Configuring Application Protocol Inspection on the Virtual Firewall

This chapter describes how to configure application protocol inspection for the VFW application. 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 FTP request command filtering.

Feature History for Configuring Application Protocol Inspection on the VFW Application

<table>
<thead>
<tr>
<th>Release</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 3.5.0</td>
<td>This feature was introduced on the multiservice blade (MSB) for the Cisco XR 12000 Series Router.</td>
</tr>
<tr>
<td>Release 3.6.0</td>
<td>No modification.</td>
</tr>
</tbody>
</table>

Contents

- Information About Application Protocol Inspection, page VFC-109
- How to Configure Application Protocol Inspection, page VFC-117
- How to Configure an HTTP Parameter Map for Use in a Layer 3 and Layer 4 Policy Map, page VFC-144
- Configuration Examples for Application Protocol Inspection, page VFC-146
- Additional References, page VFC-147

Information About Application Protocol Inspection

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

This section includes the following topics on application protocol inspection:

- Performing Application Protocol Inspection, page VFC-110
Performing Application Protocol Inspection

You can configure the VFW application to perform application protocol inspection, sometimes referred to as application protocol fixup, for applications that:

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

You may require that the VFW application perform application inspection of HTTP, FTP, DNS, ICMP, and RTSP protocols as a first step before passing the packets to the destination server. For HTTP, the VFW application performs deep packet inspection to statefully monitor the HTTP protocol and permits or denies traffic based on user-defined traffic policies. HTTP deep packet inspection focuses mainly on HTTP attributes such as HTTP header, URL, and payload. For FTP, the VFW application performs FTP command inspection for FTP sessions, allowing you to restrict specific commands by the VFW application.

Application inspection helps you identify the location of embedded IP addressing information in the TCP or UDP flow. This inspection allows the VFW application to translate embedded IP addresses and to update any checksum or other fields that are affected by the translation.

The need to translate 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 9 describes the application inspection protocols supported by the VFW application, 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 9 Application Inspection Support

<table>
<thead>
<tr>
<th>Application Protocol</th>
<th>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 background 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 secondary channel for data. See the “FTP Inspection” section for background information.</td>
</tr>
</tbody>
</table>
You configure rules for application protocol inspection through the use of 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**—Provides the Layer 7 network traffic classification to identify HTTP deep protocol inspection attributes (such as HTTP header and URL) and FTP request commands.

- **Layer 7 Policy Map**—Configures the applicable HTTP deep packet inspection or FTP request command actions executed on the network traffic that match the classifications defined in the Layer 7 class map.

- **Layer 3 and Layer 4 Class map**—Classifies network traffic passing through the VFW application for application inspection and matches traffic associated with the specified `inspect` commands in a policy map.

- **Layer 3 and Layer 4 Policy map**—Enables HTTP, DNS, FTP, ICMP, and RTSP protocol inspection and FTP command inspection for a traffic classification that matches the criteria listed the class map.
• **Service policy**—Activates the policy map and attaches the traffic policy to an interface or globally on all interfaces.

The flow chart shown in Figure 14 provides a basic overview of the process required to configure class maps and policy maps to perform application protocol inspection. The flow chart also illustrates how the VFW application associates the various components of the class map and policy map configuration with each other.

**Figure 14 Application Protocol Inspection Configuration Flow Diagram**

**Application Inspection Protocol Overview**

This section provides an overview on the following application inspection protocols supported by the VFW application:

- HTTP Deep Packet Inspection, page VFC-113
- DNS Inspection, page VFC-113
HTTP Deep Packet Inspection

The VFW application performs a stateful deep packet inspection of the HTTP protocol. Deep packet inspection is a special case of application inspection where the VFW application 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 HTTP header, URL, and to a limited extent, the payload. User-defined regular expressions can also be used to detect “signatures” in the payload.

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.

The security features covered by HTTP application inspection include:
- RFC compliance monitoring and RFC method filtering
- Content, URL, and HTTP header length checks
- Transfer-encoding methods
- Content type verification and filtering
- Port 80 misuse

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 VFW application 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; the VFW application does not handle PTR records.

Note

The DNS rewrite function is not applicable for PAT because multiple PAT rules are applicable for each A-record. The use of multiple PAT rules makes it difficult for the VFW application 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.

Note

If you enter the `inspect dns` command without specifying the `maximum-length` option, the VFW application does not check the DNS packet size.

- Performs a number of security checks, including:
  - Verification that the maximum label length is no greater than 63 bytes
  - Verification that the maximum domain name length is no greater than 255 bytes
Check for the existence of compression loops

A single connection is created for multiple DNS sessions, as long as 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 security appliance within a limited period of time and there is no resource buildup. However, if you enter the show connection command, you see the idle timer of a DNS connection being reset by a new DNS session. This reset action is due to the nature of the shared DNS connection and is intended by design.

FTP Inspection

File Transfer Protocol (FTP) inspection inspects FTP sessions for address translation in a message, dynamic opening of ports, stateful tracking of request and response messages. Each specified FTP command must be acknowledged before the VFW application allows a new command. Command filtering allows you to restrict specific commands by the VFW application. When the VFW application denies a command, it closes the connection.

The FTP command inspection process, as performed by the VFW application:

- 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 pre-negotiated. The port is negotiated through the PORT or PASV commands.
- Tracks the FTP command-response sequence. The VFW application performs the following command checks listed below. If you specify the strict keyword with the inspect ftp command in a Layer 3 and Layer 4 policy map, the VFW application tracks each FTP command and response sequence for the anomalous activity outlined below. The strict keyword is used in conjunction 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 SYST command.

Note The use of the strict option may affect FTP clients that do not comply with the RFC standards.

- **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 VFW application assumes that the PORT command is truncated and 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 VFW application closes the connection.
- **Size of RETR and STOR commands**—Checked the size of the RETR and STOR commands against a fixed constant of 256. If the size is greater, the VFW application 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 VFW application 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 VFW application denies the TCP connection. This denial prevents a security hole when the user executes "227 xxxxx x1, x2, x3, x4, p1, p2."
- **Invalid port negotiation**—Checks the negotiated dynamic port value to verify that it is greater than 1024 (port numbers in the range from 2 to 1024 are reserved for well-known connections). If the negotiated port falls in this range, the VFW application 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 VFW application closes the TCP connection.

- Translates embedded IP addresses in conjunction with NAT. FTP command inspection translates the IP address within the application payload. Refer to RFC 959 for background details.

## ICMP Inspection

Internet Control Message Protocol (ICMP) inspection allows ICMP traffic to have a “session” so it can be inspected similarly to TCP and UDP traffic. Without using ICMP inspection, we recommend that you do not allow ICMP traffic to pass through the VFW application in an ACL. Without performing 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 message
- 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. These error messages carry the original packet for which the error is generated in the data part of the message. They also contain the addresses of the intermediate node or endpoint node in the outer header and the destination in the inner header. ICMP error fixup handles address translation of node address and destination address to global addresses using NAT configuration.

ICMP error fixup is user-configurable, and if not enabled, 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 originating from the destination and the node addresses or the route to the destination is not revealed.

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 interface.
- Verifies that the connection record contains a sequence number window specifying the list of sequence numbers of outstanding requests for which replies are pending.
- Verifies that the connection record should have a timeout, so that inactive connection records can be reused for other flows and can protect 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 VFW application.
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 corresponding 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 itself is 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.

**RTSP Inspection**

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 VFW application only supports TCP 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. 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 pinhole for the data channel.

The VFW application parses SETUP response messages with a status code of 200.

