255.0 192.168.1.2
- ip route 172.31.1.0 255.255.255.0 192.168.1.3

Configuring SYN Cookie DoS Protection on an Interface

To configure SYN-cookie-based DoS protection, use the `syn-cookie` command in interface configuration mode. The syntax of this command is as follows:

```
syn-cookie number
```

The `number` is the embryonic connection threshold above which the ACE applies SYN-cookie DoS protection. Enter an integer from 1 to 65535.

For example, to configure SYN-cookie DoS protection for servers in a data center connected to VLAN 100, enter:

```
host1/C1(config)# interface vlan 100
host1/C1(config-if)# syn-cookie 4096
```

To remove SYN-cookie DoS protection from the interface, enter:

```
host1/C1(config-if)# no syn-cookie
```
Configuring How the ACE Handles the Don’t Fragment Bit

Occasionally, an ACE may receive a packet that has its Don’t Fragment (DF) bit set in the IP header. This flag tells network routers and the ACE not to fragment the packet and to forward it in its entirety. To configure how the ACE handles the DF bit, use the `ip df` command in interface configuration mode. The syntax of this command is as follows:

```
ip df {clear | allow}
```

The keywords are as follows:
- `clear`—Clears the DF bit and permits the packet. If the packet is larger than the next-hop MTU, the ACE fragments the packet.
- `allow`—Permits the packet with the DF bit set. If the packet is larger than the next-hop MTU, the ACE discards the packet and sends an ICMP unreachable message to the source host.

For example, to clear the DF bit and permit the packet, enter:

```
host1/C1(config-if)# ip df clear
```

To instruct the ACE to ignore the DF bit, enter:

```
host1/C1(config-if)# no ip df
```

Configuring How the ACE Handles IP Options

The ACE can process IP options and perform specific actions when an IP option is set in a packet. To configure how the ACE handles IP options, use the `ip options` command in interface configuration mode. The syntax of this command is as follows:

```
ip options {allow | clear | clear-invalid | drop}
```

The keywords are as follows:
- `allow`—Allows the packet with IP options set
- `clear`—Clears all IP options from the packet and allows the packet
• **clear-invalid**—(Default) Clears all IP options from the packet if the ACE encounters one or more invalid or unsupported IP options and allows the packet

• **drop**—Instructs the ACE to discard the packet regardless of any IP options that are set

For example, enter:

```
host1/C1(config-if)# ip options allow
```

To reset the ACE behavior to the default of clearing all IP options if the module encounters one or more invalid or unsupported IP options, enter:

```
host1/C1(config-if)# no ip options
```

### Setting the IP Packet TTL

The packet time to live (TTL) specifies the number of hops that a packet is allowed to reach its destination. Each router along the packet’s path decrements the TTL by one. If the packet’s TTL reaches zero before the packet reaches its destination, the packet is discarded.

To specify the minimum TTL value that the ACE accepts in the IP header of an incoming packet, use the `ip ttl` command in interface configuration mode. The default behavior of the ACE is to not rewrite the TTL value of a packet. The syntax of this command is as follows:

```
ip ttl minimum number
```

The `number` argument is the minimum number of hops that a packet is allowed to reach its destination. Enter an integer from 1 to 255 hops.

**Note**

If the TTL value of the incoming packet is lower than the configured minimum value, the ACE rewrites the TTL with the configured value. Otherwise, the ACE transmits the packet with its TTL unchanged or discards the packet if the TTL equals zero.

For example, to set the TTL to 15, enter:

```
host1/C1(config-if)# ip ttl minimum 15
```
To reset the behavior of the ACE to the default of not overwriting the TTL of an incoming IP packet, enter:

```
host1/C1(config-if)# no ip ttl minimum
```

### Configuring Unicast Reverse-Path Forwarding

Unicast reverse-path forwarding (URPF) helps to mitigate problems caused by the introduction of malformed or forged (spoofed) IP source addresses into a network by allowing the ACE to discard IP packets that lack a verifiable source IP address. This feature enables the ACE to filter both ingress and egress packets to verify addressing and route integrity. It is called RPF because the route lookup is typically based on the destination address, not the source address.

When you enable this feature, the ACE discards packets if there is no route found or if the route does not match the interface on which the packet arrived.

**Note**

If you configure the `mac-sticky` command on the interface, you cannot configure the `ip verify reverse-path` command. For details about the `mac-sticky` command, see the *Cisco Application Control Engine Module Routing and Bridging Configuration Guide*.

To enable this feature, use the `ip verify reverse-path` command in interface configuration mode. The syntax of this command is as follows:

```
ip verify reverse-path
```

For example, to enable reverse-path forwarding, enter:

```
host/C1(config-if)# ip verify reverse-path
```

To disable reverse-path forwarding, enter:

```
host/C1(config-if)# no ip verify reverse-path
```

### Configuring IP Fragment Reassembly Parameters

You can configure several parameters that control how the ACE performs IP fragment reassembly. This section contains the following topics:
IP Fragment Reassembly Configuration Quick Start

Table 4-6 provides a quick overview of the steps required to configure IP fragment reassembly. Each step includes the CLI command or a reference to the procedure required to complete the task. For a complete description of each feature and all the options associated with the CLI commands, see the sections following Table 4-6.

**Table 4-6  IP Fragment Reassembly Configuration Quick Start**

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong> If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, change to the correct context.</td>
</tr>
<tr>
<td>host1/Admin# changeto C1</td>
</tr>
<tr>
<td>host1/C1#</td>
</tr>
<tr>
<td>The rest of the examples in this table use the C1 context, unless otherwise specified. For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.</td>
</tr>
<tr>
<td><strong>2.</strong> Enter configuration mode.</td>
</tr>
<tr>
<td>host1/C1# config</td>
</tr>
<tr>
<td>host1/C1(config)#</td>
</tr>
<tr>
<td><strong>3.</strong> Enter interface configuration mode for the interface on which you want to configure fragment reassembly parameters.</td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 100</td>
</tr>
<tr>
<td>host1/C1(config-if)#</td>
</tr>
</tbody>
</table>
Configuring the MTU for an Interface

The default maximum transmission unit (MTU) is 1500 bytes in a block for Ethernet interfaces. This value is sufficient for most applications, but you can pick a lower number if network conditions require it. Data that is larger than the MTU value is fragmented before being sent to the next hop router.

Caution

If you configure a Layer 7 policy map and set the maximum transmit unit (MTU) of the ACE server-side VLAN lower than the client maximum segment size (MSS), ensure that the maximum value of the MSS that you set for the ACE using the `set tcp mss max` command is at least 40 bytes (size of the TCP header plus options) less than the MTU of the ACE server-side VLAN. Otherwise, the ACE may discard incoming packets from the server.
To specify the MTU for an interface, use the `mtu` command in interface configuration mode. This command allows you to set the data size that is sent on a connection. The syntax of this command is as follows:

```
mtu bytes
```

The `bytes` argument is the number of bytes in the MTU. Enter a number from 68 to 9216 bytes. The default is 1500 bytes.

To specify the MTU data size of 1000 bytes for an interface, enter:

```
host1/admin(config-if)# mtu 1000
```

To reset the MTU block size to 1500 bytes, use the `no mtu` command. For example, enter:

```
host1/admin(config-if)# no mtu
```

### Configuring the Maximum Number of Fragments in a Packet

You can configure the maximum number of fragments belonging to the same packet that the ACE accepts for reassembly by using the `fragment chain` command in interface configuration mode. The syntax of this command is as follows:

```
fragment chain number
```

The `number` argument, is fragment chain limit as an integer from 1 to 256 fragments. The default is 24 fragments.

For example, enter:

```
host1/C1(config-if)# fragment chain 126
```

To reset the maximum number of fragments in a packet to the default of 24, enter:

```
host1/C1(config-if)# no fragment chain
```

### Configuring the Minimum Fragment Size for Reassembly

You can configure the minimum fragment size that the ACE accepts for reassembly by using the `fragment min-mtu` command in interface configuration mode. The syntax of this command is as follows:
**Chapter 4 Configuring TCP/IP Normalization and IP Reassembly Parameters**

**Configuring the Switch Mode Feature**

Use the switch mode feature to change the way that the ACE handles:

- Connections that are not destined to a particular class map.
- Connections that do not have any policies (a policy map and service policy) associated with their traffic.
- Connections that do not have load-balance (VIP), inspection, NAT, or management traffic actions associated with a traffic policy.

---

**Configuring the Switch Mode Feature**

**fragment min-mtu number**

The *number* argument is the minimum fragment size as an integer from 28 to 9216 bytes. The default is 576 bytes.

For example, enter:

```
host1/C1(config-if)# fragment min-mtu 1024
```

To reset the minimum fragment size to the default value of 576 bytes, enter:

```
host1/C1(config-if)# no fragment min-mtu
```

**Configuring an IP Reassembly Timeout**

The IP reassembly timeout specifies the period of time after which the ACE abandons the fragment reassembly process if it does not receive any outstanding fragments for the current fragment chain (fragments that belong to the same packet). To configure a reassembly timeout, use the `fragment timeout` command in interface configuration mode. The syntax of this command is as follows:

**fragment timeout seconds**

The *seconds* argument is an integer from 1 to 30 seconds. The default is 5 seconds.

For example, enter:

```
host1/C1(config-if)# fragment timeout 15
```

To reset the fragment timeout to the default value of 5 seconds, enter:

```
host1/C1(config-if)# no fragment timeout
```
When you enable the switch mode feature, the ACE processes these connections as stateless connections. Stateless TCP connections do not undergo any TCP normalization checks (for example, TCP window, TCP state, TCP sequence number, and other normalization checks). All stateless connections (such as, TCP, UDP, ICMP) have a default inactivity timeout of 2 hours and 15 minutes.

**Note**
The switch mode default timeout of 2 hours and 15 minutes applies only to non-VIP, non-inspection, non-NATed, and non-management traffic. The ACE continues to apply local, global, and VIP-specific connection parameter maps to load-balanced (VIP), inspected, NATed, and management traffic.

The ACE also creates stateless TCP connections for non-SYN TCP packets if they satisfy all other configured requirements (for example, ACLs and other policies). This process ensures that a long-lived persistent connection passes through the ACE successfully, even if the connection times out, by being reestablished by any incoming packet related to the connection. When a stateless TCP connection times out, the ACE does not send a RST packet but instead closes the connection silently. Even though these connections are stateless, the TCP RST and FIN-ACK flags are honored and the connections are closed when the ACE sees these flags in the received packets.

To change the default timeout for stateless connections associated with a traffic policy, configure a connection parameter map and include the `set timeout inactivity` command to set the connection inactivity timeout (see the “Configuring the Connection Inactivity Timeout” section), then configure a traffic policy for TCP/IP normalization and termination (see the “Configuring a Traffic Policy for TCP/IP Normalization and Termination” section).

**Note**
To configure the default timeout stateless for connections that are not associated with a traffic policy, use the `timeout` option of the `switch-mode` command.

The SYN cookie feature still operates normally for stateless TCP connections that are not destined to any traffic policy.

To enable the switch mode feature, use the `switch-mode` command in configuration mode. The syntax of this command is as follows:

```
switch-mode [timeout seconds]
```
Use the optional `timeout` keyword to configure the inactivity timeout for connections in switch mode. The `seconds` argument is the time period in seconds for idle connections after which the ACE disconnects the connection. Enter an integer from 1 to 65535. By default, the timeout is 8100 seconds.

Because UDP connections do not have a close protocol, this timeout defines their minimum lifetime. This option was introduced to minimize the number of old connections, particularly UDP connections.

For example, to enable the switch mode feature, enter the following command:

```
host1/Admin(config)# switch-mode
```

To enable the switch mode feature with a timeout of 300 seconds (5 minutes), enter the following command:

```
host1/Admin(config)# switch-mode timeout 300
```

To reset the switch-mode timeout to the default value of 8100 seconds after you have enabled switch mode and configured a timeout, enter the following command:

```
host1/Admin(config)# switch-mode
```

To disable the switch mode feature, enter the following command:

```
host1/Admin(config)# no switch-mode
```

### Example of a TCP/IP Normalization and IP Reassembly Configuration

The following example illustrates a running-configuration in which the ACE uses TCP normalization to perform checks for Layer 4 packets that have invalid or suspect conditions and to take the appropriate actions based on the configured TCP connection parameter map settings. The ACE uses TCP normalization to block certain types of network attacks. This configuration also includes IP fragment reassembly parameters. The TCP/IP normalization and IP fragment reassembly configuration appears in bold in the example.

In the following configuration, the ACE does the following:
• Includes a connection parameter map that groups together TCP/IP normalization and termination parameters, such as a connection inactivity timer, ToS for an IP packet, and discarding the SYN segments that contain data. The connection parameter map is associated as an action in the TCP/IP policy map.

• Configures additional IP normalization parameters for a specific VLAN interface, such as clearing all IP options from the packet, define the number of hops that a packet is allowed to reach its destination, and permit the packet with the DF bit set.

• Configures IP fragment reassembly parameters for a specific VLAN interface, such as the minimum fragment size that the ACE accepts for reassembly, the maximum number of fragments that belong to the same packet that the ACE accepts for reassembly, and the minimum fragment size that the ACE accepts for reassembly.

access-list ACL1 line 10 extended permit ip any any

parameter-map type connection TCPIP_PARAM_MAP
  set timeout inactivity 30
  set ip tos 20
  tcp-options timestamp allow
  syn-data drop
  urgent-flag clear

class-map match-all L4_TCP_CLASS
  description Filter TCP Connections
  2 match destination-address 172.27.16.7
  3 match port tcp eq 21

policy-map multi-match L4_TCPIP_POLICY
  class L4_TCP_CLASS
    connection advanced-options TCP_PARAM_MAP

interface vlan 50
  access-group input ACL1
  ip address 192.168.1.100 255.255.255.0
  service-policy input L4_TCPIP_POLICY
  ip ttl minimum 15
  ip options clear
  ip df allow
  fragment size 400
  fragment chain 126
  fragment min-mtu 1024
  fragment timeout 15
  no shutdown
Displaying Configurations and Statistics for TCP/IP and UDP Connections, IP Reassembly, and SYN Cookie

This section describes the `show` commands that you can use to display configurations and statistics for the following:

- TCP connections parameters
- IP connections parameters
- UDP connection parameters
- IP fragment reassembly

This section contains the following topics:

- Displaying TCP/IP and UDP Connection Configurations
- Displaying a Connection Parameter Map
- Displaying TCP/IP and UDP Connection Statistics
- Displaying Global Context Connection Statistics
- Displaying IP Statistics
- Displaying TCP Statistics
- Displaying UDP Statistics
- Displaying Service Policy Statistics
- Displaying SYN Cookie Statistics

Displaying TCP/IP and UDP Connection Configurations

You can display TCP, IP, and UDP connection configurations by using the following `show` commands in Exec mode:

- `show running-config class-map`—Displays all traffic classifications configured in the current context, including match statements for IP addresses and TCP or UDP ports

- `show running-config policy-map`—Displays all policy maps configured in the current context, including the associated class maps
• `show running-config interface`—Displays all interface VLAN configurations in the current context

For example, to display all policy maps in the current context, enter:

```
host1/C1# show running-config policy-map
```

### Displaying a Connection Parameter Map

You can display a connection parameter map configuration by using the `show parameter-map` command in Exec mode. The syntax of this command is as follows:

```
show parameter-map name
```

The `name` argument is the name of an existing connection parameter map. Enter the name as an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, to display a connection parameter map configuration, enter:

```
host1/C1# show parameter-map CONN_PMAP
```

