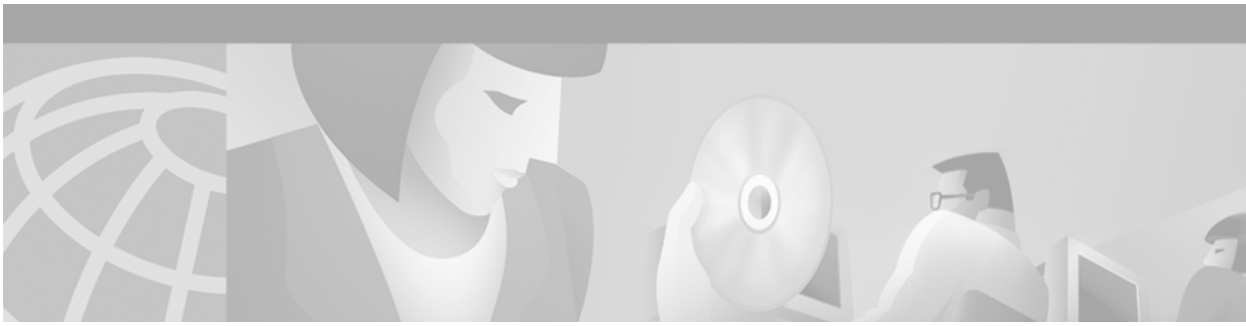


Cisco IOS WAP Gateway with WTLS Class 2 Support

Feature History

Release	Modification
12.2(2)XR	This feature was introduced.
12.2(4)XR	WTLS Class 2 Security features were added.

This document describes the Cisco IOS WAP Gateway with WTLS Class 2 Support feature in Cisco IOS Release 12.2(4)XR. It includes the following sections:

Feature Overview

The Cisco IOS WAP Gateway with WTLS Class 2 Support feature implements an optional cryptographic method for clients to authenticate the WAP gateway, and encryption between the wireless device and the gateway.

The Cisco IOS WAP Gateway is a software feature developed in compliance with Wireless Application Protocol (WAP) version 1.2. The software runs on the Cisco 3640 and 3660 routers. A Cisco router can be configured either as a dedicated gateway server, or as a multifunction box in conjunction with other Cisco IOS features.

The Cisco IOS WAP Gateway is an implementation of the gateway component of the WAP architecture. Companies can deploy the gateway to offer mobile employees and partners access to company-specific WAP content that resides on internal web servers. In the WAP architecture, WAP security exists from the

client device to the gateway but not through to the WAP content server. The Cisco IOS WAP Gateway can be implemented on a trusted server behind the company firewall, at which point it can access secure web content servers.

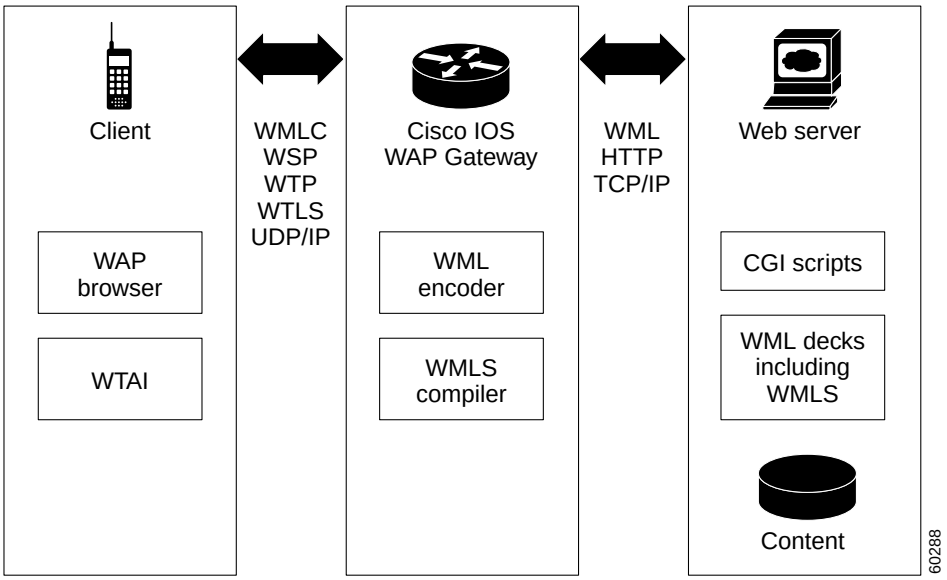
WAP is a standard for the presentation and delivery of wireless information and telephony services on wireless devices. Using an architecture based on the established World Wide Web model, WAP defines a set of protocols designed for use with mobile telephone technology and wireless devices.