Because RFC 2326 does not require that the client and server ports be contained in the SETUP response message, the VFW application must keep track of 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 VFW application does not:

- Inspect RTSP messages passing 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 VFW application:

- With Cisco IP/TV, the number of translations the VFW application 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 transport mode. For the VFW application, 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 VFW application.
  - 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 VFW application.
Configuring Application Protocol Inspection on the Virtual Firewall

How to Configure Application Protocol Inspection

The following tasks detail the procedures required to configure application protocol inspection on the VFW application:

- Configuring a Layer 7 HTTP Deep Inspection Policy, page VFC-117
- Configuring a Layer 7 FTP Command Inspection Policy, page VFC-132
- Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy, page VFC-136
- Applying a Traffic Policy to an Interface, page VFC-140
- Displaying Application Protocol Inspection Statistics and Service Policy Information, page VFC-142

Configuring a Layer 7 HTTP Deep Inspection Policy

This task describes how to create a Layer 7 class map and policy map to be used for HTTP deep packet inspection by the VFW application. The VFW application performs a stateful deep packet inspection of the HTTP protocol and permits or restricts traffic based on the actions in your configured policy maps. The following security features are included as part of HTTP deep packet inspection as performed by the VFW application:

- Regular expression matching on name in an HTTP header, URL name, or content expressions in an HTTP entity body
- Content, URL, and HTTP header length checks
- MIME-type message inspection
- Transfer-encoding methods
- Content type verification and filtering
- Port 80 misuse by tunneling protocols
- RFC compliance monitoring and RFC method 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:

- Match statements in Layer 7 class maps
- Inline match statements in Layer 7 policy maps
- Header insertion and rewrite (including SSL URL rewrite) expressions in Layer 7 action lists

To configure a Layer 7 HTTP deep inspection policy, you must perform each of the following tasks:

- Creating a Layer 7 HTTP Deep Inspection Class Map, page VFC-118
- Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map, page VFC-126
- Configuring a Layer 3 and Layer 4 Traffic Policy for Layer 7 HTTP Deep Packet Inspection, page VFC-129
- Applying a Traffic Policy to an Interface, page VFC-131
Creating a Layer 7 HTTP Deep Inspection Class Map

This task describes how to create a Layer 7 HTTP deep inspection class map.

Prerequisites

You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

SUMMARY STEPS

1. `changeto context_name`
2. `configure`
3. `class-map type http [match-all | match-any] map_name`
4. `[line_number] match content expression`
5. `[line_number] match content length operator bytes`
6. `[line_number] match header {header_name | header_field} header-value expression`
7. `[line_number] match header length {request | response} operator bytes`
8. `[line_number] match header mime-type mime_type`
9. `[line_number] match port-misuse application_category`
10. `[line_number] match request-method {ext | rfc} method`
11. `[line_number] match transfer-encoding coding_types`
12. `[line_number] match url expression`
13. `[line_number] match url length operator bytes`
14. `exit`
15. `copy running-config startup-config`

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td><code>changeto context_name</code></td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>firewall/Admin# changeto C1</td>
<td></td>
</tr>
<tr>
<td>firewall/C1#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td><code>configure</code></td>
<td></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td>firewall/Admin# configure</td>
<td></td>
</tr>
<tr>
<td>Enter configuration commands, one per line. End with CNTL/Z. firewall/Admin(config)#</td>
<td></td>
</tr>
</tbody>
</table>
### Configuring Application Protocol Inspection on the Virtual Firewall

**How to Configure Application Protocol Inspection**

**VFC-119**

*Cisco IOS XR Virtual Firewall Configuration Guide*

OL-15429-03

---

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 3</strong></td>
<td>Creates a Layer 7 class map that is used for the deep packet inspection of HTTP traffic. If you do not specify <code>match-all</code> or <code>match-any</code>, traffic must match all the match criteria to be classified as part of the traffic class.</td>
</tr>
</tbody>
</table>

**Example:**

```
firewall/Admin(config)# class-map type http
inspect match-any HTTP_INSPECT_L7CLASS
firewall/Admin(config-cmap-http-insp)#
```

| Step 4 | (Optional) Use the `match content` command to configure the class map to define HTTP application inspection decisions based on content expressions contained within the HTTP content. The `expression` argument specifies the content contained within the HTTP entity-body. The range is from 1 to 255 alphanumeric characters. See Table 11 for a list of the supported characters that you can use in regular expressions. |

**Example:**

```
firewall/Admin(config-cmap-http-insp)# match content .*newpspsig
```

| Step 5 | (Optional) Use the `match content length` command to configure the class map to define application inspection decisions in the HTTP content up to the configured maximum content parse length. Allowable operators are as follows: |

- `lt`—Less than
- `gt`—Greater than
- `eq`—Equal to
- `neq`—Not equal to
- `range`—An inclusive range of size values |

The bytes argument represents the content parse length in an HTTP message received by the VFW application. Valid entries are from 1 to 65535 bytes.

**Example:**

```
firewall/Admin(config-cmap-http-insp)# match content length eq 1000
```
### Command or Action

**Step 6**

```plaintext
[line_number] match header (header_name | header_field) header-value expression
```

**Example:**

```plaintext
firewall/Admin(config-cmap-http-insp)# match header Host header-value .mycompanyexample.com
```

**Purpose**

(Optional) 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.

- **header_name**—Specifies the 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, provided that you enclose the entire string in quotation marks (").

- **header_field**—Specifies a standard HTTP/1.1 header field. Valid selections include request-header fields, general-header fields, and entity-header field. Table 10 lists the supported HTTP/1.1 header fields.

- **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. The VFW application 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 11 for a list of the supported characters that you can use in regular expressions.

**Step 7**

```plaintext
[line_number] match header length (request | response) operator bytes
```

**Example:**

```plaintext
firewall/Admin(config-cmap-http-insp)# match header length request eq 256
```

**Purpose**

(Optional) Limits the HTTP traffic allowed through the VFW application based on the length of the entity body in the HTTP message.

- **request**—Specifies the size of the HTTP header request message that can be received by the VFW application.
- **response**—Specifies the size of the HTTP header response message sent by the VFW application.

Allowable operators are as follows:

- `lt`—Less than
- `gt`—Greater than
- `eq`—Equal to
- `neq`—Not equal to
- `range`—An inclusive range of size values

The `bytes` argument represents the size of the entity-body in an HTTP message received by the VFW application. Valid entries are from 1 to 65535 bytes.

**Step 8**

```plaintext
[line_number] match header mime-type mime_type
```

**Example:**

```plaintext
firewall/Admin(config-cmap-http-insp)# match header mime-type audio\midi
firewall/Admin(config-cmap-http-insp)# match header mime-type audio\mpeg
```

**Purpose**

(Optional) Specifies a subset of the Multipurpose Internet Mail Extension (MIME)-type messages to be permitted or denied by the VFW application. The `mime_type` argument specifies the MIME type to be permitted through the VFW application. By default all mime-types are allowed. Table 12 lists all supported MIME types.
### Command or Action

| Step 9 | [line_number] match port-misuse application_category |

**Example:**
```
firewall/Admin(config-cmap-http-insp)# match port-misuse p2p
```

(Optional) Configures the class map to define application inspection compliance decisions that restrict certain HTTP traffic from passing through the VFW application. The `application_category` argument specifies the restricted HTTP application category for the class map. The possible values for `application_category` include:

- **im**—Instant messaging application category. The VFW application 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 Httptunnel, and Firethru.

### Step 10

| [line_number] match request-method (ext | rfc) method |

**Example:**
```
firewall/Admin(config-cmap-http-insp)# match request-method rfc connect
firwall/Admin(config-cmap-http-insp)# match request-method rfc get
firewall/Admin(config-cmap-http-insp)# match request-method rfc head
defirewall/Admin(config-cmap-http-insp)# match request-method ext index
```

(Optional) Configures the class map to define application inspection compliance decisions based on the request methods defined in RFC 2616 and by HTTP extension methods.