Table 4-7 describes the fields in the `show parameter-map` command output.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter map</td>
<td>Name of the connection parameter map.</td>
</tr>
<tr>
<td>Type</td>
<td>Connection.</td>
</tr>
<tr>
<td>Nagle</td>
<td>Status of the <code>nagle</code> command: enabled or disabled.</td>
</tr>
<tr>
<td>Slow start</td>
<td>Status of the <code>slow start</code> command: enabled or disabled.</td>
</tr>
<tr>
<td>Inactivity timeout</td>
<td>Configured number of seconds after which an inactive connection times out.</td>
</tr>
<tr>
<td>(seconds)</td>
<td>Possible values are from 0 to 3217203.</td>
</tr>
<tr>
<td></td>
<td>If the <code>set timeout inactivity</code> command is not configured, the default</td>
</tr>
<tr>
<td></td>
<td>values in seconds appear, as follows:</td>
</tr>
<tr>
<td></td>
<td>• HTTP/SSL—300</td>
</tr>
<tr>
<td></td>
<td>• ICMP—2</td>
</tr>
<tr>
<td></td>
<td>• TCP—3600</td>
</tr>
<tr>
<td></td>
<td>• UDP—10</td>
</tr>
<tr>
<td>Embryonic timeout</td>
<td>Configured number of seconds after which an incomplete TCP handshake times</td>
</tr>
<tr>
<td>(seconds)</td>
<td>out. Possible values are from 0 to 4294967295.</td>
</tr>
<tr>
<td>Ack-delay</td>
<td>Configured number of seconds that the ACE delays sending an ACK from a</td>
</tr>
<tr>
<td></td>
<td>client to a server.</td>
</tr>
<tr>
<td>WAN optimization RTT</td>
<td>Configured number of milliseconds that determines how the ACE applies</td>
</tr>
<tr>
<td>(milliseconds)</td>
<td>TCP optimizations to packets on a connection that is associated with a</td>
</tr>
<tr>
<td></td>
<td>Layer 7 policy.</td>
</tr>
<tr>
<td>Half-closed timeout</td>
<td>Number of seconds after which a half-closed connection times out. Possible</td>
</tr>
<tr>
<td>(seconds)</td>
<td>values are from 0 to 4294967295.</td>
</tr>
<tr>
<td>TOS rewrite</td>
<td>State of the <code>set ip tos</code> command: enabled or disabled.</td>
</tr>
<tr>
<td>SYN retry count</td>
<td>State of the <code>set tcp syn-retry</code> command: enabled or disabled.</td>
</tr>
<tr>
<td>TCP MSS min</td>
<td>Minimum value of the TCP maximum segment size that the ACE accepts.</td>
</tr>
<tr>
<td></td>
<td>Possible values are from 0 to 65535.</td>
</tr>
<tr>
<td>TCP MSS max</td>
<td>Maximum value of the TCP maximum segment size that the ACE accepts.</td>
</tr>
<tr>
<td></td>
<td>Possible values are from 0 to 65535.</td>
</tr>
</tbody>
</table>
### Table 4-7  Field Descriptions for the show parameter-map Command Output (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tcp-options drop range</td>
<td>Range of numbers representing the TCP options that the ACE drops.</td>
</tr>
<tr>
<td>Tcp-options allow range</td>
<td>Range of numbers representing the TCP options that the ACE allows. Possible values are 6 or 7 and from 9 to 255.</td>
</tr>
<tr>
<td>Tcp-options clear range</td>
<td>Range of numbers representing the TCP options that the ACE clears. Possible values are 6 or 7 and from 9 to 255.</td>
</tr>
<tr>
<td>Selective-ack</td>
<td>Configured action that the ACE performs for the selective acknowledgement TCP option. Possible actions are allow or clear.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Configured action that the ACE performs for the time-stamp TCP option. Possible actions are allow or clear.</td>
</tr>
<tr>
<td>Window-scale</td>
<td>Configured action that the ACE performs for the window scale TCP option. Possible actions are allow, clear, or drop.</td>
</tr>
<tr>
<td>Window-scale factor</td>
<td>Value of the set tcp window-scale command. Possible values are from 0 to 14.</td>
</tr>
<tr>
<td>Reserved-bits</td>
<td>Configured action for the reserved-bits command. Possible actions are allow, clear, or drop.</td>
</tr>
<tr>
<td>Random-seq-num</td>
<td>Configured state of the random-sequence-number command. Possible states are enabled or disabled.</td>
</tr>
<tr>
<td>SYN data</td>
<td>Configured action for the syn-data command. Possible actions are allow or drop.</td>
</tr>
<tr>
<td>Exceed-mss</td>
<td>Configured action for the exceed-mss command. Possible actions are allow or drop.</td>
</tr>
<tr>
<td>urgent-flag</td>
<td>Configured action for the urgent-flag command. Possible values are allow or clear.</td>
</tr>
<tr>
<td>conn-rate-limit</td>
<td>Configured maximum number of connections per second that the ACE allows.</td>
</tr>
</tbody>
</table>
Table 4-7  Field Descriptions for the show parameter-map Command Output (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bandwidth-rate-limit</td>
<td>Configured maximum number of bytes per second that the ACE allows.</td>
</tr>
<tr>
<td>reassembly timeout (seconds)</td>
<td>Timeout configured by entering the set tcp reassembly-timeout command.</td>
</tr>
</tbody>
</table>

Displaying TCP/IP and UDP Connection Statistics

This section describes the show commands that you can use to display TCP/IP and UDP connection statistics. To display connection statistics, use the show conn command in EXEC mode. The syntax of this command is as follows:

```
show conn { address ip_address1 [ip_address2] netmask mask [detail] } | count | detail | { port number1 [number2] [detail] } | { protocol { tcp | udp } [detail] }
```

The keywords, arguments, and options are as follows:

- **address ip_address1 [ip_address2]**—Displays connection statistics for a single source or destination IP address or, optionally, for a range of source or destination IP addresses. To specify a range of IP addresses, enter an IP address for the lower limit of the range and a second IP address for the upper limit of the range. Enter one or two IP addresses in dotted-decimal notation (for example, 192.168.12.15).

- **netmask mask**—Displays the network mask for the IP address or range of IP addresses that you specify. Enter a network mask in dotted-decimal notation (for example, 255.255.255.0).

- **count**—Displays the total current connections to the ACE.

- **detail**—Displays detailed connection information.

- **port number1 [number2]**—Displays connection statistics for a single source or destination port or, optionally, for a range of source or destination ports.

- **protocol { tcp | udp }**—Displays connection statistics for TCP or UDP.
For example, to display connection statistics for a range of IP addresses, enter:

```
host1/C1#  show conn address 192.168.12.15 192.168.12.35 netmask 255.255.255.0
```

Table 4-8 describes the fields in the `show conn detail` command output.

**Table 4-8  Field Descriptions for the show conn detail Command Output**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Current Connections</td>
<td>Total number of current connections to the ACE.</td>
</tr>
<tr>
<td>Note</td>
<td>The total current connections is the number of connection objects. There are two connection objects for each flow and complete connection.</td>
</tr>
<tr>
<td>Conn-ID</td>
<td>Identifier of the inbound or outbound connection.</td>
</tr>
<tr>
<td>NP</td>
<td>Number of the network processor (NP) responsible for the connection.</td>
</tr>
<tr>
<td>Dir</td>
<td>Direction of the connection: in(bound) or out(bound).</td>
</tr>
<tr>
<td>Prot</td>
<td>Protocol used for the connection: TCP or UDP.</td>
</tr>
<tr>
<td>VLAN</td>
<td>Identifier of the interface used for the connection.</td>
</tr>
<tr>
<td>Source</td>
<td>Source IP address and port.</td>
</tr>
<tr>
<td>Destination</td>
<td>Destination IP address and port.</td>
</tr>
<tr>
<td>State</td>
<td>For TCP connections, the current state of the connection (for example, ESTAB).</td>
</tr>
<tr>
<td>Idle Time</td>
<td>Length of time that this connection has been idle.</td>
</tr>
<tr>
<td>Byte Count</td>
<td>Number of bytes that have traversed the connection.</td>
</tr>
<tr>
<td>Elapsed Time</td>
<td>Length of time that has elapsed since the connection was established.</td>
</tr>
<tr>
<td>Packet Count</td>
<td>Number of packets that have traversed the connection.</td>
</tr>
</tbody>
</table>
Displaying Global Context Connection Statistics

You can display global connection statistics for the current context by using the `show stats connection` command in Exec mode. The syntax of this command is as follows:

```
show stats connection
```

For example, to display global connection statistics for the Admin context, enter the following command:

```
host1/Admin# show stats connection
```

Table 4-9 describes the fields in the `show stats connection` command output.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Connections</td>
<td>Total number of connections that were created in the current context. This number is the sum of the Total Connections Created, Current, Destroyed, Timed-out, and Failed.</td>
</tr>
<tr>
<td>Total Connections</td>
<td>Total number of existing connections to the current context.</td>
</tr>
</tbody>
</table>
Table 4-9  
Field Descriptions for the show stats connection Command Output (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Connections Destroyed</td>
<td>Total number of connections that were torn down in the current context.</td>
</tr>
<tr>
<td>Total Connections Timed-out</td>
<td>Total number of connections that exceeded the configured timeout in the current context.</td>
</tr>
<tr>
<td>Total Connections Failed</td>
<td>Total number of connection attempts that failed to complete.</td>
</tr>
</tbody>
</table>

Displaying IP Statistics

This section describes the show commands that you can use to display IP statistics, including fragmentation, ICMP, TCP, and UDP, and ARP statistics. It contains the following topics:

- Displaying IP Traffic Information
- Displaying IP Fragmentation and Reassembly Statistics

Displaying IP Traffic Information

You can display IP traffic information by using the show ip traffic command in Exec mode. Aside from fragmentation, reassembly and ARP statistics, this command displays statistics for traffic destined to the ACE, rather than through the ACE. The syntax of this command is as follows:

```
show ip traffic
```

For example, enter:

```
host1/C1# show ip traffic
```
Table 4-10 describes the fields in the `show ip traffic` command output.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IP Statistics</strong></td>
<td></td>
</tr>
<tr>
<td>Rcvd</td>
<td>Total number of packets received by the ACE, number of bytes received by the ACE, number of input errors, number of packets received by the ACE with no route, and the number of packets received by the ACE that had an unknown protocol.</td>
</tr>
<tr>
<td>Frags</td>
<td>Number of fragments that the ACE reassembled, number of fragments that the ACE could not reassemble, number of packets that the ACE fragmented, and the number of packets that the ACE could not fragment.</td>
</tr>
<tr>
<td>Bcast</td>
<td>Number of broadcast packets received and sent.</td>
</tr>
<tr>
<td>Mcast</td>
<td>Number of multicast packets received and sent.</td>
</tr>
<tr>
<td>Sent</td>
<td>Total packets sent, number of bytes sent, and the number of packets sent with no route.</td>
</tr>
<tr>
<td>Drop</td>
<td>Number of packets discarded because they had no route, and number of packets discarded.</td>
</tr>
<tr>
<td><strong>ICMP Statistics</strong></td>
<td></td>
</tr>
</tbody>
</table>
| Rcvd  | Reports statistics for the following ICMP messages received by the ACE:  
  - Redirects  
  - ICMP Unreachable  
  - ICMP Echo  
  - ICMP Echo Reply  
  - Mask Requests  
  - Mask Replies  
  - Quench  
  - Parameter  
  - Timestamp |
Table 4-10  Field Descriptions for the show ip traffic Command Output

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sent</td>
<td>Reports statistics for the following ICMP messages sent by the ACE:</td>
</tr>
<tr>
<td></td>
<td>• Redirects</td>
</tr>
<tr>
<td></td>
<td>• ICMP Unreachable</td>
</tr>
<tr>
<td></td>
<td>• ICMP Echo</td>
</tr>
<tr>
<td></td>
<td>• ICMP Echo Reply</td>
</tr>
<tr>
<td></td>
<td>• Mask Requests</td>
</tr>
<tr>
<td></td>
<td>• Mask Replies</td>
</tr>
<tr>
<td></td>
<td>• Quench</td>
</tr>
<tr>
<td></td>
<td>• Timestamp</td>
</tr>
<tr>
<td></td>
<td>• Parameter</td>
</tr>
<tr>
<td></td>
<td>• Time Exceeded</td>
</tr>
</tbody>
</table>
You can display IP fragmentation and reassembly statistics for all interfaces in the ACE or the specified interface by using the `show fragment` command in Exec mode. The syntax of this command is as follows:

```
show fragment [vlan vlan_id]
```

For the optional `vlan_id` argument, enter the unique identifier of an existing interface as an integer from 2 to 4094. If you omit the `vlan` keyword and `vlan_id` argument, you can display statistics for all interfaces in the ACE.

For example, to display IP fragmentation and reassembly statistics for all interfaces in the ACE, enter:

```
host1/C1# show fragment
```

Table 4-11 describes the fields in the `show fragment` command output.
Displaying TCP Statistics

You can display TCP statistics by using the `show tcp statistics` command in Exec mode. This command display statistics for traffic destined to the ACE, rather than through the ACE. The syntax of this command is as follows:

```
show tcp statistics
```

For example, to display TCP statistics for the current context, enter:

```
host1/C1# show tcp statistics
```
Table 4-12 describes the fields in the **show tcp statistics** command output.

**Table 4-12  Field Descriptions for the show tcp statistics Command Output**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rcvd</td>
<td>Total number of TCP segments and errors received by the ACE.</td>
</tr>
<tr>
<td>Sent</td>
<td>Total number of TCP segments, reset flag segments, active opens, and passive opens sent by the ACE.</td>
</tr>
<tr>
<td>Connections</td>
<td>Number of failed connection attempts, established connections that were reset, and currently established connections.</td>
</tr>
</tbody>
</table>

**Displaying UDP Statistics**

You can display UDP statistics by using the **show udp statistics** command in Exec mode. This command displays statistics for traffic destined to the ACE, rather than through the ACE. The syntax of this command is as follows:

```
show udp statistics
```

For example, to display UDP statistics for the current context, enter:

```
host1/C1# show udp statistics
```

Table 4-13 describes the fields in the **show udp statistics** command output.

**Table 4-13  Field Descriptions for the show udp statistics Command Output**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rcvd</td>
<td>Total number of UDP segments, errors, and segments with no port specified that the ACE received.</td>
</tr>
<tr>
<td>Sent</td>
<td>Total number of UDP segments sent by the ACE.</td>
</tr>
</tbody>
</table>
Displaying Service Policy Statistics

You can display service-policy statistics by using the `show service-policy` command in Exec mode. The syntax of this command is as follows:

```
show service-policy name
```

The `name` argument is a unique identifier for an existing service policy, specified as an unquoted text string with a maximum of 64 alphanumeric characters.

For example, to display service-policy statistics for the current context, enter:

```
host1/C1# show service-policy POLICY1
```

Table 4-14 describes the fields in the `show service-policy` command output.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>VLAN identifier of the interface associated with the service policy.</td>
</tr>
<tr>
<td>Service-Policy</td>
<td>Identifier of the service policy.</td>
</tr>
<tr>
<td>Class</td>
<td>Identifier of the class map associated with the service policy.</td>
</tr>
<tr>
<td>Loadbalance</td>
<td>Identifier of the Layer 7 load-balancing policy map associated with the service policy.</td>
</tr>
<tr>
<td>L7 Loadbalance Policy</td>
<td>Identifier of the Layer 7 load-balancing policy map associated with the service policy.</td>
</tr>
<tr>
<td>VIP Route Metric</td>
<td>Configured distance metric for the route that needs to be entered in the routing table.</td>
</tr>
<tr>
<td>VIP Route Advertise</td>
<td>State of route health injection (RHI) using the <code>loadbalance vip advertise</code> command. Possible values are ENABLED or DISABLED.</td>
</tr>
<tr>
<td>VIP ICMP Reply</td>
<td>State of the VIP’s ability to reply to ICMP requests. Possible values are ENABLED or DISABLED.</td>
</tr>
<tr>
<td>VIP State</td>
<td>Current status of the virtual IP address. Possible values are INSERVICE or OUTOFSERVICE.</td>
</tr>
<tr>
<td>Curr Conns</td>
<td>Number of active connections.</td>
</tr>
</tbody>
</table>
Chapter 4 Configuring TCP/IP Normalization and IP Reassembly Parameters

Displaying Configurations and Statistics for TCP/IP and UDP Connections, IP Reassembly, and SYN Cookie

**Table 4-14 Field Descriptions for the show service-policy Command Output (continued)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hit Count</td>
<td>Number of connections that the ACE.</td>
</tr>
<tr>
<td>Dropped Conns</td>
<td>Number of connections that the ACE discarded.</td>
</tr>
<tr>
<td>Client Pkt Count</td>
<td>Number of packets received from clients.</td>
</tr>
<tr>
<td>Client Byte Count</td>
<td>Number of bytes received from clients.</td>
</tr>
<tr>
<td>Server Pkt Count</td>
<td>Number of packets received from servers.</td>
</tr>
<tr>
<td>Server Byte Count</td>
<td>Number of bytes received from servers.</td>
</tr>
<tr>
<td>Max Conn Limit</td>
<td>Configured maximum number of connections limit.</td>
</tr>
<tr>
<td>Conn Rate Limit</td>
<td>Configured connection rate limit.</td>
</tr>
<tr>
<td>Bandwidth Rate Limit</td>
<td>Configured bandwidth rate limit.</td>
</tr>
<tr>
<td>Drop Count</td>
<td>Number of times that a connection was dropped because the connection rate limit or the bandwidth rate limit was reached.</td>
</tr>
</tbody>
</table>

**Displaying SYN Cookie Statistics**

To display SYN cookie statistics, use the `show syn-cookie` command in Exec mode. To display SYN cookie statistics for all VLANs that are configured in the current context, enter the command with no arguments. The syntax of this command is as follows:

```
show syn-cookie [vlan number]
```

The optional `vlan number` keyword and argument instruct the ACE to display SYN cookie statistics for the specified interface. Enter an integer from 2 to 2024.
For example, to display SYN cookie statistics for VLAN 100, enter:

```
host1/C1# show syn-cookie vlan 100
```

Table 4-15 describes the fields in the `show syn-cookie` command output.