WAP was developed and is promoted by the WAP Forum. Mobile wireless devices containing WAP browsers are available from most mobile device manufacturers.

WAP is designed to be independent of both the bearer technology and the device. WAP protocols can operate with many bearer technologies such as code division multiple access (CDMA), which is currently prevalent in the United States; global system for mobile communications (GSM), which is popular in Europe; and the GSM feature known as general packet radio service (GPRS), which is starting to be deployed and provides higher-speed data services. The major bearer technologies are all supported by WAP.

Figure 1 shows how the WAP gateway functions with the various protocols used to request and deliver data between a client wireless device, the Cisco IOS WAP Gateway, and a web content server. The client wireless device—a mobile phone in Figure 1—contains a WAP browser that can display Wireless Markup Language (WML) and execute Wireless Markup Language Scripts (WMLS). WML is derived from eXtensible Markup Language (XML) but is functionally equivalent to HTML, and it is designed to display WAP content on the small screens of mobile devices that can display only four or five lines of text plus some icons or basic graphics. Wireless mobile devices typically have memory limitations compared to the average PC.

Figure 1 Functionality of the Cisco IOS WAP Gateway



The mobile phone in Figure 1 also operates the Wireless Telephony Application Interface (WTAI), which gives access to the usual telephone capabilities such as phone books and dialing. WTAI allows telephone functionality to be controlled by WMLS. One use of this feature is to make the handset place a call to a telephone number that a WAP directory application has just retrieved and displayed on the screen.

When the client wireless device in Figure 1 initiates a request for WAP content, the request is forwarded to the WAP gateway via the wireless network provider that the client is using. Protocols used in the communication between the client and the gateway may include Wireless Session Protocol (WSP), Wireless Transaction Protocol (WTP), and Wireless Transport Layer Security (WTLS). All these protocols are optimized for use with wireless devices.

The WAP gateway in Figure 1 receives the request for WAP content from the wireless device and creates or reuses a session to the web server and requests content using HTTP. When the content is supplied to the gateway, it uses the WML encoder to compress the information and, if required, the gateway uses the WMLS compiler to compile the request before sending the WAP content back to the client device. Compression is achieved using a process called tokenization.

The web content server in Figure 1 can be an existing web content server using standard URLs and Common Gateway Interface (CGI) scripts, but the content destined for client wireless devices must be in WML format. The WAP browser in the wireless device cannot display content written in HTML. Communication between the gateway and the web server occurs through conventional protocols such as HTTP and the TCP/IP protocol stack.

Benefits

Leverage Existing Equipment and Expertise

You can integrate WAP services into your existing IP network infrastructure using existing equipment because the WAP gateway can run on the Cisco 3640 and 3660 routers and will even run as part of a multifunction router. The WAP gateway software uses new and modified commands at the Cisco IOS command-line interface (CLI), but the existing commands for configuring an interface or verifying the configuration should be familiar to Cisco customers.

Secure WAP Access to Internal Web Content

Companies can make intranet services available to employees and partners via the WAP gateway without compromising security. The Cisco IOS WAP Gateway with WTLS Class 2 Support feature can run on a trusted router within the firewall and access WAP content held on an internal server.

The WAP gateway implements WTLS Class 2 security, which provides an optional cryptographic method for clients to authenticate the WAP gateway, and encryption between the wireless device and the gateway.