- **ext method**—Specifies an HTTP extension method. If the RFC request messages does not contain one of the RFC 2616 HTTP request methods, the VFW application verifies that it is an extension method. The VFW application supports the inspection of the following HTTP request extension methods: `copy`, `edit`, `getattr`, `getattname`, `getprops`, `index`, `lock`, `mkdir`, `move`, `revadd`, `relabel`, `relog`, `revnum`, `save`, `setattr`, `startrev`, `stoprev`, `unedit`, and `unlock`.
- **rfc method**—Specifies an RFC 2616 HTTP request method that you want to perform an RFC compliance check on. The VFW application supports the inspection of the following RFC 2616 HTTP request methods: `connect`, `delete`, `get`, `head`, `options`, `post`, `put`, and `trace`. 
### Command or Action

<table>
<thead>
<tr>
<th>Step 11</th>
<th><code>[line_number] match transfer-encoding coding_types</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>firewall/Admin(config-cmap-http-insp)# match transfer-encoding chunked</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>(Optional) Configures the class map to define application inspection decisions that limit the HTTP transfer-encoding types that can pass through the VFW application. The <code>coding_types</code> argument specifies the HTTP transfer-encoding type for the class map. Possible values include:</td>
</tr>
<tr>
<td></td>
<td><strong>chunked</strong> — Message body is transferred as a series of chunks.</td>
</tr>
<tr>
<td></td>
<td><strong>compress</strong> — The encoding format produced by the common UNIX file compression program “compress”. This format is an adaptive Lempel-Ziv-Welch coding (LZW).</td>
</tr>
<tr>
<td></td>
<td><strong>deflate</strong> — The .zlib format defined in RFC 1950 in combination with the deflate compression mechanism described in RFC 1951.</td>
</tr>
<tr>
<td></td>
<td><strong>gzip</strong> — An 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.</td>
</tr>
<tr>
<td></td>
<td><strong>identity</strong> — The default (identity) encoding, which does not require the use of transformation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step 12</th>
<th><code>[line_number] match url expression</code></th>
</tr>
</thead>
</table>
| **Example:** | `firewall/Admin(config-cmap-http-insp)# match url .*.gif`  
`firewall/Admin(config-cmap-http-insp)# match url .*.html` |
| **Purpose:** | (Optional) Configures the class map to define application inspection decisions based on URL name. The `expression` argument specifies the URL, or portion of a URL, to match and can be from 1 to 255 characters. Include only the portion of the URL following `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 VFW application supports the use of regular expressions for matching. See Table 11 for a list of the supported characters that you can use in regular expressions. |

<table>
<thead>
<tr>
<th>Step 13</th>
<th><code>[line_number] match url length operator bytes</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td><code>firewall/Admin(config-cmap-http-insp)# match url length eq 10000</code></td>
</tr>
<tr>
<td><strong>Purpose:</strong></td>
<td>(Optional) Limits the HTTP traffic allowed through the VFW application by specifying the maximum length of a URL in a request message that can be received by the VFW application. Allowable operators are as follows:</td>
</tr>
<tr>
<td></td>
<td><strong>lt</strong> — Less than</td>
</tr>
<tr>
<td></td>
<td><strong>gt</strong> — Greater than</td>
</tr>
<tr>
<td></td>
<td><strong>eq</strong> — Equal to</td>
</tr>
<tr>
<td></td>
<td><strong>neq</strong> — Not equal to</td>
</tr>
<tr>
<td></td>
<td><strong>range</strong> — An inclusive range of size values</td>
</tr>
<tr>
<td><strong>The <code>bytes</code> argument represents the size of the URL received by the VFW application. Valid entries are from 1 to 65535 bytes.</strong></td>
<td></td>
</tr>
</tbody>
</table>
**How to Configure Application Protocol Inspection**

**Table 10** lists the supported HTTP/1.1 header fields.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>A 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>The character sets 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>The 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 along 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>Allows the sender to specify connection options.</td>
</tr>
<tr>
<td>Content-MD5</td>
<td>An 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 what behaviors the client requires.</td>
</tr>
<tr>
<td>From</td>
<td>Contains the e-mail address of the person that controls the requesting user agent.</td>
</tr>
<tr>
<td>Host</td>
<td>The Internet host and port number of the resource being requested, as obtained from the original uniform resource identifier (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>
</tbody>
</table>

**Step 14**

```
exit
```

**Example:**

```
firewall/Admin(config-if-mgmt)# exit
firewall/Admin#
```

**Step 15**

```
copy running-config startup-config
```

**Example:**

```
firewall/Admin# copy running-config startup-config
```
Configuring Application Protocol Inspection on the Virtual Firewall

How to Configure Application Protocol Inspection

Table 10  HTTP/1.1 Header Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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. The purpose of this feature is to allow 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 &quot;*&quot; 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, a comma-separated list of entries, for which the optional parameters are separated by semicolons.</td>
</tr>
<tr>
<td>Referer</td>
<td>The address (URI) of the resource from which the URI in the request was obtained.</td>
</tr>
<tr>
<td>Transfer-Encoding</td>
<td>Indicates what (if any) type of transformation has been applied to the message body to safely transfer it between the sender and the recipient.</td>
</tr>
<tr>
<td>User-Agent</td>
<td>Information about the user agent, for example a software program originating the request. This information is for statistical purposes, the tracing of protocol violations, and automated recognition of user agents for the sake of tailoring 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>

Table 11 provides a list of the supported characters that you can use in regular expressions.

Table 11  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 of 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>
</tbody>
</table>
Table 11  Special Characters for Matching String Expressions (continued)

<table>
<thead>
<tr>
<th>Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>\w</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>

Table 12 lists the supported MIME types.

Table 12  Supported MIME Types

<table>
<thead>
<tr>
<th>application\msexcel</th>
<th>image\x-portable-bitmap</th>
<th>video\sgi</th>
</tr>
</thead>
<tbody>
<tr>
<td>application\mspowerpoint</td>
<td>image\x-portable-greymap</td>
<td>video\x-fli</td>
</tr>
<tr>
<td>application\msword</td>
<td>image\x-xpm</td>
<td>audio\x-adpcm</td>
</tr>
<tr>
<td>application\octet-stream</td>
<td>text*</td>
<td>audio\x-aiff</td>
</tr>
<tr>
<td>application\pdf</td>
<td>text\css</td>
<td>audio\x-ogg</td>
</tr>
<tr>
<td>application\postscript</td>
<td>text\html</td>
<td>audio\x-wav</td>
</tr>
<tr>
<td>application\x-gzip</td>
<td>text\plain</td>
<td>image*</td>
</tr>
<tr>
<td>application\x-java-archive</td>
<td>text\richtext</td>
<td>image\gif</td>
</tr>
<tr>
<td>application\x-java-vm</td>
<td>text\sgml</td>
<td>image\jpeg</td>
</tr>
<tr>
<td>application\x-messenger</td>
<td>text\xmcdd</td>
<td>image\png</td>
</tr>
<tr>
<td>application\zip</td>
<td>text\xml</td>
<td>image\tiff</td>
</tr>
<tr>
<td>audio*</td>
<td>video*</td>
<td>image\x-3ds</td>
</tr>
<tr>
<td>audio\basic</td>
<td>video\flc</td>
<td>image\x-bitmap</td>
</tr>
<tr>
<td>audio\midi</td>
<td>video\mpeg</td>
<td>image\x-niff</td>
</tr>
<tr>
<td>audio\mpeg</td>
<td>video\quicktime</td>
<td></td>
</tr>
</tbody>
</table>

Example

The following example illustrates how 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 VFW application to indicate a match:

```
host1/Admin(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
firewall/Admin(config-cmap-http-insp)# match header length request eq 200
firewall/Admin(config-cmap-http-insp)# match header Host header-value .\*mycompanyexample.com
```
```plaintext
firewall/Admin(config-cmap-http-insp)# match url length eq 10000
firewall/Admin(config-cmap-http-insp)# match url *.gif
```

**What to Do Next**

After configuring a Layer 7 HTTP deep inspection class map, you need to configure a Layer 7 HTTP deep packet inspection policy map as described in the “Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map” section on page VFC-126.

**Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map**

This task 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, as defined in “Creating a Layer 7 HTTP Deep Inspection Class Map” section on page VFC-118. You then associate the completed Layer 7 HTTP deep packet inspection policy with a Layer 3 and Layer 4 policy map, and activate the operation on an interface (see the “Configuring a Layer 3 and Layer 4 Traffic Policy for Layer 7 HTTP Deep Packet Inspection” section on page VFC-129 and the “Applying a Traffic Policy to an Interface” section on page VFC-131 for more information).

**Prerequisites**

- You must have configured a Layer 7 HTTP inspection class map as described in the “Creating a Layer 7 HTTP Deep Inspection Class Map” section on page VFC-118.
- You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

**SUMMARY STEPS**

1. configure
2. policy-map type inspect http all-match map_name
3. class map_name
4. permit
   or
   reset
5. end
6. copy running-config startup-config
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode. You are now within configuration mode of the VFW application.</td>
</tr>
</tbody>
</table>

Example:
```
firewall/Admin# configure
Enter configuration commands, one per line. End with CNTL/Z.
firewall/Admin(config)#
```

<table>
<thead>
<tr>
<th><strong>Step 2</strong> policy-map type inspect http all-match map_name</th>
<th>Creates and configures a Layer 7 policy map that enables the deep packet inspection of the HTTP protocol.</th>
</tr>
</thead>
</table>

Example:
```
firmware/Admin(config)# policy-map type inspect http all-match HTTP_INSPECT_L7POLICY
```

<table>
<thead>
<tr>
<th><strong>Step 3</strong> class map_name</th>
<th>Associates a class map defined in “Creating a Layer 7 HTTP Deep Inspection Class Map” section on page VFC-118 with the Layer 7 policy map, and enters policy map class configuration mode.</th>
</tr>
</thead>
</table>

It is possible to include a single inline match criteria in the policy map without specifying a traffic class using 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. Refer to the “Configuration Tip: Including Inline Match Statements in a Layer 7 HTTP Deep Packet Inspection Policy Map” section on page VFC-128 for more information.

Example:
```
firmware/Admin(config-pmap-ins-http)# class HTTP_INSPECT_L7CLASS
```

<table>
<thead>
<tr>
<th><strong>Step 4</strong> permit</th>
<th>Specifies to permit or deny the traffic defined by the class. If reset is used, a TCP reset message is sent to the client or server to close the connection.</th>
</tr>
</thead>
</table>

By default, HTTP inspection allows traffic which 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 VFW application. For example, you can include a class map to allow the HTTP GET method and use the class class-default command to block all the other requests.

**Note** By default, all matches are applied to both HTTP request and response messages, but the class class-default command is only applied to HTTP requests.
Configuration Tip: Including Inline Match Statements in a Layer 7 HTTP Deep Packet Inspection Policy Map

To include a single inline match criterion in the policy map without specifying a traffic class, enter an applicable Layer 7 match command. The inline Layer 7 policy map match commands function the same as a Layer 7 class map with match commands. However, you can use only a single inline match command in a policy map. If you require additional match criterion, you must configure a class map.

The HTTP deep packet inspection policy map inline match commands include the following:

- `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 header {header_name | header_field} header-value expression`
- `match name header length {request | response} {eq bytes | gt bytes | lt bytes | range bytes1 bytes2}`
- `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 bytes2}`

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 VFW application. 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 VFW application performs the specified Layer 7 policy map action: permit or reset.
How to Configure Application Protocol Inspection

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 VFW application.

- **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 VFW application performs the specified Layer 7 policy map action: permit or reset.

For example, to add an inline match command to a Layer 7 HTTP deep inspection policy map, enter:

```
firewall/Admin(config-pmap-ins-http)# match L7httpinspect port-misuse p2p
```

What to Do Next

You must configure a Layer 3 and Layer 4 policy map and associate it with the Layer 7 HTTP deep packet inspection policy map that you created in this task. See “Configuring a Layer 3 and Layer 4 Traffic Policy for Layer 7 HTTP Deep Packet Inspection” section on page VFC-129.

Configuring a Layer 3 and Layer 4 Traffic Policy for Layer 7 HTTP Deep Packet Inspection

After you create a Layer 7 HTTP deep packet inspection policy, you must associate it with a Layer 3 and Layer 4 policy map. This task describes briefly how to create a Layer 3 and Layer 4 policy map and associate it with the Layer 7 HTTP deep packet inspection policy map. For more information regarding Layer 3 and Layer 4 class maps and policy maps, refer to the “Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy” section on page VFC-136.

Prerequisites

- You must have configured a Layer 7 HTTP deep packet inspection policy. Refer to the “Creating a Layer 7 HTTP Deep Inspection Class Map” section on page VFC-118 and the “Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map” section on page VFC-126.
- You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

SUMMARY STEPS

1. configure
2. class-map [match-all | match-any] map_name
3. [line_number] match port {tcp | udp} {any | eq port_number | range port1 port2}
4. exit
5. policy-map multi-match map_name
6. class map_name
7. inspect http [policy policy_map2 | url-logging]
8. end
9. copy running-config startup-config
### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode. You are now within configuration mode of the VFW application.</td>
</tr>
</tbody>
</table>

**Example:**
```
firewall/Admin# configure
Enter configuration commands, one per line. End with CNTL/Z.
firewall/Admin(config)#
```

| Step 2 class-map [match-all | match-any] map_name | Creates a Layer 3 and Layer 4 class map to classify network traffic passing through the VFW application for HTTP deep packet 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. |

**Example:**
```
firewall/Admin(config)# class-map match-all HTTP_INSPECT_L4CLASS
```

| Step 3 [line_number] match port (tcp | udp) (any | eq port_number | range port1 port2) | Specifies a match command as part of the Layer 3 and Layer 4 class map. Refer to “Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy” section on page VFC-136 for a list of available match commands. |

**Example:**
```
firewall/Admin(config-cmap)# match port tcp eq 80
```

| Step 4 exit | Exits class map configuration mode. |

**Example:**
```
firewall/Admin(config-cmap)# exit
firewall/Admin(config)#
```

| Step 5 policy-map multi-match map_name | Creates a Layer 3 and Layer 4 policy map and associates the Layer 7 HTTP deep packet inspection policy map to activate the operation. Specify the actions you want to apply to the Layer 3 and Layer 4 user-defined class map and, if appropriate, to the default class map. |

**Example:**
```
firewall/Admin(config)# policy-map multi-match HTTP_INSPECT_L4POLICY
```

| Step 6 class map_name | Associates the class map defined in Step 2 with the Layer 3 and Layer 4 policy map, and enters policy map class configuration mode. |

**Example:**
```
firewall/Admin(config-pmap)# class HTTP_INSPECT_L4CLASS
```

| Step 7 inspect http [policy policy_map | url-logging] | Associates the HTTP deep packet inspection policy map with the Layer 3 and Layer 4 class map being defined. For example, the HTTP deep packet inspection policy map created in “Configuring a Layer 7 HTTP Deep Packet Inspection Policy Map” section on page VFC-126 is associated with the Layer 3 and Layer 4 class map. |

**Example:**
```
firewall/Admin(config-pmap-c)# inspect http policy HTTP_INSPECT_L7POLICY
```
What to Do Next

You must attach the Layer 3 and Layer 4 traffic policy that you created in this task to an interface. See “Applying a Traffic Policy to an Interface” section on page VFC-131.