**Table 4-15  Field Descriptions for the show syn-cookie Command Output**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Name of the VLAN interface configured on the ACE.</td>
</tr>
<tr>
<td>Configured TCP Embryonic Connection Limit</td>
<td>Configured embryonic connection threshold above which the ACE applies SYN-cookie DoS protection.</td>
</tr>
<tr>
<td>Current TCP Embryonic Connection Limit</td>
<td>Number of embryonic connections that the ACE is currently tracking.</td>
</tr>
<tr>
<td>Number of TCP SYN packets Intercepted by SYN COOKIE</td>
<td>Number of client SYN packets that the ACE intercepted because the SYN-cookie embryonic connection threshold was exceeded.</td>
</tr>
<tr>
<td>Number of TCP ACKs Successfully Processed by SYN COOKIE</td>
<td>Number of client ACK packets that the ACE saw and that matched a given SYN cookie. Each client ACK that matches a cookie creates a valid embryonic connection on the ACE.</td>
</tr>
<tr>
<td>Failed Number of TCP ACKs Processed by SYN COOKIE</td>
<td>Number of client ACK packets that did not match a SYN cookie.</td>
</tr>
</tbody>
</table>

**Clearing TCP/IP and UDP Connections and Statistics**

The following sections describe the commands that you can use to clear connections and statistics related to TCP/IP and UDP connections, IP reassembly, and SYN Cookie.

- **Clearing Connections**
- **Clearing Connection Statistics**
Clearing TCP/IP and UDP Connections and Statistics

- Clearing IP, TCP, and UDP Statistics
- Clearing IP Fragmentation and Reassembly Statistics
- Clearing SYN Cookie Statistics

Clearing Connections

You can clear ICMP, TCP, and UDP connections by using the `clear conn` command in Exec mode. The syntax of this command is as follows:

```
clear conn [all | flow {icmp | tcp | udp} | rserver]
```

The keywords are as follows:
- **all**—(Optional) Clears all connections to and through the ACE in the current context.
- **flow {icmp | tcp | udp}**—(Optional) Clears all connections of the specified flow type: ICMP, TCP, or UDP.
- **rserver**—(Optional) Clears all connections for the specified real server.

For example, to clear all TCP connections in the current context, enter:
```
host1/C1# clear conn flow tcp
```

Clearing Connection Statistics

You can clear all connection statistics in the current context by using the `clear stats conn` command in Exec mode. The syntax of this command is as follows:

```
clear stats conn
```

For example, to clear all connection statistics in the Admin context, enter:
```
host1/Admin# clear stats conn
```
Clearing IP, TCP, and UDP Statistics

Use the commands in this section to clear IP, TCP, and UDP statistics. This section contains the following topics:

- Clearing IP Statistics
- Clearing TCP Statistics
- Clearing UDP Statistics

Clearing IP Statistics

You can clear IP statistics by using the `clear ip statistics` command in Exec mode. This command clears all statistics associated with IP normalization, fragmentation, and reassembly in the current context. The syntax of this command is as follows:

```
    clear ip statistics
```

For example, to clear IP statistics in the current context, enter:

```
host1/C1# clear ip statistics
```

**Note**

If you configured redundancy, then you need to explicitly clear IP statistics on both the active and the standby ACEs. Clearing statistics on the active module alone will leave the standby module’s statistics at the old values.

Clearing TCP Statistics

You can clear TCP statistics by using the `clear tcp statistics` command in Exec mode. This command clears all statistics associated with TCP connections and normalization in the current context. The syntax of this command is as follows:

```
    clear tcp statistics
```

For example, to clear TCP statistics in the current context, enter:

```
host1/C1# clear tcp statistics
```
Clearing UDP Statistics

You can clear UDP statistics by using the `clear udp statistics` command in Exec mode. This command clears all statistics associated with UDP connections in the current context. The syntax of this command is as follows:

```
clear udp statistics
```

For example, to clear UDP statistics in the current context, enter:

```
host1/C1# clear udp statistics
```

Note
If you configured redundancy, then you need to explicitly clear UDP statistics on both the active and the standby ACEs. Clearing statistics on the active module alone will leave the standby module’s statistics at the old values.

Clearing IP Fragmentation and Reassembly Statistics

You can clear IP fragmentation and reassembly statistics by using the `clear interface` command in Exec mode. The syntax of this command is as follows:

```
clear interface [vlan vlan_id]
```

The optional `vlan_id` argument is a unique identifier of an existing interface as an integer from 2 to 4094. If you omit the `vlan` keyword and `vlan_id` argument, you can clear fragmentation and reassembly statistics for all interfaces in the context.

For example, to clear IP fragmentation and reassembly statistics for all interfaces in the C1 context, enter:

```
host1/C1# clear interface
```
If you configured redundancy, then you need to explicitly clear IP fragmentation and reassembly statistics on both the active and the standby ACEs. Clearing statistics on the active module alone will leave the standby module’s statistics at the old values.

**Clearing SYN Cookie Statistics**

To clear the SYN cookie statistics, use the **clear syn-cookie** command. To clear SYN cookie statistics for all VLANs that are configured in the current context, enter the command with no arguments. The syntax of this command is as follows:

```
clear syn-cookie [vlan number]
```

The optional number argument instructs the ACE to clear SYN cookie statistics for the specified interface. Enter an integer from 2 to 2024.

For example, to clear SYN cookie statistics for VLAN 100, enter:

```
host1/C1# clear syn-cookie vlan 100
```
Configuring Network Address Translation

This chapter contains the following major sections which describe how to configure NAT on the Cisco Application Control Engine (ACE) module:

- Network Address Translation Overview
- Configuring an Idle Timeout for NAT
- Configuring Dynamic NAT and PAT
- Configuring Server Farm-Based Dynamic NAT
- Configuring Static NAT and Static Port Redirection
- Configuring Static NAT Overwrite
- Displaying NAT Configurations and Statistics
- Clearing Xlates
- NAT Configuration Examples

Network Address Translation Overview

When a client attempts to access a server in a data center, the client incorporates its IP address in the IP header when it connects to the server. An ACE placed between the client and the server can either preserve the client IP address or translate that IP address to a routable address in the server network, based on a pool of reserved dynamic NAT addresses or a static NAT address mapping, and pass the request on to the server.
This IP address translation process is called Network Address Translation (NAT) or source NAT (SNAT). The ACE tracks all SNAT mappings to ensure that response packets from the server are routed back to the client. If your application requires that the client IP address be preserved for statistical or accounting purposes, do not implement SNAT.

Destination NAT (DNAT) translates the IP address and port of an inside host so that it appears with a publicly addressable destination IP address to the rest of the world. Typically, you configure DNAT using static NAT and port redirection. You can use port redirection to configure servers that host a service on a custom port (for example, servers hosting HTTP on port 8080).

To provide security for a server, you can map the server private IP address to a global routable IP address that a client can use to connect to the server. In this case, the ACE translates the global IP address to the server private IP address when sending data from the client to the server. Conversely, when a server responds to a client, the ACE translates the local server IP address to a global IP address for security reasons. This process is called DNAT.

You can also configure the ACE to translate TCP and UDP port numbers greater than 1024, and ICMP identifiers. This process is known as Port Address Translation (PAT). The ACE provides 64 K minus 1 K ports for each IP address for PAT. Ports 0 through 1024 are reserved and cannot be used for PAT.

By default, the ACE performs implicit PAT on flows except when:

- Only routing packets
- Only bridging packets
- Performing transparent load balancing
- Server load balancing is configured with the forward action in a policy

You can also disable implicit PAT and preserve the source port when the source and destination ports are the same by using the `hw-module cde-same-port-hash` in configuration mode. For details, see the *Cisco Application Control Engine Module Server Load-Balancing Configuration Guide*.

Some of the benefits of NAT are as follows:

- You can use private addresses on your inside networks. Private addresses are not routable on the Internet.
- NAT hides the local addresses from other networks, so attackers cannot learn the real address of a server in the data center.
- You can resolve IP routing problems, such as overlapping addresses, when you have two interfaces connected to overlapping subnets.
The ACE provides the following types of NAT and PAT:

- Interface-based dynamic NAT
- Interface-based dynamic PAT
- Server farm-based dynamic NAT
- Static NAT
- Static port redirection

This section contains the following topics:

- Dynamic NAT
- Dynamic PAT
- Server Farm-Based Dynamic NAT
- Static NAT
- Static Port Redirection
- Maximum Number of NAT Commands
- Global Address Guidelines

**Dynamic NAT**

Dynamic NAT, which is typically used for SNAT, translates a group of local source addresses to a pool of global source addresses that are routable on the destination network. The global pool can include fewer addresses than the local group. When a local host accesses the destination network, the ACE assigns an IP address from the global pool to the host.

Because the translation times out after being idle for a user-configurable period of time, a given user does not keep the same IP address. For this reason, users on the destination network cannot reliably initiate a connection to a host that uses dynamic NAT (even if the connection is allowed by an access control list [ACL]). Not only can you not predict the global IP address of the host, but the ACE does not create a translation unless the local host is the initiator. See the “Configuring Static NAT and Static Port Redirection” section for details about reliable access to hosts.
For the duration of the translation, a global host can initiate a connection to the local host if an ACL allows it. Because the address is unpredictable, a connection to the host is unlikely. However, in this case, you can rely on the security of the ACL.

Dynamic NAT has these disadvantages:

- If the global address pool has fewer addresses than the local group, you could run out of addresses if the amount of traffic is greater than expected.

  Use dynamic PAT if this event occurs often, because dynamic PAT provides over 64,000 translations using multiple ports of a single IP address.

- If you need to use a large number of routable addresses in the global pool and the destination network requires registered addresses (for example, the Internet), you may encounter a shortage of usable addresses.

The ACE allows you to configure a virtual IP (VIP) address in the NAT pool for dynamic NAT and PAT. This action is useful when you want to source NAT real server originated connections (bound to the client) using the VIP address. This feature is specifically useful when there are a limited number of real world IP addresses on the client-side network. To perform PAT for different real servers that are source-NATed to the same IP address (VIP), you must configure the \texttt{pat} keyword in the \texttt{nat-pool} command.

The advantage of dynamic NAT is that some protocols cannot use dynamic PAT. Dynamic PAT does not work with some applications that have a data stream on one port and the control path on another, such as some multimedia applications.

**Dynamic PAT**

Dynamic PAT, which is also used for Stateful Network Address Translation (SNAT), translates multiple local source addresses and ports to a single global IP address and port that are routable on the destination network from a pool of IP addresses and ports reserved for this purpose. The ACE translates the local address and local port for multiple connections and/or hosts to a single global address and a unique port starting with port numbers greater than 1024.
When a local host connects to the destination network on a given source port, the ACE assigns a global IP address to it and a unique port number. Each host receives the same IP address but, because the source port number is unique, the ACE sends the return traffic, which includes the IP address and port number as the destination, to the correct host.

The ACE supports over 64,000 ports for each unique local IP address. Because the translation is specific to the local address and local port, each connection, which generates a new source port, requires a separate translation. For example, 10.1.1.1:1025 requires a separate translation from 10.1.1.1:1026.

The translation is valid only for the duration of the connection, so a user does not keep the same global IP address and port number. For this reason, users on the destination network cannot reliably initiate a connection to a host that uses dynamic PAT (even if the connection is allowed by an ACL). Not only can you not predict the local or global port number of the host, but the ACE does not create a translation unless the local host is the initiator. See the “Configuring Static NAT and Static Port Redirection” section for details about reliable access to hosts.

Dynamic PAT allows you to use a single global address, which helps to conserve routable addresses. Dynamic PAT does not work with some multimedia applications that have a data stream on a port that is different from the control path port.

**Server Farm-Based Dynamic NAT**

In addition to the interface-level dynamic NAT, the ACE supports dynamic NAT at the server farm level. Server farm-based dynamic NAT, which is also used for SNAT, is useful in situations where you want to perform NAT on only the IP addresses of the real servers in the primary and/or the backup server farm. Like interface-based dynamic NAT, server farm-based dynamic NAT uses a pool of IP addresses to translate a source address. Unlike interface-based NAT, server farm-based NAT translates the primary server farm IP addresses, the backup server farm IP addresses, or both.

Use this feature in the following cases:

- The ACE is configured in one-arm mode, that is, there is only one VLAN between the ACE and the Cisco Systems 6500 and 7600 Series Catalyst MSFC that is used for both client and server traffic. Both the primary and backup server farms are in the internal customer network (reachable from the same VLAN or from different VLANs), the primary server farm is Layer
2-attached, and the backup server farm is several Layer 3 hops away. In this case, perform NAT only for the backup server farm and never for the primary server farm.

- The ACE is configured in one-arm mode, the primary server farm is local, and the backup server farm is remote and reachable from the public, external network. In this case, use a private pool of IP addresses for SNAT of the primary server farm and a public, externally routable set of IP addresses for the backup server farm.

- You want to perform source NAT based on a Layer 7 rule or the selected server farm.

For details about configuring server farm-based dynamic NAT, see the “Configuring Server Farm-Based Dynamic NAT” section.

### Static NAT

Static NAT, which is typically used for Destination NAT (DNAT), translates each local address to a fixed global address. With dynamic NAT and PAT, each host uses a different address or port after the translation times out. Because the global address is the same for each consecutive connection with static NAT, and a persistent translation rule exists, static NAT allows hosts on the global network to initiate traffic to a local host (if there is an ACL that allows it).

The main differences between dynamic NAT and static NAT are as follows:

- Static NAT uses a one-to-one correspondence between a local IP address and a fixed global IP address, while dynamic NAT assigns a global IP address from a pool of global addresses.

- With static NAT, you need an equal number of global IP addresses and local IP addresses. With dynamic NAT, you can have a pool of fewer global addresses than local addresses.
Static Port Redirection

Static port redirection, also used for DNAT, performs the same function as static NAT and additionally translates TCP or UDP ports or ICMP identifiers for the local and global addresses. With static port redirection, you can use the same global address in multiple static NAT statements, provided that, along with the address, you use different port numbers.

For example, if you want to provide a single address for global users to access FTP, HTTP, and SMTP, but there are different servers for each protocol on the local network, you can specify static port redirection statements for each server that use the same global IP address with different ports.

Maximum Number of NAT Commands

The ACE supports the following maximum numbers of nat, nat-pool, and nat static commands divided among all contexts:

- nat command—8192
- nat-pool command—8192
- nat static command—8192

Note The ACE also supports over 8192 static NAT configurations. For more information, see the “Configuring Static NAT Overwrite” section.

Global Address Guidelines

When you translate the local address to a global address, you can use the following global addresses:

- Addresses on the same network as the global interface—If you use addresses on the same network as the global interface (through which traffic exits the ACE), the ACE uses proxy ARP to answer any requests for translated addresses and thus intercepts traffic destined for a local address. This solution simplifies routing, because the ACE does not need to be the gateway for any additional networks. However, this approach does put a limit on the number of available addresses used for translations.
Note: You cannot use the IP address of the global interface for NAT or PAT.

