



HTTP Inspection Engine

Last Updated: January 19, 2012

The HTTP Inspection Engine feature allows users to configure their Cisco IOS Firewall to detect and prohibit HTTP connections--such as tunneling over port 80, unauthorized request methods, and non-HTTP compliant file transfers--that are not authorized within the scope of the security policy configuration. Tunneling unauthorized protocols through port 80 and over HTTP exposes a network to significant security risks.

The Cisco IOS Firewall can now be configured with a security policy that adheres to the following tasks:

- Allowing specific traffic targeted for port 80 to traverse the firewall. The traffic is inspected for protocol conformance and for the types of HTTP commands that are allowed or disallowed.
- Denying specific traffic targeted for port 80 that does not comply to HTTP traffic standards. The firewall is enabled to drop the packet, reset the connection, and send a syslog message, as appropriate.
- [Finding Feature Information, page 1](#)
- [Restrictions for HTTP Inspection Engine, page 2](#)
- [Information About HTTP Inspection Engine, page 2](#)
- [How to Define and Apply an HTTP Application Policy to a Firewall for Inspection, page 2](#)
- [Configuration Examples for Setting Up an HTTP Inspection Engine, page 10](#)
- [Additional References, page 11](#)
- [Feature Information for Setting Up an HTTP Inspection Engine, page 12](#)

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.



Restrictions for HTTP Inspection Engine

The Cisco 831 router with 48M RAM does not have enough memory to support this feature.

Information About HTTP Inspection Engine

Before configuring an application firewall to detect and police specific traffic targeted for port 80, you should understand the following concepts:

- [What Is a Security Policy, page 2](#)
- [Cisco IOS HTTP Application Policy Overview, page 2](#)

What Is a Security Policy

The application firewall uses a security policy, which consists of a collection of static signatures, to detect security violations. A static signature is a collection of parameters that specify protocol conditions that must be met before an action is taken. (For example, a signature may specify that an HTTP data stream containing the POST method must reset the connection.) These protocol conditions and reactions are defined by the end user via the command-line interface (CLI) to form a security policy.

Cisco IOS HTTP Application Policy Overview

HTTP uses port 80 to transport Internet web services, which are commonly used on the network and rarely challenged with regards to their legitimacy and conformance to standards. Because port 80 traffic is typically allowed through the network without being challenged, many application developers are leveraging HTTP traffic as an alternative transport protocol in which to enable their application to travel through or even bypass the firewall.

Most firewalls provide only packet filtering capabilities that simply permit or deny port 80 traffic without inspecting the data stream; the Cisco IOS application firewall for HTTP performs packet inspection as follows:

- Detects HTTP connections that are not authorized within the scope of the security policy configuration.
- Detects users who are tunneling applications through port 80.

If the packet is not in compliance with the HTTP protocol, it will be dropped, the connection will be reset, and a syslog message will be generated, as appropriate.

How to Define and Apply an HTTP Application Policy to a Firewall for Inspection

- [Defining an HTTP Application Policy, page 3](#)
- [Applying an HTTP Application Policy to a Firewall for Inspection, page 7](#)

Defining an HTTP Application Policy

Use this task to create an HTTP application firewall policy.



Note

Although application firewall policies are defined in global configuration mode, only one global policy for a given protocol is allowed per interface.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. `appfw policy-name policy-name`
4. **application** *protocol*
5. `strict-http action {reset | allow} [alarm]`
6. `content-length {min bytes max bytes | min bytes | max bytes} action {reset | allow} [alarm]`
7. `content-type-verification [match-req-resp] action {reset | allow} [alarm]`
8. `max-header-length {request bytes response bytes} action {reset | allow} [alarm]`
9. `max-uri-length bytes action {reset | allow} [alarm]`
10. `request method {rfc rfc-method | extension extension-method} action {reset | allow} [alarm]`
11. `port-misuse {p2p | tunneling | im | default} action {reset | allow} [alarm]`
12. `transfer-encoding type {chunked | compress | deflate | gzip | identity | default} action {reset | allow} [alarm]`
13. **timeout** *seconds*
14. `audit-trail {on | off}`
15. **exit**
16. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>enable</p> <p>Example:</p> <pre>Router> enable</pre>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	<p>configure terminal</p> <p>Example:</p> <pre>Router# configure terminal</pre>	<p>Enters global configuration mode.</p>