Applying a Traffic Policy to an Interface

After you have created the Layer 3 and Layer 4 traffic policy, you must attach it to a single interface or globally to all interfaces. This task describes how to attach the traffic policy to an interface.

Prerequisites

- You must have created a Layer 3 and Layer 4 traffic policy as described in “Configuring a Layer 3 and Layer 4 Traffic Policy for Layer 7 HTTP Deep Packet Inspection” section on page VFC-129.
- You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

SUMMARY STEPS

1. configure
2. interface interface_name
3. service-policy input policy_name
4. exit
5. copy running-config startup-config
DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> configure</td>
<td>Enters global configuration mode. You are now within configuration mode of the VFW application.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> interface interface_name</td>
<td>Enters interface configuration mode for a firewall interface.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong> service-policy input policy_name</td>
<td>Attaches the Layer 3 and Layer 4 traffic policy to an interface.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong> end</td>
<td>Exits configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong> copy running-config startup-config</td>
<td>(Optional) Saves your configuration changes to flash memory.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Configuring a Layer 7 FTP Command Inspection Policy

This section describes how to create a Layer 7 class map and policy map to be used for FTP command inspection by the VFW application, a security feature that prevents web browsers from sending embedded commands to the VFW application in FTP requests. Each FTP command must be acknowledged before the VFW application allows 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 VFW application. When the VFW application denies a command, it closes the connection.

This task describes how to perform the following main procedures:

- Create a Layer 7 class map for the inspection of FTP request commands
- Create and configure a Layer 7 policy map that enables FTP command inspection
- Create a Layer 3 and Layer 4 class map to classify network traffic for FTP command inspection
- Create a Layer 3 and Layer 4 policy map and associates the Layer 7 FTP command inspection policy map
- Attach the Layer 3 and Layer 4 traffic policy to an interface
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:

- Match statements in Layer 7 class maps
- Inline match statements in Layer 7 policy maps
- Header insertion and rewrite (including SSL URL rewrite) expressions in Layer 7 action lists

Prerequisites

You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

SUMMARY STEPS

1. Creates a Layer 7 class map for the inspection of FTP request commands
   1. configure
   2. class-map type ftp inspect match-any map_name
   3. [line_number] match request-method ftp_commands
   4. exit

2. Creates and configures a Layer 7 policy map that enables FTP command inspection
   5. policy-map type inspect ftp first-match map_name
   6. [line_number] match name request-method { appe | cdup | dele | get | help | mkd | put | rmd | rnfr | rnto | site | stou | syst }
   7. class map_name
   8. deny
   9. mask-reply
   10. exit

3. Creates a Layer 3 and Layer 4 class map to classify network traffic for FTP command inspection
   11. class-map match-all map_name
   12. [line_number] match port { tcp | udp } { any | eq port_number | range port1 port2 }
   13. exit

4. Creates a Layer 3 and Layer 4 policy map and associates the Layer 7 FTP command inspection policy map
   14. policy-map multi-match map_name
   15. class map_name
   16. inspect ftp [ strict policy policy_map ]
   17. exit
### How to Configure Application Protocol Inspection

Attaches the Layer 3 and Layer 4 traffic policy to an interface

18. `interface interface_name`
19. `service-policy input policy_name`
20. `exit`
21. `copy running-config startup-config`

#### DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>configure</code></td>
<td>Enters global configuration mode. You are now within configuration mode of the VFW application.</td>
</tr>
</tbody>
</table>

**Example:**
```
firewall/Admin# configure
Enter configuration commands, one per line.
End with CNTL/Z.
firewall/Admin(config)#
```

<table>
<thead>
<tr>
<th><strong>Step 2</strong> <code>class-map type ftp inspect match-any map_name</code></th>
<th>Creates a Layer 7 class map that is used for the inspection of FTP request commands.</th>
</tr>
</thead>
</table>

**Example:**
```
firewall/Admin(config)# class-map type ftp inspect match-any FTP_INSPECT_L7CLASS
firewall/Admin(config-cmap-ftp-insp)#
```

<table>
<thead>
<tr>
<th><strong>Step 3</strong> <code>[line_number] match request-method ftp_commands</code></th>
<th>Configures the Layer 7 class map to define FTP request command inspection decisions through the VFW application. The <strong>match request-method</strong> command identifies the FTP commands that you want filtered by the VFW application. Possible <code>ftp_commands</code> include <code>appo, cdup, dele, get, help, mkd, put, rmd, rnfr, rnto, site, stou, and syst</code>.</th>
</tr>
</thead>
</table>

**Example:**
```
firewall/Admin(config-cmap-ftp-insp)# match request-method mkdir
```

<table>
<thead>
<tr>
<th><strong>Step 4</strong> <code>exit</code></th>
<th>Exits class map configuration mode.</th>
</tr>
</thead>
</table>

**Example:**
```
firewall/Admin(config-if-mgmt)# exit
firewall/Admin#
```

<table>
<thead>
<tr>
<th><strong>Step 5</strong> <code>policy-map type inspect ftp first-match map_name</code></th>
<th>Creates and configures a Layer 7 policy map that enables FTP command inspection.</th>
</tr>
</thead>
</table>

**Example:**
```
firewall/Admin(config)# policy-map type inspect ftp first-match FTP_INSPECT_L7POLICY
```

- **first-match**—Specifies that the VFW application executes only the action specified against the first-matching classification.
- **map_name**—Specifies the name assigned to the policy map. Enter an unquoted text string with no spaces and a maximum of 64 alphanumeric characters.
### Command or Action

| Step 6 | \([\text{line\_number}]\) match name request-method \{append | \cdup | \dele | \get | \help | \mkd | \put | \rmd | \rnfr | \rnto | \site | \stou | \syst\} |
|---|---|
| **Example:** | firewall/Admin(config-pmap-ftp-ins)# match FTP_REQUEST_MATCH request-method mkdir |

Includes a single inline match criteria in the policy map without specifying a traffic class. 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** This command replaces the class-map definition and the commands in Step 7 to Step 9.

<table>
<thead>
<tr>
<th>Step 7</th>
<th>\textbf{class} map_name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-pmap-ftp-ins)# class FTP_INSPECT_L7CLASS</td>
</tr>
</tbody>
</table>

Associates a class map defined in Step 2 with the Layer 7 policy map, and enters policy map class configuration mode for you to define the actions you want to apply.

**Note** When a class map is used, the inline `match` command in Step 6 is not required.

<table>
<thead>
<tr>
<th>Step 8</th>
<th>deny or mask-reply</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-pmap-ftp-ins-c)# deny</td>
</tr>
</tbody>
</table>

The `deny` command denies the FTP request commands against the single inline `match` command or specified in the class map by resetting the FTP session.

The `mask-reply` command masks the system reply to the FTP SYST command by filtering sensitive information from the command output. This is applicable only to the FTP SYST command and its associated reply.

<table>
<thead>
<tr>
<th>Step 9</th>
<th>exit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin()# exit firewall/Admin(config-pmap-ftp-ins)#</td>
</tr>
</tbody>
</table>

Exits policy map class configuration mode.

<table>
<thead>
<tr>
<th>Step 10</th>
<th>\textbf{class-map match-all} map_name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config)# class-map match-all FTP_INSPECT_L4CLASS</td>
</tr>
</tbody>
</table>

Creates a Layer 3 and Layer 4 class map to classify network traffic passing through the VFW application for FTP command inspection.

| Step 11 | \([\text{line\_number}]\) match port \{tcp | udp\} \{any | eq port\_number | range port1 port2\} |
|---|---|
| **Example:** | firewall/Admin(config-cmap)# match port tcp eq 21 |

Specifies a `match` command as part of the Layer 3 and Layer 4 class map. Include one or more `match` commands as required. Refer to “Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy” section on page VFC-136 for a list of available `match` commands.