Customers can implement their own user authentication methods by configuring the **wap authentication service** and associated optional commands.

Support for Microbrowsers

The Cisco IOS WAP Gateway implements the version 1.2 standards from the WAP Forum and will support all WAP microbrowsers that also implement these standards. Current and future WAP-enabled wireless devices will work with the gateway if they implement the WAP Forum version 1.2 standards.

Related Features and Technologies

- Cisco CTE 1400 Series Content Transformation Engine
- Cisco IOS Server Load Balancing (SLB)
- *V.110/WAP Access Solution*

Related Documents

- *Cisco IOS IP Command Reference, Volume 1 of 3: Addressing and Services*, Release 12.2
- *Cisco IOS IP Command Reference, Volume 2 of 3: Routing Protocols*, Release 12.2
- *Cisco IOS IP Command Reference, Volume 3 of 3: Multicast*, Release 12.2
- *Cisco IOS IP Configuration Guide*, Release 12.2

Supported Platforms

- Cisco 3640
- Cisco 3660

Determining Platform Support Through Feature Navigator

Cisco IOS software is packaged in feature sets that support specific platforms. To get updated information regarding platform support for this feature, access Feature Navigator. Feature Navigator dynamically updates the list of supported platforms as new platform support is added for the feature.

Feature Navigator is a web-based tool that enables you to quickly determine which Cisco IOS software images support a specific set of features and which features are supported in a specific Cisco IOS image.

To access Feature Navigator, you must have an account on Cisco.com. If you have forgotten or lost your account information, send a blank e-mail to cco-locksmith@cisco.com. An automatic check will verify that your e-mail address is registered with Cisco.com. If the check is successful, account details with a new random password will be e-mailed to you. Qualified users can establish an account on Cisco.com by following the directions at http://www.cisco.com/register.

Feature Navigator is updated regularly when major Cisco IOS software releases and technology releases occur. For the most current information, go to the Feature Navigator home page at the following URL:

http://www.cisco.com/go/fn

Supported Standards, MIBs, and RFCs

Standards

The Cisco IOS WAP Gateway with WTLS Class 2 Support feature conforms to all the mandatory standards requirements set out in the WAP 1.2 specifications created by the WAP Forum. All mandatory and certain optional features have been implemented. Your Cisco sales representative can provide a product bulletin containing the WAP Server Implementation Conformance Statement (WICS) for the Cisco IOS WAP Gateway with WTLS Class 2 Support feature.

MIBs

No new or modified MIBs are supported by this feature.

To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL:

http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

- RFC 1738, *Uniform Resource Locators (URL)*
- RFC 2068, *Hypertext Transfer Protocol—HTTP 1.1*

Prerequisites

WML Content Access

You must ensure that you have access to a web content server that contains WML and WMLS files that will be displayed on the WAP-enabled wireless devices. The software running the web server must be configured with WAP Multipurpose Internet Mail Extension (MIME) types to handle the various types of WAP files. We recommend that you configure the web server software to return an index.wml page when a URL is requested without specifying a page, because it can save the user some keystrokes on a WAP phone.

Client Software

The Cisco IOS WAP Gateway with WTLS Class 2 Support feature implements the version 1.2 standards from the WAP Forum and will support all WAP microbrowsers that also implement these standards. You must confirm that all your client wireless devices implement the 1.2 standards and are configured to access the WAP gateway.

WAP-enabled wireless devices have many different menu configurations but two parameters must be configured on each device that will access the Cisco IOS WAP Gateway. The IP address that must be entered in the WAP browser is the primary IP address of the interface on which the WAP gateway is configured. The User Datagram Protocol (UDP) port number being used by the wireless device must correspond to the protocol stack configured on the WAP gateway.