Addresses on a unique network—If you need more addresses than are available on the global interface network, you can identify addresses on a different subnet. The ACE uses proxy ARP to answer any requests for translated addresses, so it intercepts traffic destined for a local address. You need to add a static route on the upstream router that sends traffic destined for the translated addresses on the ACE.

You cannot configure global IP address ranges 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
```

You must configure a netmask when configuring a NAT pool. A netmask of 255.255.255.255 instructs the ACE to use all the IP addresses in the range.

## Configuring an Idle Timeout for NAT

You can configure an idle timeout for NAT by using the `timeout xlate` command in configuration mode. The syntax of this command is as follows:

```
timeout xlate seconds
```

The `seconds` argument is an integer from 60 to 2147483. The default is 10800 seconds (3 hours). The `seconds` value determines how long the ACE waits to free the Xlate slot after it becomes idle.

For example, to specify an idle timeout of 120 seconds (2 minutes), enter:
```
host1/Admin(config)# timeout xlate 120
```

To reset the NAT idle timeout to the default value of 10800 seconds, enter:
```
host1/Admin(config)# no timeout xlate 120
```
Configuring Dynamic NAT and PAT

This section describes how to configure dynamic NAT and PAT on an ACE for SNAT. For overview information about dynamic NAT and dynamic PAT, see the “Network Address Translation Overview” section. This section contains the following topics:

- Dynamic NAT and PAT Configuration Quick Start
- Configuring an ACL
- Configuring Interfaces for Dynamic NAT and PAT
- Creating a Global IP Address Pool for NAT
- Configuring a Class Map
- Configuring a Policy Map
- Configuring Dynamic NAT and PAT as a Layer 3 and Layer 4 Policy-Map Action
- Applying the Dynamic NAT and PAT Policy Map to an Interface Using a Service Policy

Dynamic NAT and PAT Configuration Quick Start

Table 5-1 provides a quick overview of the steps required to configure dynamic NAT and PAT. Each step includes the CLI command or a reference to the procedure required to complete the task. For a complete description of each feature and all the options associated with the CLI commands, see the sections following Table 5-1.
Table 5-1  Dynamic NAT and PAT Configuration Quick Start

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, change to the correct context.</td>
</tr>
<tr>
<td>host1/Admin# changeo C1</td>
</tr>
<tr>
<td>host1/C1#</td>
</tr>
</tbody>
</table>

The rest of the examples in this table use the C1 user context, unless otherwise specified. For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Enter configuration mode.</td>
</tr>
<tr>
<td>host1/C1# config</td>
</tr>
<tr>
<td>host1/C1(config)#</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. Configure an ACL to allow traffic that requires NAT.</td>
</tr>
<tr>
<td>host1/C1(config)# access-list NAT_ACCESS extended permit tcp 192.168.12.0 255.255.255.0 172.27.16.0 255.255.255.0 eq 80</td>
</tr>
<tr>
<td>host1/C1(config-acl)# exit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Configure a local interface (client interface) to receive traffic that requires NAT. If you are operating the ACE in one-arm mode, omit this step.</td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 100</td>
</tr>
<tr>
<td>host1/C1(config-if)# mtu 1500</td>
</tr>
<tr>
<td>host1/C1(config-if)# ip address 192.168.12.100 255.255.255.0</td>
</tr>
<tr>
<td>host1/C1(config-if)# no shutdown</td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Configure a second interface (server interface) for the global IP address pool.</td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 200</td>
</tr>
<tr>
<td>host1/C1(config-if)# mtu 1500</td>
</tr>
<tr>
<td>host1/C1(config-if)# ip address 172.27.16.2 255.255.255.0</td>
</tr>
<tr>
<td>host1/C1(config-if)# no shutdown</td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>6. Configure a class map and define a match statement for the ACL that you configured in Step 3 for the client source address.</td>
</tr>
<tr>
<td>host1/C1(config)# class-map match-any NAT_CLASS</td>
</tr>
<tr>
<td>host1/C1(config-cmap)# match access-list NAT_ACCESS</td>
</tr>
<tr>
<td>host1/C1(config-cmap)# exit</td>
</tr>
</tbody>
</table>
### Configuring Dynamic NAT and PAT

You can use a security access control list (ACL) to permit the traffic that requires NAT. For details about configuring an ACL, see Chapter 1, Configuring Security Access Control Lists.

To configure an ACL for dynamic NAT, use the `access-list` command in configuration mode. The syntax of this command is as follows:

```
5-11
```

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Configure a policy map and associate the class map with the policy map.</td>
</tr>
</tbody>
</table>
| host1/C1(config)# policy-map multi-match NAT_POLICY  
host1/C1(config-pmap)# class NAT_CLASS  
host1/C1(config-pmap-c)# |
| 8. Configure dynamic NAT as a policy-map action. |
| host1/C1(config-pmap-c)# nat dynamic 1 vlan 200  
host1/C1(config-pmap-c)# exit  
host1/C1(config-pmap-c)# exit |
| 9. Activate the policy on the client interface using a service policy. If you are operating the ACE in one-arm mode, configure the `service-policy` command on the interface specified in Step 10. |
| host1/C1(config)# interface vlan 100  
host1/C1(config-if)# service-policy input NAT_POLICY  
host1/C1(config-if)# ctrl-z |
| 10. Configure the NAT pool on the server interface. To configure dynamic PAT, include the `pat` keyword in the `nat-pool` command. |
| host1/C1(config)# interface vlan 200  
host1/C1(config-if)# nat-pool 1 172.27.16.10 172.27.16.41 netmask 255.255.255.0 pat  
host1/C1(config-if)# Ctrl-Z |
| 11. (Optional) Save your configuration changes to flash memory. |
| host1/Admin# copy running-config startup-config |
| 12. Display and verify your dynamic NAT and PAT configuration. |
| host1/C1# show running-config class-map  
host1/C1# show running-config policy-map  
host1/C1# show running-config service-policy |
Chapter 5 Configuring Network Address Translation

Configuring Dynamic NAT and PAT

access-list name [line number] extended {deny | permit} {protocol} {src_ip_address netmask | any | host src_ip_address} [operator port1 [port2]] {dest_ip_address netmask | any | host dest_ip_address} [operator port3 [port4]]

For example, enter:

host1/C1(config)# access-list NAT_ACCESS extended permit tcp 192.168.12.0 255.255.255.0 172.27.16.0 255.255.255.0 eq 80

To delete the ACL from the configuration, enter:

host1/C1(config)# no access-list NAT_ACCESS

Configuring Interfaces for Dynamic NAT and PAT

Configure an interface for clients and an interface for the real servers. If you are operating the ACE in one-arm mode, do not configure an interface for clients. For details, see the Cisco Application Control Engine Module Routing and Bridging Configuration Guide.

Creating a Global IP Address Pool for NAT

Dynamic NAT uses a pool of global IP addresses that you specify. You can define either a single global IP address for a group of servers with PAT to differentiate between them, or a range of global IP addresses when using dynamic NAT only. To use a single IP address or a range of addresses, you assign an identifier to the address pool. You configure the NAT pool on the server VLAN interface.

Note

If a packet egresses an interface that you have not configured for NAT, the ACE transmits the packet untranslated.

To create a pool of IP addresses for dynamic NAT, use the nat-pool command in interface configuration mode. The syntax of this command is as follows:

nat-pool pool_id ip_address1 [ip_address2] netmask mask [pat]

The keywords, arguments, and options are as follows:
• **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, 172.27.16.10).

• **ip_address2**—(Optional) Highest IP address in a range of global IP addresses used for NAT. Enter an IP address in dotted-decimal notation (for example, 172.27.16.109). You can configure a maximum of 64 K addresses in a NAT pool.

  If you specify PAT, you can configure a maximum of 32 IP addresses in a NAT pool range. 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`.

  **Note**  
  The ACE allows you to configure a virtual IP (VIP) address in the NAT pool for dynamic NAT and PAT. This action is useful when you want to source NAT real server originated connections (bound to the client) using the VIP address. This feature is specifically useful when there are a limited number of real world IP addresses on the client-side network. To perform PAT for different real servers that are source-NATed to the same IP address (VIP), you must configure the **pat** keyword in the **nat-pool** command.

• **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**—(Optional) Specifies that the ACE perform Port Address Translation (PAT) in addition to NAT.

  If the ACE runs out of IP addresses in a NAT pool, it can switch over to a PAT rule, if configured. For example, you can configure the following:
Chapter 5 Configuring Network Address Translation

Configuring Dynamic NAT and PAT

host1/Admin(config-if)# nat-pool 1 10.1.100.10 10.1.100.99 netmask 255.255.255.255
host1/Admin(config-if)# nat-pool 1 10.1.100.100 10.1.100.100 netmask 255.255.255.255 pat

If your network configuration has the following conditions, you should configure multiple PAT pools with a single IP address in each pool:

- Traffic coming from the same source IP address
- Source ports varying from 1 to 64000
- The same destination port going to different destination addresses
- All ports in one PAT pool are used

So instead of configuring:

host1/Admin(config-if)# nat-pool 1 3.3.3.3 3.3.3.5 netmask 255.255.255.255 pat

configure:

host1/Admin(config-if)# nat-pool 1 192.161.12.3 netmask 255.255.255.255 pat
host1/Admin(config-if)# nat-pool 1 192.161.12.4 netmask 255.255.255.255 pat
host1/Admin(config-if)# nat-pool 1 192.161.12.5 netmask 255.255.255.255 pat

To configure a NAT pool consisting of a range of 32 (the maximum number of IP addresses per PAT pool) global IP addresses with PAT, enter:

host1/C1(config)# interface vlan 200
host1/C1(config-if)# nat-pool 1 172.27.16.10 172.27.16.41 netmask 255.255.255.255 pat

Note

Before you can remove a NAT pool from an interface, you must remove the service policy and the policy map associated with the NAT pool.

To remove a NAT pool from the configuration, enter:

host1/C1(config-if)# no nat-pool 1
Configuring a Class Map

You can configure a traffic class for dynamic NAT and PAT by using the `class-map` command in configuration mode. For more information about class maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
class-map match-any name
```

The `name` argument is a unique identifier for the class map, specified as an unquoted text string with a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config)# class-map match-any NAT_CLASS
host1/C1(config-cmap)#
```

To remove a class-map from the configuration, enter:

```
host1/C1(config)# no class-map match-any NAT_CLASS
```

Enter match criteria for the ACL or the client source address using the `match` command in class-map configuration mode. For example, enter:

```
host1/C1(config-cmap)# match access-list NAT_ACCESS
```

or

```
host1/C1(config-cmap)# match source-address 192.168.12.15 255.255.255.0
```

To remove a match statement from a class map, enter:

```
host1/C1(config-cmap)# no match access-list NAT_ACCESS
```
Configuring a Class Map for Passive FTP

If you are using passive FTP with source NAT, you must configure an additional class map to source NAT the passive data connection. You then associate this class map with the Layer 4 multimatch policy and configure the `nat dynamic` command as an action in the policy map under this class map. To configure a class map for passive FTP, enter the following commands:

```
host1/C1(config)# class-map match-any FTP_NAT_CLASS
host1/C1(config-cmap)# match virtual address 172.16.35.37 any
```

Configuring a Policy Map

You can configure a traffic policy for dynamic NAT and PAT by using the `policy-map` command in configuration mode. For more information about policy maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
policy-map multi-match name
```

The `name` argument is the name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config)# policy-map multi-match NAT_POLICY
host1/C1(config-pmap)#
```

To remove a policy map from the configuration, enter:

```
host1/C1(config)# no policy-map multi-match NAT_POLICY
```

Associate the previously created class map with the policy map. For example, enter:

```
host1/C1(config-pmap)# class NAT_CLASS
host1/C1(config-pmap-c)#
```

To dissociate a class map from a policy map, enter:

```
host1/C1(config-pmap)# no class NAT_CLASS
```

Configure policy-map actions as required. For example, configure:

```
host1/C1(config-pmap-c)# loadbalance policy L7_POLICY
```
host1/C1(config-pmap-c)# **loadbalance VIP inservice**

For passive FTP, associate the FTP_NAT_CLASS class map (see the Configuring a Class Map for Passive FTP section) with the Layer 4 policy map. For example, enter the following commands in policy map configuration mode:

```
host1/C1(config)# **policy-map multi-match NAT_POLICY**
host1/C1(config-pmap)# **class FTP_NAT_CLASS**
```

Proceed with the following section and configure the **nat dynamic** command as a policy action under the FTP class map if you are using passive FTP. Otherwise, configure the **nat dynamic** command as a policy action under the NAT_CLASS class map.

### Configuring Dynamic NAT and PAT as a Layer 3 and Layer 4 Policy-Map Action

You can configure dynamic NAT and PAT (SNAT) as an action in a Layer 3 and Layer 4 policy map by using the **nat dynamic** command in policy-map class configuration mode. The ACE applies dynamic NAT from the interface to which the traffic policy is attached (through the **service-policy** interface configuration command) to the interface specified in the **nat** command. If you are operating in one-arm mode, there is only one VLAN interface.

The syntax of this command is as follows:

```
nat dynamic pool_id vlan number
```

The keywords, arguments, and options are as follows:

- **dynamic pool_id**—Refers to the identifier of a global pool of IP addresses that was configured using the **nat-pool** command on the specified VLAN (see the “Creating a Global IP Address Pool for NAT” section). Dynamic NAT translates a group of local source IP addresses to a pool of global IP addresses that are routable on the destination network. All packets egressing the interface attached to the traffic policy have their source address translated to one of the available addresses in the global pool. Enter an integer from 1 to 2147483647.
- **vlan number**—Specifies the server interface for the global IP address. This interface must be different from the interface that the ACE uses to filter and receive traffic that requires NAT, unless the network design operates in one-arm mode. In that case, the VLAN number is the same.

**Note**

If a packet egresses an interface that you have not configured for NAT, the ACE transmits the packet untranslated.

The following example specifies the `nat` command as an action for a dynamic NAT Layer 3 and Layer 4 policy map:

```
host1/C1(config)# policy-map multi-action NAT_POLICY
host1/C1(config-pmap)# class NAT_CLASS
host1/C1(config-pmap-c)# nat dynamic 1 vlan 200
```

To remove a dynamic NAT action from a policy map, enter:

```
host1/C1(config-pmap-c)# no nat dynamic 1 vlan 200
```

### Applying the Dynamic NAT and PAT Policy Map to an Interface Using a Service Policy

Activate the dynamic NAT and PAT policy map and associate it with an interface by using the `service-policy` command in interface configuration mode. For details about the `service-policy` command, see the *Cisco Application Control Engine Module Administration Guide*.

**Note**

You can configure dynamic NAT as an input service policy only, not as an output service policy. You cannot apply the same NAT policy both locally and globally.

The syntax of this command is as follows:

```
service-policy input policy_name
```
Chapter 5      Configuring Network Address Translation

Configuring Dynamic NAT and PAT

The keywords and arguments are as follows:

- **input**—Specifies that the traffic policy is to be attached to the input direction of a VLAN interface. The traffic policy evaluates all traffic received by that interface.

- **policy_name**—Name of a previously defined policy map. The name can have a maximum of 64 alphanumeric characters.

For example, to apply a service policy to a specific interface, enter:

```
host1/C1(config)# interface vlan 100
host1/C1(config-if)# mtu 1700
host1/C1(config-if)# ip address 192.168.1.100 255.255.255.0
host1/C1(config-if)# service-policy input NAT_POLICY
```

To apply a service policy globally to all interfaces in a context, enter:

```
host1/C1(config)# service-policy input NAT_POLICY
```

To remove a service policy from an interface, enter:

```
host1/C1(config-if)# no service-policy input NAT_POLICY
```

To remove a service policy globally from all interfaces in a context, enter:

```
host1/C1(config)# no service-policy input NAT_POLICY
```

---

**Note**

When you detach a traffic policy either individually from the last VLAN interface on which you applied the service policy or globally from all VLAN interfaces in the same context, the ACE automatically resets the associated service-policy statistics. The ACE performs this action to provide a new starting point for the service-policy statistics the next time that you attach a traffic policy to a specific VLAN interface or globally to all VLAN interfaces in the same context.
Configuring Server Farm-Based Dynamic NAT

This section describes how to configure server farm-based dynamic NAT on an ACE for SNAT. For overview information about server farm-based dynamic NAT, see the “Network Address Translation Overview” section. This section contains the following topics:

- Server Farm-Based Dynamic NAT Configuration Quick Start
- Configuring an ACL for Server Farm-Based Dynamic NAT
- Configuring Interfaces for Server Farm-Based Dynamic NAT
- Creating a Global IP Address Pool for Dynamic NAT
- Configuring Real Servers and a Server Farm
- Configuring a Layer 7 Load-Balancing Class Map for Server Farm-Based Dynamic NAT
- Configuring a Layer 7 Load-Balancing Policy Map for Server Farm-Based Dynamic NAT
- Configuring Server Farm-Based Dynamic NAT as a Layer 7 Policy Action
- Configuring a Layer 3 and Layer 4 Class Map for Server Farm-Based Dynamic NAT
- Configuring a Layer 3 and Layer 4 Policy Map for Server Farm-Based Dynamic NAT
- Applying the Layer 3 and Layer 4 Policy Map to an Interface Using a Service Policy

Server Farm-Based Dynamic NAT Configuration Quick Start

Table 5-2 provides a quick overview of the steps required to configure server farm-based dynamic NAT. Each step includes the CLI command or a reference to the procedure required to complete the task. For a complete description of each feature and all the options associated with the CLI commands, see the sections following Table 5-2.
Table 5-2  Server Farm-Based Dynamic NAT Configuration Quick Start

Task and Command Example

1. If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, change to the correct context.

   host1/Admin# changeto C1
   host1/C1#

   The rest of the examples in this table use the C1 user context, unless otherwise specified. For details on creating contexts, see the Cisco Application Control Engine Module Virtualization Configuration Guide.

2. Enter configuration mode.

   host1/C1# config
   host1/C1(config)#

3. Configure an ACL to allow traffic that requires NAT.

   host1/C1(config)# access-list ACL1 line 10 extended permit tcp 10.0.0.0 255.0.0.0 eq 8080 any
   host1/C1(config-acl)# exit

4. Configure real servers and a server farm for load balancing. The nat dynamic command in Step 9 references this server farm.

   host1/C1(config)# rserver SERVER1
   host1/C1(config-rserver-host)# ip address 172.27.16.201
   host1/C1(config-rserver-host)# active
   host1/C1(config-rserver-host)# exit
   host1/C1(config)# rserver SERVER2
   host1/C1(config-rserver-host)# ip address 172.27.16.202
   host1/C1(config-rserver-host)# active
   host1/C1(config-rserver-host)# exit
   host1/C1(config)# serverfarm SF1
   host1/C1(config-sfarm-host)# rserver SERVER1 3000
   host1/C1(config-sfarm-host-rs)# active
   host1/C1(config-sfarm-host-rs)# exit
   host1/C1(config-sfarm-host)# rserver SERVER2 3001
   host1/C1(config-sfarm-host-rs)# active
   host1/C1(config-sfarm-host-rs)# exit
   host1/C1(config-sfarm-host)# exit
5. Configure a local interface (client VLAN) to filter and receive client traffic. If you are operating the ACE in one-arm mode, omit this step.

   host1/C1(config)# interface vlan 100
   host1/C1(config-if)# mtu 1500
   host1/C1(config-if)# ip address 192.168.12.100 255.255.255.0
   host1/C1(config-if)# no shutdown
   host1/C1(config-if)# exit

6. Configure a second interface (server VLAN) for the NAT pool.

   host1/C1(config)# interface vlan 200
   host1/C1(config-if)# mtu 1500
   host1/C1(config-if)# ip address 172.27.16.200 255.255.255.0
   host1/C1(config-if)# no shutdown
   host1/C1(config-if)# exit

7. Configure a Layer 7 load-balancing class map and define match criteria.

   host1/C1(config)# class-map type http loadbalance match-any
                      L7_CLASS
   host1/C1(config-cmap-http-lb)# match http content .*cisco.com

8. Configure a Layer 7 load-balancing policy map and associate the class map with the policy map.

   host1/C1(config)# policy-map type loadbalance http first-match
                    L7_POLICY
   host1/C1(config-pmap-lb)# class L7_CLASS
   host1/C1(config-pmap-lb-c)#

9. Configure server farm-based dynamic NAT as a policy-map action in the Layer 7 load-balancing policy. You can configure multiple instances of this command for each primary and backup serverfarm and each outgoing server VLAN.

   host1/C1(config-pmap-lb-c)# nat dynamic 1 vlan 200 serverfarm
                                primary
   host1/C1(config-pmap-lb-c)# exit
   host1/C1(config-pmap-lb)# exit
   host1/C1(config)#
**Chapter 5  Configuring Network Address Translation**

**Configuring Server Farm-Based Dynamic NAT**

10. Configure a Layer 3 and Layer 4 class map and define match criteria.

    host1/C1(config)# class-map match-any SLB_CLASS
    host1/C1(config-cmap)# match virtual-address 172.16.27.52 tcp eq http
    host1/C1(config-cmap)# exit

11. Configure a Layer 3 and Layer 4 policy map and associate the class map with the policy map.

    host1/C1(config)# policy-map multi-match SLB_POLICY
    host1/C1(config-pmap)# class SLB_CLASS
    host1/C1(config-pmap-c)#

12. Configure Layer 3 and Layer 4 policy map actions.

    host1/C1(config-pmap-c)# loadbalance policy L7_POLICY
    host1/C1(config-pmap-c)# loadbalance vip inservice
    host1/C1(config-pmap-c)# exit
    host1/C1(config-pmap)# exit
    host1/C1(config)#

13. Activate the policy on the client interface using a service policy. If you are operating the ACE in one-arm mode, configure the `service-policy` command on the interface specified in Step 14.

    host1/C1(config)# interface vlan 100
    host1/C1(config-if)# service-policy input SLB_POLICY
    host1/C1(config-if)# exit

14. Configure the NAT pool on the server interface.

    host1/C1(config)# interface vlan 200
    host1/C1(config-if)# nat-pool 1 172.27.16.10 172.27.26.49 255.255.255.0
    host1/C1(config-if)# Ctrl-Z

15. (Optional) Save your configuration changes to flash memory.

    host1/Admin# copy running-config startup-config

16. Display and verify your server farm-based dynamic NAT configuration.

    host1/C1# show running-config class-map
    host1/C1# show running-config policy-map
    host1/C1# show running-config service-policy

**Table 5-2  Server Farm-Based Dynamic NAT Configuration Quick Start (continued)**

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>10. Configure a Layer 3 and Layer 4 class map and define match criteria.</td>
</tr>
<tr>
<td>host1/C1(config)# class-map match-any SLB_CLASS</td>
</tr>
<tr>
<td>host1/C1(config-cmap)# match virtual-address 172.16.27.52 tcp eq http</td>
</tr>
<tr>
<td>host1/C1(config-cmap)# exit</td>
</tr>
<tr>
<td>11. Configure a Layer 3 and Layer 4 policy map and associate the class map with the policy map.</td>
</tr>
<tr>
<td>host1/C1(config)# policy-map multi-match SLB_POLICY</td>
</tr>
<tr>
<td>host1/C1(config-pmap)# class SLB_CLASS</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)#</td>
</tr>
<tr>
<td>12. Configure Layer 3 and Layer 4 policy map actions.</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)# loadbalance policy L7_POLICY</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)# loadbalance vip inservice</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)# exit</td>
</tr>
<tr>
<td>host1/C1(config-pmap)# exit</td>
</tr>
<tr>
<td>host1/C1(config)#</td>
</tr>
<tr>
<td>13. Activate the policy on the client interface using a service policy. If you are operating the ACE in one-arm mode, configure the <code>service-policy</code> command on the interface specified in Step 14.</td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 100</td>
</tr>
<tr>
<td>host1/C1(config-if)# service-policy input SLB_POLICY</td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
</tr>
<tr>
<td>14. Configure the NAT pool on the server interface.</td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 200</td>
</tr>
<tr>
<td>host1/C1(config-if)# nat-pool 1 172.27.16.10 172.27.26.49 255.255.255.0</td>
</tr>
<tr>
<td>host1/C1(config-if)# Ctrl-Z</td>
</tr>
<tr>
<td>15. (Optional) Save your configuration changes to flash memory.</td>
</tr>
<tr>
<td>host1/Admin# copy running-config startup-config</td>
</tr>
<tr>
<td>16. Display and verify your server farm-based dynamic NAT configuration.</td>
</tr>
<tr>
<td>host1/C1# show running-config class-map</td>
</tr>
<tr>
<td>host1/C1# show running-config policy-map</td>
</tr>
<tr>
<td>host1/C1# show running-config service-policy</td>
</tr>
</tbody>
</table>
Configuring an ACL for Server Farm-Based Dynamic NAT

Use an access control list (ACL) to permit the traffic that requires NAT. For details about configuring an ACL, see Chapter 1, Configuring Security Access Control Lists.

To configure an ACL for dynamic NAT, use the `access-list` command in configuration mode. The syntax of this command is as follows:

```
access-list name [line number] extended {deny | permit} {protocol} {src_ip_address netmask | any | host src_ip_address} [operator port1 [port2]] {dest_ip_address netmask | any | host dest_ip_address} [operator port3 [port4]]
```

For example, enter:

```
host1/C1(config)# access-list NAT_ACCESS extended permit tcp 192.168.12.0 255.255.255.0 172.27.16.0 255.255.255.0 eq 80
```

To delete the ACL from the configuration, enter:

```
host1/C1(config)# no access-list nat_access
```

Configuring Interfaces for Server Farm-Based Dynamic NAT

Configure an interface for clients and an interface for the real servers. If you are operating the ACE in one-arm mode, omit the client interface. For details about configuring interfaces, see the Cisco Application Control Engine Module Routing and Bridging Configuration Guide.

Creating a Global IP Address Pool for Dynamic NAT

Dynamic NAT uses a pool of global IP addresses that you specify. You can define a range of global IP addresses when using dynamic NAT. To use a range of addresses, you assign an identifier to the address pool. You then associate the NAT pool with the server VLAN interface.

*Note*

If a packet egresses an interface that you have not configured for NAT, the ACE transmits the packet untranslated.
To create a pool of IP addresses for dynamic NAT, use the `nat-pool` command in interface configuration mode. The syntax of this command is as follows:

```
nat-pool pool_id ip_address1 ip_address2 netmask mask
```

The keywords, arguments, and options are as follows:

- **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, 172.27.16.10).

- **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, 172.27.16.26). You can configure a maximum of 64 K addresses in a NAT pool.

  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`.

  **Note** The ACE allows you to configure a virtual IP (VIP) address in the NAT pool for dynamic NAT. This action is useful when you want to source NAT real server originated connections (bound to the client) using the VIP address. This feature is specifically useful when there are a limited number of real world IP addresses on the client-side network.

- **netmask**—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.

To configure a NAT pool consisting of a range of 32 global IP addresses, enter:

```
host1/C1(config)# interface vlan 200
```
host1/C1(config-if)# nat-pool 1 172.27.16.10 172.27.16.41 netmask 255.255.255.255

Note
Before you can remove a NAT pool from an interface, you must remove the service policy and the policy map associated with the NAT pool.

To remove a NAT pool from the configuration, enter:
host1/C1(config-if)# no nat-pool 1

Configuring Real Servers and a Server Farm

For details about configuring real servers and server farms, see the Cisco Application Control Engine Module Server Load-Balancing Configuration Guide.

Configuring a Layer 7 Load-Balancing Class Map for Server Farm-Based Dynamic NAT

Configure a Layer 7 traffic class for server farm-based dynamic NAT by using the class-map command in configuration mode. The syntax of this command is as follows:

```
class-map type http loadbalance match-any name
```

The name argument is a unique identifier for the class map, specified as an unquoted text string with a maximum of 64 alphanumeric characters.

For example, enter:
host1/C1(config)# class-map type http loadbalance match-any L7_CLASS
host1/C1(config-cmap-http-lb)#

To remove a class-map from the configuration, enter:
host1/C1(config)# no class-map type http loadbalance match-any L7_CLASS

Enter match criteria as required using the match command in class-map load balancing configuration mode. For example, enter:
host1/C1(config-cmap-http-lb)# match http content .*cisco.com
To remove a match statement from a class map, enter:

`host1/C1(config-cmap-http-lb)# no match http content .*cisco.com`

### Configuring a Layer 7 Load-Balancing Policy Map for Server Farm-Based Dynamic NAT

Configure a Layer 7 load-balancing policy map by using the `policy-map` command in configuration mode. The syntax of this command is:

```
policy-map type loadbalance http first-match name
```

The `name` argument is a unique identifier for the policy map, specified as an unquoted text string with a maximum of 64 alphanumeric characters. For example, enter:

```
host1/C1(config)# policy-map type loadbalance http first-match L7_POLICY
host1/C1(config-pmap-lb)#
```

To remove a policy map from the configuration, enter:

```
host1/C1(config)# no policy-map multi-match NAT_POLICY
```

To associate the previously created class map with the policy map. For example, enter:

```
host1/C1(config-pmap-lb)# class L7_CLASS
host1/C1(config-pmap-lb-c)#
```

To disassociate a class map from a policy map, enter:

```
host1/C1(config-pmap-lb)# no class L7_CLASS
```
Configuring Server Farm-Based Dynamic NAT as a Layer 7 Policy Action

Configure server farm-based dynamic NAT as an action in a Layer 7 load-balancing policy map by using the `nat` command in policy-map load-balancing class configuration mode. Typically, you use dynamic NAT for SNAT. Dynamic NAT allows you to identify local traffic for address translation by specifying the source and destination addresses in an extended ACL, which is referenced as part of the class map traffic classification. The ACE applies dynamic NAT from the interface to which the traffic policy is attached (through the `service-policy` interface configuration command) to the interface specified in the `nat dynamic` command.

The syntax of this command is as follows:

```
nat dynamic pool_id vlan number serverfarm {primary | backup}
```

The keywords and arguments are as follows:

- `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.

- `vlan number`—Specifies the server interface for the global IP address. This interface must be different from the interface that the ACE uses to filter and receive traffic that requires NAT, unless the network design operates in one-arm mode. In that case, the VLAN number is the same.

- `serverfarm`—Specifies server farm-based dynamic NAT.

- `primary | backup`—Specifies that the dynamic NAT applies to either the primary server farm or the backup server farm.

  **Note** If a packet egresses an interface that you have not configured for NAT, the ACE transmits the packet untranslated.
The following SNAT server farm-based dynamic NAT example specifies the `nat` command as an action for a Layer 7 policy map:

```
host1/C1(config)# policy-map type loadbalance http first-match L7_POLICY
host1/C1(config-pmap-lb)# class L7_CLASS
host1/C1(config-pmap-lb-c)# nat dynamic serverfarm primary 1 vlan 200
```

To remove a server farm-based dynamic NAT action from a policy map, enter:

```
host1/C1(config-pmap-lb-c)# no nat dynamic serverfarm primary 1 vlan 200
```

### Configuring a Layer 3 and Layer 4 Class Map for Server Farm-Based Dynamic NAT

Configure a Layer 3 and Layer 4 traffic class for server farm-based dynamic NAT by using the `class-map` command in configuration mode. For more information about class maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
class-map match-any name
```

The `name` argument is a unique identifier for the class map, specified as an unquoted text string with a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config)# class-map match-any NAT_CLASS
host1/C1(config-cmap)#
```

To remove a class map from the configuration, enter:

```
host1/C1(config)# no class-map match-any NAT_CLASS
```

Enter match criteria as required using the `match` command in class-map configuration mode. For example, enter:

```
host1/C1(config-cmap)# match access-list NAT_ACCESS
```

or

```
host1/C1(config-cmap)# match source address 192.168.12.15
```
To remove a match statement from a class map, enter:

```
host1/C1(config-cmap)# no match access-list NAT_ACCESS
```

## Configuring a Layer 3 and Layer 4 Policy Map for Server Farm-Based Dynamic NAT

Configure a Layer 3 and Layer 4 traffic policy for NAT by using the `policy-map` command in configuration mode. For more information about policy maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
policy-map multi-match name
```

The `name` argument is the name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config)# policy-map multi-match NAT_POLICY
host1/C1(config-pmap)#
```

To remove a policy map from the configuration, enter:

```
host1/C1(config)# no policy-map multi-match NAT_POLICY
```

To associate the previously created class map with the policy map. For example, enter:

```
host1/C1(config-pmap)# class NAT_CLASS
host1/C1(config-pmap-c)#
```

To dissociate a class map from a policy map, enter:

```
host1/C1(config-pmap)# no class NAT_CLASS
```

Configure policy-map actions as required. For example, configure:

```
host1/C1(config-pmap-c)# loadbalance policy L7_POLICY
host1/C1(config-pmap-c)# loadbalance VIP inservice
```
Applying the Layer 3 and Layer 4 Policy Map to an Interface Using a Service Policy

You can activate the server farm-based dynamic NAT policy and assign it to an interface by using the `service-policy` command in interface configuration mode. For details about the `service-policy` command, see the Cisco Application Control Engine Module Administration Guide.

**Note**
You can configure dynamic NAT as an input service policy only, not as an output service policy. You cannot apply the same NAT policy both locally and globally.

The syntax of this command is as follows:

```
service-policy input policy_name
```

The keywords and arguments are as follows:

- **input**—Specifies that the traffic policy is to be attached to the input direction of a VLAN interface. The traffic policy evaluates all traffic received by that interface.
- **policy_name**—Name of a previously defined policy map. The name can have a maximum of 64 alphanumeric characters.

For example, enter:
```
host1/C1(config)# interface vlan 100
host1/C1(config-if)# mtu 1700
host1/C1(config-if)# ip address 192.168.12.100 255.255.255.0
host1/C1(config-if)# service-policy input NAT_POLICY
```

To remove a service policy from an interface, enter:
```
host1/C1(config-if)# no service-policy input NAT_POLICY
```

**Note**
When you remove a traffic policy from the last VLAN interface on which you applied the service policy, the ACE automatically resets the associated service-policy statistics. The ACE performs this action to provide a new starting point for the service-policy statistics the next time that you attach a traffic policy to a specific VLAN interface.
Configuring Static NAT and Static Port Redirection

This section describes how to configure static NAT and static port redirection on an ACE for DNAT. For overview information about static NAT and static port redirection, see the “Network Address Translation Overview” section. This section contains the following topics:

- Static NAT Configuration Quick Start
- Configuring an ACL for Static NAT and Static Port Redirection
- Configuring a Class Map
- Configuring a Policy Map
- Configuring Static NAT and Static Port Redirection as a Policy Action
- Applying the Static NAT and Static Port Redirection Policy Map to an Interface Using a Service Policy

Static NAT Configuration Quick Start

Table 5-3 provides a quick overview of the steps required to configure static NAT and static port redirection. Each step includes the CLI command or a reference to the procedure required to complete the task. For a complete description of each feature and all the options associated with the CLI commands, see the sections following Table 5-3.
Configuring Static NAT and Static Port Redirection

**Table 5-3**  
**Static NAT Configuration Quick Start**

<table>
<thead>
<tr>
<th>Task and Command Example</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong> If you are operating in multiple contexts, observe the CLI prompt to verify that you are operating in the desired context. If necessary, change to the correct context.</td>
<td></td>
</tr>
<tr>
<td>host1/Admin# changeto C1</td>
<td></td>
</tr>
<tr>
<td>host1/C1#</td>
<td></td>
</tr>
<tr>
<td>The rest of the examples in this table use the C1 user context, unless otherwise specified. For details on creating contexts, see the <em>Cisco Application Control Engine Module Virtualization Configuration Guide.</em></td>
<td></td>
</tr>
<tr>
<td><strong>2.</strong> Enter configuration mode.</td>
<td></td>
</tr>
<tr>
<td>host1/C1# config</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config)#</td>
<td></td>
</tr>
<tr>
<td><strong>3.</strong> Configure an ACL to allow traffic that requires NAT.</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config)# access-list ACL1 line 10 extended permit tcp 10.0.0.0 255.0.0.0 eq 8080 any</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-acl)# exit</td>
<td></td>
</tr>
<tr>
<td><strong>4.</strong> Configure a local interface to filter and receive traffic that requires NAT.</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 100</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# mtu 1500</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# ip address 192.168.1.100 255.255.255.0</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# no shutdown</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
<td></td>
</tr>
<tr>
<td><strong>5.</strong> Configure a second interface (global interface) for performing NAT.</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config)# interface vlan 101</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# mtu 1500</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# ip address 172.27.16.100 255.255.255.0</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# no shutdown</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-if)# exit</td>
<td></td>
</tr>
<tr>
<td><strong>6.</strong> Configure a class map and define match criteria.</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config)# class-map match-any NAT_CLASS</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-cmap)# match access-list ACL1</td>
<td></td>
</tr>
<tr>
<td>host1/C1(config-cmap)# exit</td>
<td></td>
</tr>
</tbody>
</table>
Table 5-3  Static NAT Configuration Quick Start (continued)

<table>
<thead>
<tr>
<th>Task and Command Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. Configure a policy map and associate the class map with the policy map.</td>
</tr>
<tr>
<td>host1/C1(config)# policy-map multi-match NAT_POLICY</td>
</tr>
<tr>
<td>host1/C1(config-pmap)# class NAT_CLASS</td>
</tr>
<tr>
<td>host1/C1(config-pmap-c)#</td>
</tr>
</tbody>
</table>

| 8. Configure static NAT as a policy-map action. |
| host1/C1(config-pmap-c)# nat static 192.0.0.0 netmask 255.0.0.0 vlan 101 |
| host1/C1(config-pmap-c)# exit  |
| host1/C1(config-pmap-c)# exit  |
| host1/C1(config)# |

| 9. Activate the policy on an interface using a service policy. |
| host1/C1(config)# interface vlan 100 |
| host1/C1(config-if)# service-policy input NAT_POLICY |
| host1/C1(config-if)# Ctrl-Z |

| 10. (Optional) Save your configuration changes to flash memory. |
| host1/Admin# copy running-config startup-config |

| 11. Display and verify your static NAT and static port redirection configuration. |
| host1/C1# show running-config class-map |
| host1/C1# show running-config policy-map |

Configuring an ACL for Static NAT and Static Port Redirection

Use an access control list (ACL) to permit the traffic that requires static NAT and port redirection. For details about configuring an ACL, see Chapter 1, Configuring Security Access Control Lists.

To configure an ACL for static NAT, use the access-list command in configuration mode. The syntax of this command is as follows:

```
access-list name [line number] extended {deny | permit}  
  {protocol} {src_ip_address netmask | any | host src_ip_address}  
  [operator port1 [port2]] {dest_ip_address netmask | any | host dest_ip_address} [operator port3 [port4]]
```
For example, enter:

```
host1/C1(config)# access-list acl1 line 10 extended permit tcp
    10.0.0.0 255.0.0.0 eq 8080 any
```

To delete the ACL from the configuration, enter:

```
host1/C1(config)# no access-list nat_access
```

### Configuring Interfaces for Static NAT and Static Port Redirection

Configure an interface for clients and an interface for the real servers. For details, see the *Cisco Application Control Engine Module Routing and Bridging Configuration Guide*.

### Configuring a Class Map

You can configure a traffic class for static NAT and port redirection by using the `class-map` command in configuration mode. For more information about class maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
class-map match-any name
```

The `name` argument is a unique identifier for the class map, specified as an unquoted text string with a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config)# class-map match-any NAT_CLASS
host1/C1(config-cmap)#
```

To remove a class-map from the configuration, enter:

```
host1/C1(config)# no class-map match-any NAT_CLASS
```
Enter match criteria as required using the `match` command in class-map configuration mode. For example, enter:

```
host1/C1(config-cmap)# match access-list NAT_ACCESS
```

or

```
host1/C1(config-cmap)# match source address 192.168.12.15
```

To remove a match statement from a class map, enter:

```
host1/C1(config-cmap)# no match access-list NAT_ACCESS
```

### Configuring a Policy Map

You can configure a traffic policy for NAT by using the `policy-map` command in configuration mode. For more information about policy maps, see the *Cisco Application Control Engine Module Administration Guide*.

The syntax of this command is as follows:

```
policy-map multi-match name
```

The `name` argument is the name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.

For example, enter:

```
host1/C1(config)# policy-map multi-match NAT_POLICY
host1/C1(config-pmap)#
```

To remove a policy map from the configuration, enter:

```
host1/C1(config)# no policy-map multi-match NAT_POLICY
```

To associate the previously created class map with the policy map. For example, enter:

```
host1/C1(config-pmap)# class NAT_CLASS
host1/C1(config-pmap-c)#
```

To dissociate a class map from a policy map, enter:

```
host1/C1(config-pmap)# no class NAT_CLASS
```
Configuring Static NAT and Static Port Redirection as a Policy Action

You can configure static NAT and static port redirection as an action in a policy map by using the `nat static` command in policy-map class configuration mode. Typically, you use static NAT and port redirection for DNAT. Static NAT allows you to identify local traffic for address translation by specifying the source and destination addresses in an extended ACL, which is referenced as part of the class map traffic classification. The ACE applies static NAT from the interface to which the traffic policy is attached (through the `service-policy` interface configuration command) to the interface specified in the `nat static` command.

The syntax of this command is as follows:

```
   nat static ip_address netmask mask {port1 | tcp eq port2 | udp eq port3} vlan number
```

The keywords and arguments are as follows:

- **static ip_address**—Sets up a single static translation. The `ip_address` argument establishes the globally unique IP address of a host as it appears to the outside world. The policy map performs the global IP address translation for the source IP address specified in the ACL (as part of the class-map traffic classification).

- **netmask mask**—Specifies the subnet mask for the static IP address. Enter a subnet mask in dotted-decimal notation (for example, 255.255.255.0).

- **port1**—Global TCP or UDP port for static port redirection. Enter an integer from 0 to 65535.

- **tcp eq port2**—Specifies a TCP port name or number. Enter an integer from 0 to 65535. A value of 0 instructs the ACE to match any port. Alternatively, you can enter a protocol keyword that corresponds to a TCP port number. See Table 5-4 for a list of supported well-known TCP port names and numbers.
Table 5-4  Well-Known TCP Port Numbers and Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftp</td>
<td>21</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>http</td>
<td>80</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>https</td>
<td>443</td>
<td>HTTP over TLS/SSL</td>
</tr>
<tr>
<td>irc</td>
<td>194</td>
<td>Internet Relay Chat</td>
</tr>
<tr>
<td>matip-a</td>
<td>350</td>
<td>Mapping of Airline Traffic over Internet Protocol (MATIP) Type A</td>
</tr>
<tr>
<td>nntp</td>
<td>119</td>
<td>Network News Transport Protocol</td>
</tr>
<tr>
<td>pop2</td>
<td>109</td>
<td>Post Office Protocol v2</td>
</tr>
<tr>
<td>pop3</td>
<td>110</td>
<td>Post Office Protocol v3</td>
</tr>
<tr>
<td>rtsp</td>
<td>554</td>
<td>Real Time Streaming Protocol</td>
</tr>
<tr>
<td>smtp</td>
<td>25</td>
<td>Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>Telnet</td>
</tr>
</tbody>
</table>

- **udp eq port3**—Specifies a UDP port name or number. Enter an integer from 0 to 65535. A value of 0 instructs the ACE to match any port. Alternatively, you can enter a protocol keyword that corresponds to a UDP port number. See Table 5-5 for a list of supported well-known UDP port names and numbers.

Table 5-5  Well-Known UDP Port Numbers and Keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dns</td>
<td>53</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>wsp</td>
<td>9200</td>
<td>Connectionless Wireless Session Protocol (WSP)</td>
</tr>
<tr>
<td>wsp-wtls</td>
<td>9202</td>
<td>Secure Connectionless WSP</td>
</tr>
<tr>
<td>wsp-wtp</td>
<td>9201</td>
<td>Connection-based WSP</td>
</tr>
<tr>
<td>wsp-wtp-wtls</td>
<td>9203</td>
<td>Secure Connection-based WSP</td>
</tr>
</tbody>
</table>

- **vlan number**—Specifies the interface for the global IP address.
If a packet egresses an interface that you have not configured for NAT, the ACE transmits the packet untranslated.

The following DNAT static port redirection example specifies the `nat static` command as an action for a static NAT policy map:

```
host1/C1(config)# policy-map multi-action NAT_POLICY
host1/C1(config-pmap)# class NAT_CLASS
host1/C1(config-pmap-c)# nat static 192.168.12.0 255.255.255.0 80 vlan 101
```

To remove a NAT action from a policy map, enter:

```
host1/C1(config-pmap-c) no nat static 192.168.12.15 255.255.255.0 vlan 200
```

### Applying the Static NAT and Static Port Redirection Policy Map to an Interface Using a Service Policy

You can activate the static NAT and port redirection policy and assign it to an interface by using the `service-policy` command in interface configuration mode. For details about the `service-policy` command, see the *Cisco Application Control Engine Module Administration Guide*.

**Note** You can configure static NAT as an input service policy only; you cannot configure it as an output service policy.

The syntax of this command is as follows:

```
service-policy input policy_name
```

The keywords and arguments are as follows:

- **input**—Specifies that the traffic policy is to be attached to the input direction of a VLAN interface. The traffic policy evaluates all traffic received by that interface.

- **policy_name**—Name of a previously defined policy map. The name can have a maximum of 64 alphanumeric characters.
For example, enter:

```plaintext
host1/C1(config)# interface vlan 100
host1/C1(config-if)# mtu 1700
host1/C1(config-if)# ip address 192.168.1.100 255.255.255.0
host1/C1(config-if)# service-policy input NAT_POLICY
```

To remove a service policy from an interface, enter:

```plaintext
host1/C1(config-if)# no service-policy input NAT_POLICY
```

**Note**

When you remove a traffic policy from the last VLAN interface on which you applied the service policy, the ACE automatically resets the associated service-policy statistics. The ACE performs this action to provide a new starting point for the service-policy statistics the next time that you attach a traffic policy to a specific VLAN interface.

## Configuring Static NAT Overwrite

By default, the ACE allows you to configure a maximum of 8,192 (8 K) static NAT configurations. However, the static NAT overwrite feature allows a maximum number of 400 K NATs. The ACE creates static connections that contain the NATs as soon as the configuration is applied. Because these connections exist before the packets are received, no ACL is required to permit flows that will be translated.

To configure static NAT overwrite, use the `static` command in configuration mode. The syntax of this command is as follows:

```plaintext
static vlan mapped_vlan_id vlan real_vlan_id mapped_ip_address
    {real_ip_address [netmask mask]}
```

The keywords and arguments are:

- `mapped_vlan_id`—VLAN ID of the interface connected to the mapped IP address network. You cannot map the same real interface to different mapped interfaces.

- `real_vlan_id`—VLAN ID of the interface connected to the real IP address network.

- `mapped_ip_address`—Translated IP address for the real address. Enter an IP address in dotted-decimal notation (for example, 172.27.16.10).
• **real_ip_address**—Real server IP address for translation. Enter an IP address in dotted-decimal notation (for example, 172.27.16.10). In a context, you must configure a different address for configurations that have the same real interface.

• **netmask mask**—(Optional) Specifies the subnet mask for the real address. Enter a subnet mask in dotted-decimal notation (for example, 255.255.255.0). The smallest netmask is /24.

When using this command, consider the following restrictions:

• The ACE supports this configuration only in routed mode.

• The ACE allows only one mapped interface in a context. However, each static NAT configuration must have a different mapped IP address.

• The ACE does not support bidirectional NAT, which is source address and destination address translation for the same flow.

• You must limit the number of real IP addresses on the same subnet as the real interface to less than 1 K. Also, limit the number of mapped IP addresses on the same subnet as the mapped interface to less than 1 K.

• The real interface has no restrictions. The ACE supports Equal Cost Multipath Protocol (ECMP) routes on the real interface.

• You must not configure more than one next-hop at any point on the mapped interface. If the ACE detects more than one next hop on the mapped interface, it rejects the `static` command. But if you configure the static route after configuring the `static` command, the ACE does not detect the multiple hops.

• More than one static route can use the same next hop on the mapped interface. The ACE uses the encap-id to forward traffic out of the mapped interface.

• Although multiple public-side next hops are not allowed (see Figure 5-1), multiple private-side next hops are allowed (see Figure 5-2).

---

![Figure 5-1](image)

*Figure 5-1    Not Allowed: Multiple Public-Side Next Hops*
Figure 5-2  Allowed: Multiple Private-Side Next Hops

We do not recommend that you configure MPC-based NAT for the same context in which you configure the static command.

When the ACE performs connection lookup, it performs it in the following order:

- Full tuple connection lookup. This lookup includes the source IP, source port, destination IP, destination port and interface ID.
- Static NAT connection lookup.

If there are existing connections before you configure the static command, the ACE forwards packets based on the former connection records. After these connections time out or are removed, the static NAT connection lookup determines the forwarding of all traffic.

For example, to create a static NAT configuration for the mapped interface VLAN 176, real interface VLAN 171, and real IP address of 10.181.0.2 255.255.255.255 to be translated to the mapped address 5.6.7.4, enter:

```
host1/C1(config)# static vlan 176 vlan 171 5.6.7.4 10.81.0.2 netmask 255.255.255.255
```

To remove this configuration, enter:

```
host1/C1(config)# no static vlan 176 vlan 171 5.6.7.4 10.81.0.2 netmask 255.255.255.255
```

To display static NAT information, use the show nat-fabric global-static command.
Displaying NAT Configurations and Statistics

The following sections describe the commands used to display dynamic and static NAT and PAT configurations and statistics:

- Displaying NAT and PAT Configurations
- Displaying IP Address and Port Translations

Displaying NAT and PAT Configurations

You can display NAT and PAT configurations by using the `show running-config class-map` and `show running-config policy-map` commands in Exec mode.

For example, enter:

```
host1/C1# show running-config class-map
host1/C1# show running-config policy-map
```

Displaying IP Address and Port Translations

You can display IP address and port translation (Xlate) information by using the `show xlate` command in Exec mode. The syntax of this command is as follows:

```
show xlate [global {ip_address1 [ip_address2 [netmask mask1]]}] [local {ip_address3 [ip_address4 [netmask mask2]]}] [gport port1 [port2]] [lport port1 [port2]]
```

The keywords, arguments, and options are as follows:

- **global ip_address1 ip_address2**—(Optional) Displays information for a global IP address or range of global IP addresses to which the ACE translates source addresses for static and dynamic NAT. For a single global IP address, enter the address in dotted-decimal notation (for example, 192.168.12.15). To specify a range of IP addresses, enter a second IP address.
- **netmask mask**—(Optional) Displays the subnet mask for the specified IP addresses.
Displaying NAT Configurations and Statistics

- **local ip_address3 ip_address4**—(Optional) Displays the local IP address or range of local IP addresses. For a single local IP address, enter the address in dotted-decimal notation (for example, 192.168.12.15). To specify a range of local IP addresses, enter a second IP address.

- **gport port1 port2**—(Optional) Displays information for a global port or a range of global ports to which the ACE translates source ports for static port redirection and dynamic PAT, respectively. Enter a port number as an integer from 0 to 65535. To specify a range of port numbers, enter a second port number.

- **lport port3 port4**—(Optional) Displays information for a local port or a range of local ports. Enter a port number as an integer from 0 to 65535. To specify a range of port numbers, enter a second port number.

For example, enter:

```
host1/Admin# show xlate global 172.27.16.3 172.27.16.10 netmask 255.255.255.0 gport 100 200
```

You can also use the **show conn** command to display NAT information. See the examples in the following sections.

This section contains the following topics:

- Dynamic NAT Example
- Dynamic PAT Example
- Static NAT Example
- Static Port Redirection (Static PAT) Example

### Dynamic NAT Example

The following example output of the **show xlate** command shows dynamic NAT (SNAT in this example). When a user uses Telnet from 172.27.16.5 in VLAN 2020, the ACE translates it to 192.168.100.1 in VLAN 2021.

```
host1/Admin# show xlate global 192.168.100.1 192.168.100.10
NAT from vlan2020:172.27.16.5 to vlan2021:192.168.100.1 count:1
```
Dynamic PAT Example

The following example shows dynamic PAT. When a user uses Telnet from 172.27.16.5 in VLAN 2020, the ACE translates it to 192.168.201.1 in VLAN 2021.

host1/Admin# show xlate
TCP PAT from vlan200:172.27.16.5/38097 to vlan2021:192.168.201.1/1025
**Static NAT Example**

The following example shows static NAT. The ACE maps a real IP address (172.27.16.5) to 192.168.210.1.

```
host1/Admin# show xlate
NAT from vlan2020:172.27.16.5 to vlan2021:192.168.210.1 count:1
```

```
host1/Admin# show conn

total current connections : 2

<table>
<thead>
<tr>
<th>conn-id</th>
<th>dir</th>
<th>prot</th>
<th>vlan</th>
<th>source</th>
<th>destination</th>
<th>state</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>in</td>
<td>TCP</td>
<td>2020</td>
<td>172.27.16.5</td>
<td>192.168.100.1</td>
<td>ESTAB</td>
</tr>
<tr>
<td>6</td>
<td>out</td>
<td>TCP</td>
<td>2021</td>
<td>192.168.100.1</td>
<td>192.168.210.1</td>
<td>ESTAB</td>
</tr>
</tbody>
</table>
```

**Static Port Redirection (Static PAT) Example**

The following example shows static port redirection (DNAT in this example). A host at 192.168.0.10:37766 uses Telnet to connect to 192.168.211.1:3030 on VLAN 2021 on the ACE. The ACE maps 172.27.0.5:23 on VLAN 2020 to 192.168.211.1:3030 on VLAN 2021.

```
host1/Admin# show xlate
TCP PAT from vlan2020:172.27.0.5/23 to vlan2021:192.168.211.1/3030
```

```
Mar 24 2006 20:05:41 : %ACE-7-111009: User 'admin' executed cmd: show xlate
```

```
host1/Admin# show conn

total current connections : 2

<table>
<thead>
<tr>
<th>conn-id</th>
<th>dir</th>
<th>prot</th>
<th>vlan</th>
<th>source</th>
<th>destination</th>
<th>state</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>in</td>
<td>TCP</td>
<td>2021</td>
<td>192.168.10:37766</td>
<td>192.168.211.1:3030</td>
<td>ESTAB</td>
</tr>
<tr>
<td>7</td>
<td>out</td>
<td>TCP</td>
<td>2020</td>
<td>172.27.0.5:23</td>
<td>192.168.0.10:1025</td>
<td>ESTAB</td>
</tr>
</tbody>
</table>
Clearing Xlates

You can clear the global address-to-local address mapping information based on the global address, the global port, the local address, the local port, the interface address as the global address, and the NAT type by using the `clear xlate` command in Exec mode. When you enter this command, the ACE releases sessions that are using the translations (Xlates). The syntax of this command is as follows:

```
clear xlate [{global | local} start_ip [end_ip [netmask netmask]]] [{gport | lport} start_port [end_port]] [interface vlan number] [state static] [portmap]
```

The keyword, arguments, and options are as follows:

- `global`—(Optional) Clears the active translation by the global IP address.
- `local`—(Optional) Clears the active translation by the local IP address.
- `start_ip`—Global or local IP address in a global or local range of IP addresses. Enter an IP address in dotted-decimal notation (for example, 172.27.16.10).
- `end_ip`—(Optional) Last IP address in a global or local range of IP addresses. Enter an IP address in dotted-decimal notation (for example, 172.27.16.10).
- `netmask netmask`—(Optional) Specifies the network mask for global or local IP addresses. Enter a mask in dotted-decimal notation (for example, 255.255.255.0).
- `gport`—(Optional) Clears active translations by the global port.
- `lport`—(Optional) Clears active translations by the local port.
- `start_port`—Global or local port number.
- `end_port`—(Optional) Last port number in a global or local range of ports.
- `interface vlan number`—(Optional) Clears active translations by the VLAN number.
- `state static`—(Optional) Clears active translations by the state.
- `portmap`—(Optional) Clears active translations by the port map.

**Note**

If you configured redundancy, then you need to explicitly clear Xlates on both the active and the standby ACEs. Clearing Xlates on the active module alone will leave the standby module’s Xlates at the old mappings.
For example, to clear all static translations, enter:

```
host1/Admin# clear xlate state static
```

### NAT Configuration Examples

The following sections show typical scenarios that use dynamic and static NAT solutions:

- **Dynamic NAT and PAT (SNAT) Configuration Example**
- **Server Farm-Based Dynamic NAT (SNAT) Configuration Example**
- **Static Port Redirection (DNAT) Configuration Example**
- **SNAT with Cookie Load Balancing Example**

### Dynamic NAT and PAT (SNAT) Configuration Example

The following SNAT configuration example shows the commands that you use to configure dynamic NAT and PAT on your ACE. In this SNAT example, packets that ingress the ACE from the 192.168.12.0 network are translated to one of the IP addresses in the NAT pool defined on VLAN 200 by the `nat-pool` command. The `pat` keyword indicates that ports higher than 1024 are also translated.

If you are operating the ACE in one-arm mode, omit interface VLAN 100 and configure the service policy on interface VLAN 200.

```
access-list NAT_ACCESS line 10 extended permit tcp 192.168.12.0 255.255.255.0 1 72.27.16.0 255.255.255.0 eq http
class-map match-any NAT_CLASS
  match access-list NAT_ACCESS
policy-map multi-match NAT_POLICY
  class NAT_CLASS
    nat dynamic 1 vlan 200

interface vlan 100
  mtu 1500
  ip address 192.168.1.100 255.255.255.0
  service-policy input NAT_POLICY
  no shutdown

interface vlan 200
```
Server Farm-Based Dynamic NAT (SNAT) Configuration Example

The following SNAT configuration example shows the commands that you use to configure server farm-based dynamic NAT on your ACE. In this SNAT example, real servers addresses on the 172.27.16.0 network are translated to one of the IP addresses in the NAT pool defined on VLAN 200 by the `nat-pool` command.

If you are operating the ACE in one-arm mode, omit interface VLAN 100 and configure the service policy on interface VLAN 200.

```plaintext
access-list NAT_ACCESS line 10 extended permit tcp 192.168.12.0 255.255.255.0 1 72.27.16.0 255.255.255.0 eq http
rserver SERVER1
  ip address 172.27.16.3
  inservice
rserver SERVER2
  ip address 172.27.16.4
  inservice
serverfarm SFARM1
  rserver SERVER1
    inservice
  rserver SERVER2
    inservice
class-map type http loadbalance match-any L7_CLASS
  match http content .*cisco.com
class-map match-any NAT_CLASS
  match access-list NAT_ACCESS
policy-map type loadbalance http first-match L7_POLICY
  class L7_CLASS
    serverfarm SFARM1
      nat dynamic 1 vlan 200 serverfarm primary
policy-map multi-match NAT_POLICY
  class NAT_CLASS
    loadbalance policy L7_POLICY
    loadbalance vip inservice
interface vlan 100
  mtu 1500
  ip address 192.168.1.100 255.255.255.0
```
service-policy input NAT_POLICY
no shutdown

interface vlan 200
 mtu 1500
 ip address 172.27.16.2 255.255.255.0
 nat-pool 1 172.27.16.15 172.27.16.24 netmask 255.255.255.0
no shutdown

**Static Port Redirection (DNAT) Configuration Example**

The following DNAT configuration example shows those sections of the running configuration related to the commands necessary to configure static port redirection on your ACE. Typically, this configuration is used for DNAT, where HTTP packets that are destined to 192.0.0.0/8 and ingressing the ACE on VLAN 101 are translated to 10.0.0.0/8 and port 8080. In this example, the servers are hosting HTTP on custom port 8080.

access-list acl1 line 10 extended permit tcp 10.0.0.0 255.0.0.0
 eq 8080 any

class-map match-any NAT_CLASS
 match access-list acl1

policy-map multi-match NAT_POLICY
 class NAT_CLASS
  nat static 192.0.0.0 255.0.0.0 80 vlan 101

interface vlan 100
 mtu 1500
 ip address 192.168.1.100 255.255.255.0
 service-policy input NAT_POLICY
no shutdown

interface vlan 101
 mtu 1500
 ip address 172.27.16.100 255.255.255.0
no shutdown
SNAT with Cookie Load Balancing Example

The following configuration example shows those sections of the running configuration related to the commands necessary to configure SNAT (dynamic NAT) with cookie load balancing. Any source host that sends traffic to the VIP 20.11.0.100 is translated to one of the free addresses in the NAT pool in the range 30.11.100.1 to 30.11.200.1, inclusive. If you want to use PAT instead of NAT, replace “nat dynamic 1 vlan 2021” with “nat dynamic 2 vlan 2021” in the L7SLBCookie policy map.

```
server host http
  ip address 30.11.0.10
  inservice
serverfarm host httpsf
  rserver http
  inservice

class-map match-any vip4
  2 match virtual-address 20.11.0.100 tcp eq www
class-map type http loadbalance match-any L7SLB_Cookie
  3 match http cookie JG cookie-value ".**"
policy-map type loadbalance first-match L7SLB_Cookie
  class L7SLB_Cookie
  serverfarm httpsf
policy-map multi-match L7SLBCookie
  class vip4
    loadbalance vip inservice
    loadbalance L7SLB_Cookie
    nat dynamic 1 vlan 2021

interface vlan 2020
  ip address 20.11.0.2 255.255.0.0
  alias 20.11.0.1 255.255.0.0
  peer ip address 20.11.0.3 255.255.0.0
  service-policy input L7SLBCookie
  no shutdown
interface vlan 2021
  ip address 30.11.0.2 255.255.0.0
  alias 30.11.0.1 255.255.0.0
  peer ip address 30.11.0.3 255.255.0.0
  fragment min-mtu 68
  nat-pool 2 30.11.201.1 30.11.201.1 netmask 255.255.255.255 pat
  nat-pool 1 30.11.100.1 30.11.200.1 netmask 255.255.255.255
  no shutdown
```
INDEX

A

AAA
  accounting configuration, displaying 2-50
  accounting log information, displaying 2-50
  accounting method, defining default 2-45
  authentication configuration, displaying 2-53
  groups, displaying 2-47
  LDAP server, configuring for 2-33
  LDAP server configuration, displaying 2-49
  local and remote support 2-3
  login authentication method, defining 2-43
  overview 2-2
  quick start 2-8
  RADIUS server, configuring for 2-23
  RADIUS server configuration, displaying 2-47
  server, adding 2-22
  server groups, configuring 2-36
  status and statistics 2-46
  TACACS+ server, configuring for 2-29
  TACACS+ server configuration, displaying 2-48
  user accounts, creating 2-21

configuration, displaying 2-50
  default method, defining 2-45
  log information, displaying 2-50
  RADIUS server accounting settings, configuring 2-15
  TACACS+ server accounting settings, configuring 2-11

ACLs
  alternate address, ICMP message 1-15
  BPDUs 1-17
  clearing statistics 1-44
  comments in extended ACLs 1-16
  configuration information, displaying 1-42
  dynamic NAT 5-11
  EtherType, configuring 1-17
  EtherType examples 1-41
  expanded 1-4
  extended, configuring 1-6
  extended examples 1-33
  guidelines 1-3
  ICMP 1-7
  implicit deny 1-4
  inbound 1-34
  IP extended ACL 1-7
  IPs with NAT 1-37
maximum entries 1-4
merged 1-2
object groups 1-20 to 1-30
order of entries 1-3
outbound 1-34
overview 1-2
quick start 1-5
resequencing entries 1-19
static NAT 5-24, 5-34
statistics, displaying 1-42
types 1-3

application protocol inspection
class map overview 3-6
configuration examples 3-126, 3-128, 3-129
DNS 3-9, 3-102
FTP 3-10, 3-102
HTTP 3-12, 3-103
ICMP 3-15, 3-103
ILS 3-5, 3-16, 3-102, 3-104
Layer 3 and 4 HTTP parameter map 3-109
Layer 3 and 4 quick start 3-29
Layer 3 and 4 traffic policy configuration 3-90
Layer 7 FTP command inspection class map 3-31
Layer 7 FTP command inspection configuration 3-30
Layer 7 FTP command inspection quick start 3-23
Layer 7 HTTP deep packet inspection class map 3-39
Layer 7 HTTP deep packet inspection configuration 3-38
Layer 7 HTTP deep packet inspection policy map 3-63
Layer 7 HTTP deep packet inspection quick start 3-25
limitations 3-3
NAT and PAT support 3-3
overview 3-2
policy map overview 3-6
process flow diagram 3-8
protocol inspection overview 3-2
RTSP 3-18, 3-104
SCCP 3-6, 3-19, 3-70, 3-97, 3-102, 3-104, 3-112
service policy, defining 3-124
service policy, displaying 3-130
SIP 3-6, 3-19, 3-73, 3-97, 3-102, 3-104, 3-116
standards 3-3
statistics 3-130
supported protocols 3-3
authentication
configuration, displaying 2-53
local and remote support 2-3
local database 2-5
login method, defining 2-43
overview 2-7
RADIUS server authentication settings, configuring 2-14
TACACS+ server accounting settings, configuring 2-10
Index