<table>
<thead>
<tr>
<th>Step 12</th>
<th>exit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-cmap)# exit firewall/Admin(config)#</td>
</tr>
</tbody>
</table>

Exits class map configuration mode.

<table>
<thead>
<tr>
<th>Step 13</th>
<th>\textbf{policy-map multi-match} map_name</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config)# policy-map multi-match FTP_STRICT_INSPECT_L4POLICY</td>
</tr>
</tbody>
</table>

Creates a Layer 3 and Layer 4 policy map and associates the Layer 7 FTP command inspection policy map to activate the operation.
### Configuring Application Protocol Inspection on the Virtual Firewall

**How to Configure Application Protocol Inspection**

This section describes how to create a Layer 3 and Layer 4 class map and policy map to classify network traffic passing through the VFW application to perform an applicable application protocol inspection traffic policy. The Layer 3 and Layer 4 traffic policy defines the Layer 3 and Layer 4 HTTP deep packet inspection traffic policy.

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 14</strong> <em>class map_name</em></td>
<td>Associates a class map defined in Step 10 with the Layer 7 FTP command inspection policy map, and enters policy map class configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-pmap)# class FTP_INSPECT_L4CLASS</td>
</tr>
<tr>
<td><strong>Step 15</strong> <em>inspect ftp [strict policy policy_map]</em></td>
<td>Specifies to examine the FTP protocol to verify the protocol behavior and identify unwanted or malicious traffic passing through the VFW application.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-pmap-c) inspect ftp strict policy FTP_INSPECT_L7POLICY</td>
</tr>
<tr>
<td><strong>Step 16</strong> <em>exit</em></td>
<td>Exits class map configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-pmap-c)# exit firewall/Admin(config-pmap)#</td>
</tr>
<tr>
<td><strong>Step 17</strong> <em>exit</em></td>
<td>Exits policy map configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-pmap)# exit firewall/Admin(config)#</td>
</tr>
<tr>
<td><strong>Step 18</strong> <em>interface interface_name</em></td>
<td>Enters interface configuration mode for an interface.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config)# interface management m1</td>
</tr>
<tr>
<td><strong>Step 19</strong> <em>service-policy input policy_name</em></td>
<td>Attaches the Layer 3 and Layer 4 traffic policy to the firewall interface and specifies the direction in which the policy is applied.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-if)# service-policy input FTP_INSPECT_L4POLICY</td>
</tr>
<tr>
<td><strong>Step 20</strong> <em>exit</em></td>
<td>Exits interface configuration mode.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin(config-if)# end firewall/Admin#</td>
</tr>
<tr>
<td><strong>Step 21</strong> <em>copy running-config startup-config</em></td>
<td>(Optional) Saves your configuration changes to flash memory.</td>
</tr>
<tr>
<td><strong>Example:</strong></td>
<td>firewall/Admin# copy running-config startup-config</td>
</tr>
</tbody>
</table>

### 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 passing through the VFW application to perform an applicable application protocol inspection traffic policy.
Configuring Application Protocol Inspection on the Virtual Firewall

How to Configure Application Protocol Inspection

This task describes how to configure a Layer 3 and Layer 4 inspection traffic policy.

**Prerequisites**

You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

**SUMMARY STEPS**

1. configure
2. class-map [match-all | match-any] map_name
3. [line_number] match access-list identifier
4. [line_number] match port {tcp | udp} {any | eq port_number | range port1 port2}
5. exit
6. policy-map multi-match map_name
7. class map_name
8. inspect {dns [maximum-length bytes]} | {ftp [strict policy policy_map]} | {http [policy policy_map | url-logging]} | {icmp [error]} | rtsp
9. exit
10. exit
11. copy running-config startup-config

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
</table>
| **Step 1**        | configure
|                   | Enters global configuration mode. You are now within configuration mode of the VFW application. |

**Example:**

```
firewall/Admin# configure
Enter configuration commands, one per line. End with CNTL/Z.
firewall/Admin(config)#
```

| **Step 2** | class-map [match-all | match-any] map_name
|------------|-------------------------------|
| **Example:**
| firewall/Admin(config)# class-map match-all DNS_INSPECT_L4CLASS
| firewall/Admin(config-cmap)# |

**Purpose:**

Creates a Layer 3 and Layer 4 class map to classify network traffic passing through the VFW application for DNS, FTP, HTTP, ICMP, and RTSP application protocol inspection.

- **match-all** (default)—Specifies to match all the criteria listed in the class map. This is typically used match commands of different types.
- **match-any**—Specifies that only one match criteria from the class map is required to match. This is typically used to match commands of the same type.
## Configuring Application Protocol Inspection on the Virtual Firewall

### Command or Action

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td><code>[line_number] match access-list identifier</code></td>
<td>(Optional) Configures 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 VFW application allows the matching result. If it is a deny entry, the VFW application blocks the matching result. Refer to “Configuring Security Access Control Lists on the Virtual Firewall” for details about the creating access control lists.</td>
</tr>
</tbody>
</table>
| 4    | `[line_number] match port {tcp | udp} {any | eq port_number | range port1 port2}` | Specifies a TCP or UDP port number or port range as the Layer 3 and Layer 4 network traffic matching criteria. Keywords and arguments are:  
- `tcp` | `udp` — Specifies the protocol, TCP or UDP.  
- `any` — Wildcard value for the TCP or UDP port number. With any used in place of either the `eq` or `range` values, packets from any incoming port 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 VFW application to include all ports. Alternatively, you can enter the name of a well-known TCP port as listed in Table 13 or a well-known UDP port as listed in Table 14.  
- `range port1 port2` — Specifies a port range to use for the TCP or UDP port. Valid port ranges are 0 to 65535. A value of 0 instructs the VFW application to match all ports. |
| 5    | `exit` | Exits class map configuration mode. |
| 6    | `policy-map multi-match map_name` | Creates and configures a Layer 3 and Layer 4 policy map. |
| 7    | `class map_name` | Associates a class map defined in Step 2 with the Layer 3 and Layer 4 policy map, and enters policy map class configuration mode. |

**Example:**

```
firewall/Admin(config-cmap)# match access-list INBOUND_ACL1

firewall/Admin(config-cmap)# match port udp eq domain

firewall/Admin(config-cmap)# exit

firewall/Admin(config)#

firewall/Admin(config)# policy-map multi-match DNS_INSPECT_L4POLICY

firewall/Admin(config-pmap)# class DNS_INSPECT_L4CLASS
```
Configuring Application Protocol Inspection on the Virtual Firewall

How to Configure Application Protocol Inspection

Table 13 lists the well-known TCP port numbers and key words.

<table>
<thead>
<tr>
<th>Port</th>
<th>Port Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>53</td>
<td>Specifies Domain Name System</td>
</tr>
<tr>
<td>ftp</td>
<td>21</td>
<td>Specifies File Transfer Protocol</td>
</tr>
<tr>
<td>ftp-data</td>
<td>20</td>
<td>Specifies File Transfer Protocol Data</td>
</tr>
<tr>
<td>http</td>
<td>80</td>
<td>Specifies Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>https</td>
<td>443</td>
<td>Specifies HTTP over SSL protocol</td>
</tr>
<tr>
<td>irc</td>
<td>194</td>
<td>Specifies Internet Relay Chat protocol</td>
</tr>
<tr>
<td>matip-a</td>
<td>350</td>
<td>Specifies Matip Type A protocol</td>
</tr>
<tr>
<td>nntp</td>
<td>119</td>
<td>Specifies Network News Transport Protocol</td>
</tr>
<tr>
<td>pop2</td>
<td>109</td>
<td>Specifies Post Office Protocol v2</td>
</tr>
<tr>
<td>pop3</td>
<td>110</td>
<td>Specifies Post Office Protocol v3</td>
</tr>
<tr>
<td>rtsp</td>
<td>554</td>
<td>Specifies Real Time Stream Control Protocol</td>
</tr>
<tr>
<td>smtp</td>
<td>25</td>
<td>Specifies Simple Mail Transfer Protocol</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>Specifies Telnet protocol</td>
</tr>
<tr>
<td>www</td>
<td>80</td>
<td>Specifies World Wide Web</td>
</tr>
</tbody>
</table>

Example:

```
firewall/Admin(config-pmap-c)# inspect dns maximum-length 1000
```

Specifies to examine DNS, FTP, HTTP, ICMP or RTSP protocols to verify the protocol behavior and identify unwanted or malicious traffic passing through the VFW application.