Configuration Tasks

See the following sections for configuration tasks for the Cisco IOS WAP Gateway with WTLS Class 2 Support feature. Each task in the list is identified as either required or optional:

- Configuring a WAP Gateway Interface (required)
- Configuring Customer-Supplied User Authentication on the WAP Gateway (optional)
- Configuring a Proxy List on the WAP Gateway (optional)
- Configuring Security Features on the WAP Gateway (optional)
- Setting the Maximum HTTP Header Length (optional)
- Configuring Support for UP Browsers (optional)
- Configuring WSP Sessions (optional)
- Configuring a WAP Gateway on a Multifunction Access Server (optional)
- Specifying How the WAP Gateway Locates Content Servers (optional)
- Verifying the Cisco IOS WAP Gateway (optional)

Configuring a WAP Gateway Interface

The Cisco IOS WAP Gateway is configured on only one interface but it operates over all physical interfaces to take advantage of any redundancy and to maximize availability. The interface on which the WAP gateway is configured can be a physical or loopback (virtual) interface. The gateway uses the primary IP address of this interface as the IP address for all WAP traffic, regardless of the actual physical interface over which the packets arrive or depart. To reduce the dependence on a physical interface that may be subject to physical connection issues or network failures, we recommend that the WAP gateway be configured on a loopback interface.

To enable the Cisco IOS WAP gateway on a router, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# interface type number`	Specifies the type and number of the interface on which the feature is to be configured. Enters interface configuration mode.
Step 2	`Router(config-if)# ip address ip-address mask`	Configures the interface with an IP address. This is the address with which the WAP-enabled wireless devices must be configured to communicate with the gateway.
Step 3	`Router(config-if)# wap {all | protocol-stack}`	Configures the interface to operate all the protocol stacks or a list specifying one or more of the options. The *protocol-stack* argument can be one or more of the following keywords: **cl**, **co**, **secure-cl**, and **secure-co**.

Configuring Customer-Supplied User Authentication on the WAP Gateway

The Cisco IOS WAP Gateway contains a feature and associated commands that allow the device browser to be redirected to a URL where customer-supplied user authentication can occur before the gateway will display any requested web content.

The user authentication feature can be used to supplement the static WAP username and password provided by most browsers. Static passwords may not provide the required level of security for an enterprise where all network access is controlled using one-time passwords. In this environment, the user must change the password on the WAP-enabled device before establishing each WAP session. The navigation on the device is tedious and may discourage use of the service. Using some form of initial group ID and password on the WAP-enabled device, and implementing a customer-supplied user authentication on the gateway, could allow the one-time password to be verified using WAP itself. A filtering mechanism may be employed on the firewall to ensure that the group ID initial requests access only the WAP gateway.

To configure this optional task, use the following commands in global configuration mode, as needed. The authentication is activated when a new session begins.

Command	Purpose
`Router(config)# wap authentication service url`	Configures customer-supplied user authentication by redirecting the browser to the specified URL. The URL normally points to a customer-supplied user authentication application where username and password information is entered and verified. **Note** The authentication scheme cannot be used when the gateway is operating the unsecured connectionless WSP protocol between the wireless device and the gateway because no session context is maintained between requests. If the user authentication feature is being used, the gateway should operate one or a combination of the co, secure-cl, and secure-co protocol stack options.
`Router(config)# wap authentication completed url`	Specifies a URL to be accessed by the browser to indicate to the gateway that the customer-supplied user authentication has validated the user. The gateway will detect that the URL was accessed and treat the session as authenticated.
`Router(config)# wap authentication prefix url`	Specifies a URL prefix to allow certain pages to be displayed that are related to the authentication process, for example, a screen with a corporate logo. The gateway processes requests on an unauthenticated session provided that the URL begins with this prefix. Other page requests will be redirected to the URL specified by the **wap authentication service** command.
`Router(config)# wap authentication timeout seconds`	Specifies an interval (in seconds) after which the user is forced to reauthenticate.

Configuring a Proxy List on the WAP Gateway

Using a proxy list allows the WAP gateway to determine which URL requests should be handled directly and which should be forwarded to a specified HTTP proxy server. The gateway searches through the proxy list in the order in which each filter request is entered when processing a request for a page. The gateway tests a page request against each line until a match is found. If no match is found the page request is serviced directly. An asterisk (*) wildcard can be used in the proxy filter entries.