B
bandwidth rate limiting 4-8
BPDU, in ACL 1-17
buffer size
  for connection parameter map 4-9
  receive or transmit data for each TCP connection 4-10

C
class map
  associating with Layer 7 policy map 3-37
  associating with policy map 3-67, 3-99
description, entering 3-106, 3-113, 3-118, 4-8
dynamic NAT 5-15
Layer 3 and 4 access list match criteria 3-95
Layer 3 and 4 class map, associating with policy map 4-32
Layer 3 and 4 class map, creating 3-92
Layer 3 and 4 description 3-94
Layer 3 and 4 port range criteria 3-96
Layer 4, creating 4-26
Layer 4 description 4-28
Layer 4 IP address criteria 4-28
Layer 4 port number criteria 4-29
Layer 7 FTP command inspection, configuring 3-31
Layer 7 FTP command inspection description 3-32
Layer 7 FTP request methods 3-33
Layer 7 HTTP deep packet inspection, configuring 3-40
Layer 7 HTTP deep packet inspection description 3-42
overview in application protocol inspection process 3-6
static NAT 5-29, 5-35
configurational examples
  application protocol inspection 3-129
  FTP 3-128
  HTTP 3-126
  TCP/IP normalization 4-49
connection parameter map
  action for segment overrun 4-12
  associating with policy map 4-33
  buffer size setting 4-9
  configuring for TCP/IP normalization 4-5
  creating for TCP/IP, UDP, and ICMP 4-7
  embryonic connection timeout 4-15
  half-closed connection timeout 4-15
  inactive connection timeout 4-17
  Nagle’s algorithm 4-13
  random TCP sequence numbers 4-13
  reserved bit handling 4-14
  segment size setting 4-10
  slow start algorithm 4-20
  TCP options, handling 4-21
  TCP SYN retries, limiting 4-12
  TCP SYN segments with data, handling 4-21
  type of service 4-26
urgent pointer policy 4-25
connections
  clearing 4-67
  embryonic, handling timeout of 4-15
  half-closed, handling timeout of 4-15
  inactive, handling timeout of 4-17
  rate limiting 4-8
  statistics, clearing 4-67
content type verification
  HTTP message 3-66
dynamic NAT
  See NAT

E
embryonic connection, handling timeout of 4-15
EtherType ACL
  configuring 1-17
  examples 1-41
extended ACL
  comments in 1-16
  configuring 1-6
  examples 1-33

F
fixups
  See application protocol inspection
fragment reassembly parameters
  See IP fragment reassembly parameters
FTP
  application protocol support 3-3, 3-4
  associating class map with policy map 3-37
  class map 3-31
  configuration examples 3-128
  inline match commands in policy map 3-35
  inspection overview 3-10
Layer 3 and 4 FTP application protocol inspection, configuring 3-102
Layer 7 FTP command inspection, configuring 3-30
passive with source NAT 5-16
policy actions 3-37
policy map 3-34
request methods, defining for command inspection 3-33
strict 3-11, 3-102

global addresses, guidelines for NAT 5-7

HTTP
application protocol support 3-4
associating class map with policy map 3-67
class map 3-39
configuration examples 3-126
content length, defining 3-43
content match criteria, defining 3-42
content type verification match criteria, defining 3-66
header for inspection 3-48
header value string expressions 3-51
HTTP/1/1 header fields, supported 3-49
inline match commands in policy map 3-65
inspection overview 3-12
internal compliance checks 3-67
Layer 3 and 4 HTTP application protocol inspection, configuring 3-103
Layer 7 HTTP deep packet inspection, configuring 3-38
Layer 7 HTTP deep packet inspection policy map 3-63
maximum header length for inspection 3-52
MIME type for inspection 3-53
parameter map 3-109
policy actions 3-68
policy map 3-63
request method for inspection 3-57
restricted category, defining (port misuse) 3-56
statistics from inspection 3-130
strict HTTP match criteria, defining 3-67
transfer encoding type for inspection 3-59
URL for inspection 3-60
URL length for inspection 3-62
HTTP/1/1 header fields, supported 3-49

ICMP
ACL 1-7
application protocol inspection, configuring 3-103
application protocol support 3-4, 3-5
conversion-error, ICMP message 1-15
echo, ICMP message 1-15
Index

echo reply, ICMP message 1-14
information reply, ICMP message 1-15
information request, ICMP message 1-15
inspection overview 3-15
mask reply, ICMP message 1-15
mask request, ICMP message 1-15
mobile redirect, ICMP message 1-15
NAT of ICMP error messages 3-104
parameter-problem, ICMP message 1-15
redirect, ICMP message 1-14, 1-15
router-advertisement, ICMP message 1-15
router-solicitation, ICMP message 1-15
security, disabling 4-36
source quench, ICMP message 1-14
time-exceeded, ICMP message 1-15
timestamp-reply, ICMP message 1-15
timestamp-request, ICMP message 1-15
trace route, ICMP message 1-15
types 1-14
unreachable, ICMP message 1-14
ILS inspection 3-5, 3-16, 3-102, 3-104
implicit PAT 5-2
inbound ACLs 1-34
inline match commands
  content type verification for HTTP inspection 3-66
in Layer 7 FTP command inspection policy map 3-35
in Layer 7 HTTP deep packet inspection policy map 3-65
strict HTTP for HTTP inspection 3-67
inspection engines
  See application protocol inspection
Internet Locator Service. See ILS
IP
  ACL 1-7
  address pool, for dynamic NAT 5-12, 5-24
  for ACL with NAT 1-37
  normalization, overview 4-3
  options, handling 4-41
IP fragment reassembly parameters
  configurational example 4-49
  configuring 4-43
  maximum fragment size setting 4-46
  maximum fragments setting 4-46
  MTU setting 4-45
  quick start 4-44
  reassembly timeout setting 4-47

Layer 3 and 4 application protocol inspection, configuring
  associating class map with policy map 3-99
  class map 3-92
  policy actions 3-101
  policy map 3-98
LDAP server
  ACE configuration 2-33
  configuration, displaying 2-49
configuration overview 2-18
directory server overview 2-6
parameters, setting 2-34
port, setting 2-35
search filter configuration 2-42
server group, creating 2-37
timeout, setting 2-36
user profile attribute type configuration 2-40
virtualization attributes, defining 2-12, 2-16, 2-19
local database authentication 2-5
login authentication method, defining 2-43

M

merged ACLs 1-2
MIME type, supported for HTTP inspection 3-53
MPLS, in ACL 1-17, 1-19
MTU
  in IP fragment reassembly configuration 4-45

N

Nagle’s algorithm 4-13
NAT
  ACL configuration, dynamic 5-11
  ACL configuration, static 5-24, 5-34
  application protocol inspection support 3-3
  as policy map action, dynamic 5-17
  as policy map action, static 5-28, 5-37
class map configuration, dynamic 5-15
class map configuration, static 5-29, 5-35
creating over 8 K static configurations 5-40
destination 5-2, 5-6, 5-7, 5-29, 5-32, 5-39, 5-50
dynamic NAT, overview 5-3
dynamic NAT and PAT, configuring 5-9
dynamic PAT, overview 5-4
global address guidelines 5-7
global IP address pool 5-12, 5-24
idle timeout, configuring 5-8
IPs in ACLs 1-37
maximum number of statements 5-7
overview 5-1
policy map configuration, dynamic 5-16
policy map configuration, static 5-30, 5-36
quick start, dynamic NAT and PAT 5-9
quick start, static NAT 5-20, 5-32
service policy, global dynamic 5-19
service policy, local dynamic 5-19
service policy, static 5-31, 5-39
source 5-2, 5-3, 5-4, 5-9
static NAT, overview 5-6
static NAT and port redirection, configuring 5-32
static port redirection 5-7
network address translation
  See NAT
normalization parameters
  configuring 4-34
Don’t Fragment bit, handling 4-41
ICMP security, disabling 4-36
IP options, handling 4-41
normalization send-reset, enabling 4-36
packet TTL setting 4-42
TCP normalization, disabling 4-34
unicast reverse-path forwarding, configuring 4-43

O

object groups
expanded 1-4
network 1-9
overview 1-20
service 1-14
order of ACL entries 1-3
outbound ACLs 1-34

P

packet TTL setting 4-42
parameter map
associating with Layer 3 and 4 policy map 3-108, 3-111, 3-115, 3-124
case sensitivity, disabling 3-109
configuring for Layer 3 and 4 HTTP inspection 3-109
maximum content bytes setting 3-110
maximum header bytes setting 3-110

passive FTP with source NAT 5-16
PAT
configuring 5-9
implicit 5-2
overview 5-4
policy map
actions, defining 3-37, 3-68, 3-101
associating with connection parameter map 4-33
dynamic NAT 5-16
dynamic NAT as policy map action 5-17
Layer 3 and 4, associating with class map 3-99
Layer 3 and 4, associating with parameter map 3-108, 3-111, 3-115, 3-124
Layer 3 and 4, associating with service policy 4-33
Layer 3 and 4, configuring HTTP parameter map 3-109
Layer 3 and 4, creating 3-98, 4-31
Layer 3 and 4, defining 3-98
Layer 3 and 4, description 3-99
Layer 3 and 4 policy map, associating with class map 4-32
Layer 7 FTP command inspection, adding description 3-35
Layer 7 FTP command inspection, associating with class map 3-37
Layer 7 FTP command inspection, creating 3-34
Layer 7 FTP command inspection, defining 3-34
Layer 7 FTP command inspection, inline match commands 3-35
Layer 7 HTTP deep packet inspection, adding description 3-64
Layer 7 HTTP deep packet inspection, associating with class map 3-67
Layer 7 HTTP deep packet inspection, creating 3-63
Layer 7 HTTP deep packet inspection, inline match commands 3-65
Layer 7 HTTP deep packet inspection, inline match commands 3-65
overview in application protocol inspection process 3-6
static NAT 5-30, 5-36
static NAT as policy map action 5-28, 5-37
port
for LDAP server 2-35
number or range for Layer 3 and 4 application protocol inspection 3-96
port redirection, configuring 5-32
port redirection
configuring 5-32
overview 5-7
preshared key
RADIUS, setting for 2-26
TACACS+, setting for 2-31

Q
quick start
AAA configuration 2-8
ACL configuration 1-5
dynamic NAT and PAT configuration 5-9
IP fragment reassembly configuration 4-44
Layer 3 and 4 application protocol inspection 3-29
Layer 7 FTP command inspection 3-23
Layer 7 HTTP deep packet inspection 3-25
static NAT configuration 5-20, 5-32
TCP/IP normalization 4-3

R
RADIUS server
ACE configuration 2-23
adding 2-22
authentication settings, configuring 2-14
configuration, displaying 2-47
dead-time setting 2-27
global preshared key setting 2-26
NAS-IP-Address attribute setting 2-26
number of retransmissions, setting 2-28
parameters, setting 2-23
server accounting settings, configuring 2-15
server group, creating 2-37
server group dead-time setting 2-39
server overview 2-5
timeout setting 2-29
rate limiting
bandwidth 4-8
connection 4-8
remarks in extended ACLs 1-16
reordering ACL entries 1-19
request methods
  FTP command inspection, defining for 3-33
  HTTP inspection, defining for 3-57
resequencing ACL entries 1-19
reserved bits, handling in connection parameter map 4-14
restricted category, defining for HTTP inspection (port misuse) 3-56
reverse-path forwarding, configuring 4-43
RTSP
  application protocol inspection, configuring 3-104
  application protocol support 3-6
  inspection overview 3-18
  restrictions 3-18
rules, maximum in ACL 1-4

TACACS+ 2-37
service policy
  applying to VLAN interfaces 3-124
  associating with Layer 3 and 4 policy map 4-33
  configuration information 3-131
dynamic NAT, global 5-19
dynamic NAT, local 5-18
static NAT, local 5-31, 5-39
Session Initiation Protocol. See SIP
SIP
  inspection 3-6, 3-19, 3-73, 3-97, 3-102, 3-104, 3-116
  inspection, enabling logging of packets 3-123
Skinny Client Control Protocol. See SCCP
slow start algorithm, enabling in connection parameter map 4-20
source NAT 5-2, 5-3, 5-4, 5-9
static NAT
  See NAT
statistics
  AAA 2-46
  ACL, clearing 1-44
  ACL, displaying 1-42
  connection, clearing 4-67
  HTTP inspection 3-130
  IP, clearing 4-68
  IP fragmentation and reassembly, clearing 4-69
  IP fragmentation and reassembly, displaying 4-61
  IP traffic 4-58

SCCP
  inspection 3-6, 3-19, 3-70, 3-97, 3-102, 3-104, 3-112
segment size
  action for overrun 4-12
  for connection parameter map 4-10
server groups
  configuring 2-36
  creating 2-37
LDAP 2-37
RADIUS 2-37
Index

service policy 4-64
TCP, clearing 4-68
TCP, displaying 4-62
TCP/IP and UDP connections 4-55
TCP/IP connections and IP reassembly, clearing 4-68
TCP/IP connections and IP reassembly, displaying 4-51
UDP, clearing 4-69
UDP, displaying 4-63
switch mode, configuring 4-47
SYN cookie
  configurational and operational considerations 4-39
  configuring on an interface 4-40
  displaying statistics 4-65
  overview 4-38
SYN flood attack 4-37

TACACS+ server
  accounting settings, configuring 2-11
  ACE configuration 2-29
  adding 2-22
Cisco Secure Access Control Server (ACS) 2-10, 2-11
  configuration, displaying 2-48
  dead-time setting 2-32
  global preshared key setting 2-31
parameters, setting 2-30
server authentication settings, configuring 2-10
server group, creating 2-37
server group dead-time setting 2-39
server overview 2-5
timeout setting 2-33
TCP
  connection, receive or transmit buffer size 4-10
  normalization, disabling 4-34
  normalization, overview 4-2
  normalization send-reset, enabling 4-36
  options, handling in connection parameter map 4-21
  port numbers and key words 1-10
  sequence numbers, randomizing 4-13
  slow start algorithm, enabling in connection parameter map 4-20
  SYN retries, limiting in connection parameter map 4-12
  SYN segments with data, handling in connection parameter map 4-21
  WAN optimization 4-17
TCP/IP and UDP configurations, displaying 4-51
TCP/IP normalization
  clearing connections 4-67
  configuration example 4-49
  connection parameter map, configuring 4-5
  IP fragment reassembly parameters, configuring 4-43
Layer 3 and 4 policy map, configuring 4-31
Layer 4 class map, configuring 4-26
normalization parameters, configuring 4-34
overview 4-2
quick start 4-3
statistics, clearing 4-68, 4-69
statistics, displaying 4-51
statistics, IP fragmentation and reassembly 4-61
statistics, IP traffic 4-58
statistics, service policy 4-64
statistics, TCP 4-62
statistics, TCP/IP connections 4-55
statistics, UDP 4-63
TCP/IP and UDP configurations, displaying 4-51
traffic policy, configuring 4-26
traffic class
See class map
traffic policies
TCP/IP normalization 4-26
transfer encoding, defining for HTTP inspection 3-59
TTL setting 4-42
type of service, setting in connection parameter map 4-26

 UDP

port numbers and key words 1-12
UDP and TCP/IP configurations, displaying 4-51
unicast reverse-path forwarding, configuring 4-43
urgent pointer policy, setting in connection parameter map 4-25

URL
defining for HTTP deep packet inspection 3-60
length, defining for HTTP deep packet inspection 3-62
regular expressions 3-61
URL request logging 3-103

W

WAN optimization 4-17

U

UDP