Refer to the “Configuration Tips: Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions” section on page VFC-140 for more information.

Example:

```
firewall/Admin(config-pmap-c)# exit
firewall/Admin(config-pmap-c)# exit
firewall/Admin(config-pmap)#
```

Exits policy map configuration mode.

Example:

```
firewall/Admin(config-pmap)# exit
firewall/Admin(config)#
```

Exits policy map configuration mode.

Example:

```
f handgun/Admin# copy running-config startup-config
```

(Optional) Saves your configuration changes to flash memory.
Table 14 lists the well-known UDP port numbers and key words.

<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>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>

Configuration Tips: Defining Layer 3 and Layer 4 Application Protocol Inspection Policy Actions

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, and RTSP to verify the protocol behavior and identify unwanted or malicious traffic passing through the VFW application.

If you intend to perform Layer 7 application inspection of network traffic, first create a Layer 7 policy as described below:

- To perform the deep packet inspection of Layer 7 HTTP application traffic by the VFW application, 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 using the Layer 3 and Layer 4 `inspect http` command.
- To perform the request inspection of FTP commands, first create a Layer 7 policy using the `policy-map type inspect ftp` command (see the “Configuring a Layer 7 FTP Command Inspection Policy” section). You nest the Layer 7 FTP inspection policy 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 be associated only within a Layer 3 and Layer 4 policy map. Only a Layer 3 and Layer 4 policy map can be applied to an interface or applied globally to all interfaces in the same context; a Layer 7 policy map cannot be directly applied on an interface.

If you do not specify a Layer 7 HTTP or FTP policy map, the VFW application performs a general set of Layer 3 and Layer 4 HTTP or FTP protocol fixup actions. For example, the VFW application performs strict HTTP.

Applying a Traffic Policy to an Interface

After you have created a traffic policy, you must attach it to a single interface or globally to all interfaces. This task describes how to attach the traffic policy to an interface.

Prerequisites

- You must have created a Layer 3 and Layer 4 traffic policy as described in “Configuring a Layer 3 and Layer 4 Application Protocol Inspection Traffic Policy” section on page VFC-136.
You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

**SUMMARY STEPS**

1. configure
2. interface *interface_name*
3. service-policy input *policy_name*
4. end
5. copy running-config startup-config

**DETAILED STEPS**

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong>  configure</td>
<td>Enters global configuration mode. You are now within configuration mode of the VFW application.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin# configure</td>
<td></td>
</tr>
<tr>
<td>Enter configuration commands, one per line.</td>
<td></td>
</tr>
<tr>
<td>End with CNTL/Z.</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin(config)#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong>  interface <em>interface_name</em></td>
<td>Enters interface configuration mode for a firewall interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin(config)# interface i1</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong>  service-policy input <em>policy_name</em></td>
<td>Attaches the Layer 3 and Layer 4 traffic policy to an interface.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin(config-if)# service-policy input HTTP_INSPECT_L4POLICY</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong>  end</td>
<td>Exits configuration mode.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin(config-if)# end</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin#</td>
<td></td>
</tr>
<tr>
<td><strong>Step 5</strong>  copy running-config startup-config</td>
<td>(Optional) Saves your configuration changes to flash memory.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td>firewall/Admin# copy running-config startup-config</td>
<td></td>
</tr>
</tbody>
</table>
Displaying Application Protocol Inspection Statistics and Service Policy Information

This task illustrates how to use the `show` commands that display application protocol inspection statistics and service policy configuration information. There is no particular order to the steps in this procedure.

Prerequisites

You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

SUMMARY STEPS

1. `show stats inspect http`
2. `clear stats inspect http`
3. `show service-policy name`
4. `clear service-policy name`

DETAILED STEPS

<table>
<thead>
<tr>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong> <code>show stats inspect http</code></td>
<td>(Optional) Displays HTTP protocol inspection statistics.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>firewall/Admin# show stats inspect http</code></td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong> <code>clear stats inspect http</code></td>
<td>(Optional) Clears the HTTP protocol inspection statistics.</td>
</tr>
<tr>
<td>Example:</td>
<td></td>
</tr>
<tr>
<td><code>firewall/Admin# clear stats inspect http</code></td>
<td></td>
</tr>
</tbody>
</table>
| **Step 3** `show service-policy name` | (Optional) Displays service policy statistics. The statistics that appear in the output are dependent on the configuration of the associated Layer 3 and Layer 4 policy map. The following information is displayed:  
  - Interface to which the policy is applied  
  - Class map associated with the policy  
  Note The VFW application updates the counters that the `show service-policy` command displays after the applicable connections are closed. |
| Example:                           |                                                                         |
| `firewall/Admin# show service-policy HTTP_INSPECT_L4POLICY` |                                                                         |
| **Step 4** `clear service-policy name` | (Optional) Clears the service policy statistics.                        |
| Example:                           |                                                                         |
| `firewall/Admin# clear service-policy HTTP_INSPECT_L4POLICY` |                                                                         |
Examples

The following example illustrates sample output from the `show stats inspect http` command.

```
firewall/Admin# show stats inspect http
+------------------------------------------+
| HTTP Inspect statistics                  |
+------------------------------------------+
Total request/response : 0
Total allow decisions : 0
Total drop decisions : 0
Total logging decisions : 0
```

The following example displays service policy statistics for the HTTP_INSPECT_L4POLICY policy map:

```
firewall/Admin# show service-policy HTTP_INSPECT_L4POLICY
Status : ACTIVE
Description: HTTP protocol deep inspection of incoming traffic
-----------------------------------------
Interface: management ctx1
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
```

The following example displays service policy statistics for the FTP_INSPECT_L4POLICY policy map:

```
firewall/Admin# show service-policy FTP_INSPECT_L4POLICY
Status : ACTIVE
Description: FTP command inspection of incoming traffic
-----------------------------------------
Context Global Policy:
  service-policy: FTP_INSPECT_L4POLICY
  class: class-default
  inspect ftp:
    strict ftp: ENABLED
    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
L7 policy: FTP_INSPECT_L4POLICY
  TotalReplyMasked : 0 TotalDropped: 0
```

The following example displays service policy statistics for the APP_INSPECT_L4POLICY policy map:

```
firewall/Admin# show service-policy APP_INSPECT_L4POLICY
Status : ACTIVE
-----------------------------------------
```
How to Configure an HTTP Parameter Map for Use in a Layer 3 and Layer 4 Policy Map

- Configuring an HTTP Parameter Map, page VFC-144

Configuring an HTTP Parameter Map

A parameter map is a means 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. This task describes how to configure an HTTP parameter map.

Prerequisites

You must attach from the route processor to the VFW application before you can perform this task. See the “Attaching to the VFW Application” section on page VFC-16.