Proxy servers are servers that will process information for another server and are sometimes used for security reasons to keep external requests from reaching internal servers. Proxy servers can also help ease performance issues because they take some of the load off other servers.

To create a proxy list, use the following command in global configuration mode. The proxy list records are searched in the order in which they are entered.

Command	Purpose
`Router(config)# wap proxy-list` *`http-server`*`[:`*`port-number`*`] [`*`proxy-server`*`[:`*`port-number`*`]]`	Specifies a filter record that the gateway can use to filter requests to be forwarded to a proxy server, and not passed directly to the server specified in the request. The *http-server* argument identifies a Domain Name System (DNS) name or IP address corresponding to an HTTP server. Asterisk (*) wildcard symbols can be used. The optional *proxy-server* argument identifies a DNS name or IP address corresponding to a proxy server. Both arguments may include an optional *port-number* argument separated by a colon. The default port number is 8080. Repeat the command, as needed, to create a list of filter records.

Configuring Security Features on the WAP Gateway

The Cisco IOS WAP Gateway with WTLS Class 2 Support feature implements an optional cryptographic method for clients to authenticate the WAP gateway, and encryption between the wireless device and the gateway. Several commands have been implemented in the software to allow the customer to configure the behavior of the gateway.

The gateway implements a number of different encryption, hash, and key-exchange algorithms. The use of each algorithm can be explicitly enabled or disabled. Deciding which algorithm to enable may depend on your company policy or the set of algorithms supported by the wireless devices with which the WAP gateway must communicate. Many wireless devices only support a subset of the available algorithms. While a WAP session is being established, the WAP-enabled device proposes the use of an algorithm and the gateway agrees to the proposal if it supports the proposed algorithm. That algorithm is then enabled. Unless you have a specific security requirement, the default configurations of both the wireless devices and the Cisco IOS WAP Gateway will usually work for all wireless devices.

For each type of algorithm you can select different strengths of security. A shorter key length is easier to compute and will impose less overhead on the processor than a longer key length, but a shorter key length offers weaker security. The level of security you need to configure will be determined by the type of information that can be accessed through the gateway. Confidential corporate information requires a higher level of security than information about the weather, for example, although having current access to such information may be invaluable.

Timeout intervals for idle WTLS sessions or connections can also be configured. A balance must be found between configuring a shorter interval in the interests of security and allowing a reasonable interval that stops the user from constantly needing to reauthenticate or reconnect when the interval expires.

To configure security options, use any or all of the following optional commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# wap wtls certificate gateway {rsa-512 | rsa-768 | rsa-unrestricted} filespec`	Registers a certificate for use with the specified key-exchange algorithm. The certificate must be obtained in advance from a certification authority (CA) in response to a certificate signing request (CSR), and copied to a file system accessible to the gateway. The CSR may be generated using the **wap wtls certificate generate csr** command. The *filespec* argument is the name, including path, of the certificate file obtained from a certification authority.
`Router(config)# wap wtls certificate generate csr {rsa-512 | rsa-768 | rsa-unrestricted} [common common-name] [country country-code] [state state] [locality city] [organization org-name] [orgunit org-unit-name] filespec filespec`	Generates a CSR suitable for sending to a certification authority.
`Router(config)# wap wtls encryption {all | cipher-block}`	Specifies the encryption algorithms operated by the WAP gateway. The *cipher-block* argument can be one or more of the following keywords: **rc5-cbc-40**, **rc5-cbc-56**, and **rc5-cbc-128**.
`Router(config)# wap wtls hash {all | algorithm}`	Specifies the hash algorithms operated by the WAP gateway. The *algorithm* argument can be one or more of the following keywords: **md5-40**, **md5-80**, **md5-128**, **sha-0**, **sha-40**, **sha-80**, **sha-160**, and **sha-xor-40**.
`Router(config)# wap wtls key-exchange {all | algorithm}`	Specifies the key-exchange algorithms operated by the WAP gateway. The *algorithm* argument can be one or more of the following keywords: **dh-anon-512**, **dh-anon-768**, **dh-anon-unrestricted**, **rsa-anon-512**, **rsa-anon-768**, **rsa-anon-unrestricted**, **rsa-512**, **rsa-768**, and **rsa-unrestricted**.
`Router(config)# wap wtls key-pair generate {rsa-512 | rsa-768 | rsa-unrestricted}`	Generates a Rivest, Shamir, and Adelman (RSA) key pair suitable for use with the specified key-exchange algorithm.
`Router(config)# wap wtls key-pair remove {rsa-512 | rsa-768 | rsa-unrestricted}`	Deletes an RSA key pair for the specified key-exchange algorithm. The key pair must have been previously created by the **wap wtls key-pair generate** command. **Note** Disable the use of the associated key-exchange algorithm before deleting the key pair.