Command or Action	Purpose
<p>Step 3 appfw policy-name policy-name</p> <p>Example:</p> <pre>Router(config)# appfw policy-name mypolicy</pre>	<p>Defines an application firewall policy and puts the router in application firewall policy configuration mode.</p>
<p>Step 4 application <i>protocol</i></p> <p>Example:</p> <pre>Router(cfg-appfw-policy)# application http</pre>	<p>Allows you to configure inspection parameters for a given protocol. Currently, only HTTP traffic can be inspected.</p> <ul style="list-style-type: none"> • <i>protocol</i> --Specify the http keyword. <p>This command puts you in appfw-policy-<i>protocol</i> configuration mode, where “<i>protocol</i>” is dependent upon the specified protocol. Because only HTTP can be specified, the configuration mode is appfw-policy-http.</p>
<p>Step 5 strict-http action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# strict- http action allow alarm</pre>	<p>(Optional) Allows HTTP messages to pass through the firewall or resets the TCP connection when HTTP noncompliant traffic is detected.</p>
<p>Step 6 content-length {min bytes max bytes min bytes max bytes} action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# content- length max 1 action allow alarm</pre>	<p>(Optional) Permits or denies HTTP traffic through the firewall on the basis of message size.</p> <ul style="list-style-type: none"> • min max bytes--Minimum or maximum content length, in bytes, allowed per message. Number of bytes range: 0 to 65535.
<p>Step 7 content-type-verification [match-req-resp] action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# content- type- verification match-req-resp action allow alarm</pre>	<p>(Optional) Permits or denies HTTP traffic through the firewall on the basis of content message type.</p>

Command or Action	Purpose
<p>Step 8 max-header-length {request bytes response bytes} action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# max-header-length request 1 response 1 action allow alarm</pre>	<p>(Optional) Permits or denies HTTP traffic on the basis of the message header length.</p> <ul style="list-style-type: none"> • <i>bytes</i> --Number of bytes ranging from 0 to 65535.
<p>Step 9 max-uri-length bytes action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# max-uri-length 1 action allow alarm</pre>	<p>(Optional) Permits or denies HTTP traffic on the basis of the URI length in the request message.</p>
<p>Step 10 request method {rfc rfc-method extension extension-method} action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# request-method rfc default action allow alarm</pre>	<p>(Optional) Permits or denies HTTP traffic according to either the request methods or the extension methods.</p> <ul style="list-style-type: none"> • rfc --Specifies that the supported methods of RFC 2616, <i>Hypertext Transfer Protocol--HTTP/1.1</i>, are to be used for traffic inspection. • <i>rfc-method</i> --Any one of the following RFC 2616 methods can be specified: connect, default, delete, get, head, options, post, put, trace. • extension --Specifies that the extension methods are to be used for traffic inspection. • <i>extension-method</i> --Any one of the following extension methods can be specified: copy, default, edit, getattribute, getproperties, index, lock, mkdir, move, revadd, revlabel, revlog, save, setattribute, startrev, stoprev, unedit, unlock.
<p>Step 11 port-misuse {p2p tunneling im default} action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# port-misuse default action allow alarm</pre>	<p>(Optional) Permits or denies HTTP traffic through the firewall on the basis of specified applications in the HTTP message.</p> <ul style="list-style-type: none"> • p2p --Peer-to-peer protocol applications subject to inspection: Kazaa and Gnutella. • tunneling --Tunneling applications subject to inspection: HTTPPort/HTTPHost, GNU Httptunnel, GotoMyPC, Firethru, Http-tunnel.com Client • im --Instant messaging protocol applications subject to inspection: Yahoo Messenger. • default --All applications are subject to inspection.

Command or Action	Purpose
<p>Step 12 transfer-encoding type {chunked compress deflate gzip identity default} action {reset allow} [alarm]</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# transfer-encoding type default action allow alarm</pre> <p>Example:</p>	<p>(Optional) Permits or denies HTTP traffic according to the specified transfer-encoding of the message.</p> <ul style="list-style-type: none"> • chunked --Encoding format (specified in RFC 2616, <i>Hypertext Transfer Protocol--HTTP/1</i>) in which the body of the message is transferred in a series of chunks; each chunk contains its own size indicator. • compress --Encoding format produced by the UNIX “compress” utility. • deflate --“ZLIB” format defined in RFC 1950, <i>ZLIB Compressed Data Format Specification version 3.3</i> , combined with the “deflate” compression mechanism described in RFC 1951, <i>DEFLATE Compressed Data Format Specification version 1.3</i> . • gzip --Encoding format produced by the “gzip” (GNU zip) program. • identity --Default encoding, which indicates that no encoding has been performed. • default --All of the transfer encoding types.
<p>Step 13 timeout <i>seconds</i></p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# timeout 60</pre>	<p>(Optional) Overrides the global TCP idle timeout value for HTTP traffic.</p> <p>Note If this command is not issued, the default value specified via the ip inspect tcp idle-timecommand will be used.</p>
<p>Step 14 audit-trail {on off}</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# audit- trail on</pre>	<p>(Optional) Turns audit trail messages on or off.</p> <p>Note If this command is not issued, the default value specified via the ip inspect audit-trailcommand will be used.</p>
<p>Step 15 exit</p> <p>Example:</p> <pre>Router(cfg-appfw-policy-http)# exit</pre>	<p>Exits cfg-appfw-policy-http configuration mode.</p>
<p>Step 16 exit</p> <p>Example:</p> <pre>Router(cfg-appfw-policy)# exit</pre>	<p>Exits cfg-appfw-policy configuration mode.</p>

- [What to Do Next, page 7](#)

What to Do Next

After you have successfully defined an application policy for HTTP traffic inspection, you must apply the policy to an inspection rule. Thereafter, the inspection rule must be applied to an interface. For information on completing this task, see the section “Applying an HTTP Application Policy to a Firewall for Inspection.”