SUMMARY STEPS

1. configure
2. parameter-map type http `name`
3. case-insensitive
4. set header-maxparse-length `bytes`
5. set content-maxparse-length `bytes`
6. exit
7. policy-map multi-match `map_name`
8. class `map_name`
9. appl-parameter http advanced-options `name`
10. copy running-config startup-config
## DETAILED STEPS

<table>
<thead>
<tr>
<th>Step</th>
<th>Command or Action</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>configure</td>
<td>Enters global configuration mode. You are now within configuration mode of the VFW application.</td>
</tr>
</tbody>
</table>
|      | **Example:**      | firewall/Admin# configure  
Enter configuration commands, one per line.  
End with CNTL/Z.  
firewall/Admin(config)# |
| 2    | parameter-map type http name | Configures advanced HTTP behavior for HTTP deep packet inspection. |
|      | **Example:**      | firewall/Admin(config)# parameter-map type  
http HTTP_PARAM_MAP1  
firewall/Admin(config-parammap-http)# |
| 3    | case-insensitive  | Enables case-insensitive HTTP matching. With case-insensitive matching enabled, uppercase and lowercase letters are considered the same. When case sensitivity is disabled, it applies to:  
• HTTP header names and values  
• URL strings  
• HTTP content inspection |
|      | **Example:**      | firewall/Admin(config-parammap-http)#  
case-insensitive |
| 4    | set header-maxparse-length bytes | Configures the maximum number of bytes to parse in HTTP headers. Enter an integer from 1 to 65535. The default is 2048 bytes. |
|      | **Example:**      | firewall/Admin(config-parammap-http)# set  
header-maxparse-length 8192 |
| 5    | set content-maxparse-length bytes | Configures the maximum number of bytes to parse in HTTP content. Enter an integer from 1 to 65535. The default is 4096 bytes. |
|      | **Example:**      | firewall/Admin(config-parammap-http)# set  
content-maxparse-length 8192 |
| 6    | exit              | Exits parameter-map configuration mode. |
|      | **Example:**      | firewall/Admin(config-parammap-http)# exit  
firewall/Admin(config)# |
| 7    | policy-map multi-match map_name | Creates and configures a Layer 3 and Layer 4 policy map. |
|      | **Example:**      | firewall/Admin(config)# policy-map multi-match  
HTTP_INSPECT_L4POLICY |
Configuration Examples for Application Protocol Inspection

This section provides the following configuration examples:

- Layer 7 HTTP Deep Inspection Policy Configuration: Example, page VFC-146
- Layer 7 FTP Inspection Policy Configuration: Example, page VFC-147

Layer 7 HTTP Deep Inspection Policy Configuration: Example

Create a Layer 7 class map that is used for the deep packet inspection of HTTP traffic

```
firwall/AAA(config)# class-map type http inspect match-any HTTP_INSPECT_L7CLASS
firwall/AAA(config-cmap-http-insp)# match header length request eq 200
firwall/AAA(config-cmap-http-insp)# match header Host header-value .*mycompanyexample.com
firwall/AAA(config-cmap-http-insp)# match url length eq 10000
firwall/AAA(config-cmap-http-insp)# match url .*.gif
```

Create and configure a Layer 7 policy map that enables deep packet inspection of the HTTP protocol

```
firwall/AAA(config)# policy-map type inspect http all-match HTTP_INSPECT_L7POLICY
firwall/AAA(config-pmap-http)# class HTTP_INSPECT_L7CLASS
firwall/AAA(config-pmap-http-c)# permit
firwall/AAA(config-pmap-http-c)# exit
firwall/AAA(config-pmap-http-c)# exit
firwall/AAA(config)#
```

Configure Layer 3 and Layer 4 class map to classify network traffic for HTTP deep packet inspection

```
firwall/AAA(config)# class-map match-all HTTP_INSPECT_L4CLASS
firwall/AAA(config-cmap)# description HTTP protocol deep inspection of incoming traffic
firwall/AAA(config-cmap)# match port tcp eq 80
firwall/AAA(config-cmap)# exit
firwall/AAA(config)#
```
Create a Layer 3 and Layer 4 policy map and associate the Layer 7 HTTP deep packet inspection policy map

```
firewall/Admin(config)# policy-map multi-match HTTP_INSPECT_L4POLICY
firewall/Admin(config-pmap)# class HTTP_INSPECT_L4CLASS
firewall/Admin(config-pmap-c)# inspect http policy HTTP_INSPECT_L7POLICY
firewall/Admin(config-pmap-c)# exit
firewall/Admin(config-pmap)# exit
```

Attach the Layer 3 and Layer 4 traffic policy to an interface

```
firewall/Adminhost1/Admin(config)# interface interface_name
firewall/Admin(config-if)# service-policy input HTTP_INSPECT_L4POLICY
```

Layer 7 FTP Inspection Policy Configuration: Example

Create a Layer 7 class map for the inspection of FTP request commands

```
firewall/Admin# configure
firewall/Admin(config)# class-map type ftp inspect match-any FTP_INSPECT_L7CLASS
firewall/Admin(config-cmap-ftp-insp)# match request-method mkdir
firewall/Admin(config-cmap-ftp-insp)# exit
```

Create and configure a Layer 7 policy map that enables FTP command inspection

```
firewall/Admin(config)# policy-map type inspect ftp first-match FTP_INSPECT_L7POLICY
firewall/Admin(config-pmap-ftp-ins)# class FTP_INSPECT_L7CLASS
firewall/Admin(config-pmap-ftp-ins-c)# deny
firewall/Admin(config-pmap-ftp-ins-c)# exit
```

Create a Layer 3 and Layer 4 class map to classify network traffic passing through the VFW application for FTP command inspection

```
firewall/Admin(config)# class-map match-all FTP_INSPECT_L4CLASS
firewall/Admin(config-cmap)# description FTP command inspection of incoming traffic
firewall/Admin(config-cmap)# match port tcp eq 21
firewall/Admin(config-cmap)# exit
```

Create a Layer 3 and Layer 4 policy map and associate the Layer 7 FTP command inspection policy map

```
firewall/Admin(config)# policy-map multi-match FTP_STRRICT_INSPECT_L4POLICY
firewall/Admin(config-pmap)# class FTP_INSPECT_L4CLASS
firewall/Admin(config-pmap-c)# inspect ftp strict policy FTP_INSPECT_L7POLICY
firewall/Admin(config-pmap-c)# exit
firewall/Admin(config)#
```

Attach the Layer 3 and Layer 4 traffic policy to an interface

```
firewall/Admin(config)# interface interface_name
firewall/Admin(config-if)# service-policy input FTP_INSPECT_L4POLICY
```

Additional References

The following sections provide references related to application protocol inspection.
Related Documents

<table>
<thead>
<tr>
<th>Related Topic</th>
<th>Document Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual firewall interface command syntax</td>
<td>“Interface Commands on the Virtual Firewall” chapter in Cisco IOS XR Virtual Firewall Command Reference</td>
</tr>
<tr>
<td>Virtual firewall policy map command syntax</td>
<td>“Policy Map Commands on the Virtual Firewall” chapter in Cisco IOS XR Virtual Firewall Command Reference</td>
</tr>
<tr>
<td>Virtual firewall class map command syntax</td>
<td>“Class Map Commands on the Virtual Firewall” chapter in Cisco IOS XR Virtual Firewall Command Reference</td>
</tr>
</tbody>
</table>

Standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.</td>
<td>—</td>
</tr>
</tbody>
</table>

MIBs

<table>
<thead>
<tr>
<th>MIBs</th>
<th>MIBs Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: <a href="http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml">http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml</a></td>
</tr>
</tbody>
</table>

RFCs

<table>
<thead>
<tr>
<th>RFCs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.</td>
</tr>
</tbody>
</table>
## Technical Assistance

<table>
<thead>
<tr>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.</td>
<td><a href="http://www.cisco.com/techsupport">http://www.cisco.com/techsupport</a></td>
</tr>
</tbody>
</table>