Applying an HTTP Application Policy to a Firewall for Inspection

Use this task to apply an HTTP application policy to an inspection rule, followed by applying the inspection rule to an interface.



Note

An application policy can coexist with other inspection protocols (for example, an HTTP policy and an FTP policy can coexist).

You must have already defined an application policy (as shown in the section “Defining an HTTP Application Policy”).

or

```
show ip inspect {name inspection-name | config | interfaces | session [detail] | statistics | all}
```

SUMMARY STEPS

1. enable
2. configure terminal
3. ip inspect name *inspection-name* appfw *policy-name*
4. ip inspect name *inspection-name* http [alert {on | off}] [audit-trail {on | off}] [timeout *seconds*]
5. interface *type number*
6. ip inspect *inspection-name* in | out}
7. exit
8. exit
9. show appfw configuration [name]

DETAILED STEPS

Command or Action	Purpose
Step 1 enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.

Command or Action	Purpose
<p>Step 2 <code>configure terminal</code></p> <p>Example:</p> <pre>Router# configure terminal</pre>	<p>Enters global configuration mode.</p>
<p>Step 3 <code>ip inspect name inspection-name appfw policy-name</code></p> <p>Example:</p> <pre>Router(config)# ip inspect name firewall appfw mypolicy</pre>	<p>Defines a set of inspection rules for the application policy.</p> <ul style="list-style-type: none"> • <i>policy-name</i> --Must match the policy name specified via the appfw policy-name command.
<p>Step 4 <code>ip inspect name inspection-name http [alert {on off}] [audit-trail {on off}] [timeout seconds]</code></p> <p>Example:</p> <pre>Router(config)# ip inspect name firewall http</pre>	<p>Defines a set of inspection rules that is to be applied to all HTTP traffic.</p> <ul style="list-style-type: none"> • The <i>inspection-name</i> argument must match the <i>inspection-name</i> argument specified in Step 3.
<p>Step 5 <code>interface type number</code></p> <p>Example:</p> <pre>Router#(config)# interface FastEthernet0/0</pre>	<p>Configures an interface type and enters interface configuration mode.</p>
<p>Step 6 <code>ip inspect inspection-name in out</code></p> <p>Example:</p> <pre>Router#(config-if)# ip inspect firewall in</pre>	<p>Applies the inspection rules (defined in Step 3 and Step 4) to all traffic entering the specified interface.</p> <ul style="list-style-type: none"> • The <i>inspection-name</i> argument must match the inspection name defined via the ip inspect name command.
<p>Step 7 <code>exit</code></p> <p>Example:</p> <pre>Router#(config-if)# exit</pre>	<p>Exits interface configuration mode.</p>
<p>Step 8 <code>exit</code></p> <p>Example:</p> <pre>Router(config)# exit</pre>	<p>Exits global configuration mode.</p>

Command or Action	Purpose
<p>Step 9 show appfw configuration [name]</p> <p>Example:</p> <pre>Router# show appfw configuration</pre> <p>Example:</p> <p>or</p> <p>Example:</p> <pre>show ip inspect {name inspection-name config interfaces session [detail] statistics all}</pre> <p>Example:</p> <pre>Router# show ip inspect config</pre>	<p>(Optional) Displays application firewall policy configuration information.</p> <p>(Optional) Displays firewall-related configuration information.</p>

- [Troubleshooting Tips, page 9](#)

Troubleshooting Tips

To help troubleshoot the application firewall configuration, issue the following application-firewall specific debug command: **debug appfw{application protocol | function-trace | object-creation | object-deletion | events | timers | detailed}**.

The following sample configuration shows how to configure an HTTP policy with application firewall debugging enabled:

```
Router(config)# appfw policy-name myPolicyAPPFW FUNC:appfw_policy_find
APPFW FUNC:appfw_policy_find -- Policy myPolicy is not found
APPFW FUNC:appfw_policy_alloc
APPFW FUNC:appfw_policy_alloc -- policy_alloc 0x65727278
APPFW FUNC:appfw_policy_alloc -- Policy 0x65727278 is set to valid
APPFW FUNC:appfw_policy_alloc -- Policy myPolicy has been created
APPFW FUNC:appfw_policy_command -- memlock policy 0x65727278

! Debugging sample for application (HTTP) creation

Router(cfg-appfw-policy)# application httpAPPFW FUNC:appfw_http_command
APPFW FUNC:appfw_http_appl_find
APPFW FUNC:appfw_http_appl_find -- Application not found
APPFW FUNC:appfw_http_appl_alloc
APPFW FUNC:appfw_http_appl_alloc -- appl_http 0x64D7A25C
APPFW FUNC:appfw_http_appl_alloc -- Application HTTP parser structure 64D7A25C created
! Debugging sample for HTTP-specific application inspection
Router(cfg-appfw-policy-http)#
Router(cfg-appfw-policy-http)# strict-http action reset alarm
APPFW FUNC:appfw_http_subcommand
APPFW FUNC:appfw_http_subcommand -- strict-http cmd turned on
Router# debug appfw detailed
APPFW Detailed Debug debugging is on
```

```
fw7-7206a#debug appfw object-creation
APPFW Object Creations debugging is on
fw7-7206a#debug appfw object-deletion
APPFW Object Deletions debugging is on
```

Configuration Examples for Setting Up an HTTP Inspection Engine

- [Example Setting Up and Verifying an HTTP Inspection Engine, page 10](#)

Example Setting Up and Verifying an HTTP Inspection Engine

The following example show how to define the HTTP application firewall policy “mypolicy.” This policy includes all supported HTTP policy rules. This example also includes sample output from the **show appfw configuration** and **show ip inspect config** commands, which allow you to verify the configured setting for the application policy.

```
! Define the HTTP policy.
appfw policy-name mypolicy
  application http
    strict-http action allow alarm
    content-length maximum 1 action allow alarm
    content-type-verification match-req-rsp action allow alarm
    max-header-length request 1 response 1 action allow alarm
    max-uri-length 1 action allow alarm
    port-misuse default action allow alarm
    request-method rfc put action allow alarm
    transfer-encoding type default action allow alarm
!
!
! Apply the policy to an inspection rule.
ip inspect name firewall appfw mypolicy
ip inspect name firewall http
!
!
! Apply the inspection rule to all HTTP traffic entering the FastEthernet0/0 interface.
interface FastEthernet0/0
  ip inspect firewall in
!
!
! Issue the show appfw configuration
  command and the show ip inspect config
  command after the inspection rule “mypolicy” is applied to all incoming HTTP traffic on
  the FastEthernet0/0 interface.
!
Router# show appfw configuration

Application Firewall Rule configuration
  Application Policy name mypolicy
    Application http
      strict-http action allow alarm
      content-length minimum 0 maximum 1 action allow alarm
      content-type-verification match-req-rsp action allow alarm
      max-header-length request length 1 response length 1 action allow alarm
      max-uri-length 1 action allow alarm
      port-misuse default action allow alarm
      request-method rfc put action allow alarm
      transfer-encoding default action allow alarm
Router# show ip inspect config

Session audit trail is disabled
Session alert is enabled
one-minute (sampling period) thresholds are [400:500] connections
```

```

max-incomplete sessions thresholds are [400:500]
max-incomplete tcp connections per host is 50. Block-time 0 minute.
tcp synwait-time is 30 sec -- tcp finwait-time is 5 sec
tcp idle-time is 3600 sec -- udp idle-time is 30 sec
dns-timeout is 5 sec
Inspection Rule Configuration
Inspection name firewall
http alert is on audit-trail is off timeout 3600

```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Firewall commands: complete command syntax, command mode, defaults, usage guidelines, and examples	<i>Cisco IOS Security Command Reference</i>

Standards

Standards	Title
No new or modified standards are supported by this feature.	--

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
RFC 2616	Hypertext Transfer Protocol -- HTTP/1.1

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Setting Up an HTTP Inspection Engine

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1 **Feature Information for Setting Up an HTTP Inspection Engine**

Feature Name	Releases	Feature Information
Setting Up an HTTP Inspection Engine	12.3(14)T	<p>The HTTP Inspection Engine feature allows users to configure their Cisco IOS Firewall to detect and prohibit HTTP connections--such as tunneling over port 80, unauthorized request methods, and non-HTTP compliant file transfers--that are not authorized within the scope of the security policy configuration. Tunneling unauthorized protocols through port 80 and over HTTP exposes a network to significant security risks.</p> <p>The following commands were introduced or modified: appfw policy-name, application, audit-trail, content-length, content-type-verification, debug appfw, ip inspect name, max-header-length, max-uri-length, port-misuse, request-method, show appfw, strict-http, timeout, transfer-encoding type.</p>

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: 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. (1110R)

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

© 2012 Cisco Systems, Inc. All rights reserved.