Command	Purpose
`Router(config)# wap wtls timeout connection` *seconds*	Specifies an interval (in seconds) after which an inactive WTLS connection will be closed by the gateway.
`Router(config)# wap wtls timeout handshake` *seconds*	Specifies an interval (in seconds) that the gateway allows for the WTLS handshake process to complete.
`Router(config)# wap wtls timeout key` *seconds*	Specifies an interval (in seconds) during which the gateway will retain a WTLS session key when the session is unused.
`Router(config)# wap wtls timeout session` *seconds*	Specifies an interval (in seconds) after which an inactive WTLS session will be closed by the gateway.

Setting the Maximum HTTP Header Length

The WAP gateway builds HTTP headers using comma-separated values to request information from web content servers. When a web content server cannot handle a long line of text, use the **wap http maximum header-length** command to force line wrapping of HTTP headers.

To specify the maximum HTTP header length, use the following command beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# wap http maximum header-length` *max-characters*	Specifies how the gateway splits individual HTTP headers into multiple lines. The *max-characters* argument defines the maximum number of characters allowed before the gateway splits a line of HTTP headers at the end of the next comma-separated value. The default is set to 100 characters.

Configuring Support for UP Browsers

The Cisco IOS WAP Gateway with WTLS Class 2 Support feature contains commands to provide compatibility with Unwired Planet (UP) v4.0 browsers. When a UP browser on the wireless device starts, it requests a string of characters (known as “device:home”) instead of a URL. Subsequent page requests specify another string of characters (known as “device:base”). Using the **wap up-browser device-home** and the **wap up-browser device-base** commands, the WAP gateway translates the requests into previously configured URLs.

The **wap up-browser subno-suffix** command allows a gateway tag, identifying the WAP gateway, to be supplied to a header field request from a UP browser.

To support UP v4.0 browsers, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)# wap up-browser device-base url`	Specifies a URL prefix to be used as a substitute for the string of characters (known as "device:base") received in a request from a UP browser. The default *url* argument is a null string.
Step 2	`Router(config)# wap up-browser device-home url`	Specifies a URL to be used as a substitute for the string of characters (known as "device:home") received in a request from a UP browser. The default *url* argument is a null string.
Step 3	`Router(config)# wap up-browser subno-suffix gateway-tag`	Specifies a gateway tag to be appended to the header field to identify the WAP gateway before transmission to a WML content server. By default, no gateway tag is appended to the header field.

Configuring WSP Sessions

The Cisco IOS WAP Gateway contains commands to configure the maximum number of concurrent WSP sessions, and an interval after which an inactive WSP session will be timed out.

To configure WSP sessions, use any or all of the following optional commands in global configuration mode, as needed:

Command	Purpose
`Router(config)# wap wsp maximum sessions number`	Specifies the maximum number of concurrent WSP sessions. By default, there is no limit to the number of WSP sessions although practical limits exist, depending on the type of CPU in the router and the amount of memory.
`Router(config)# wap wsp timeout session seconds`	Specifies the number of seconds that a WSP session will continue to exist when there is no traffic for that session. By default, the number of seconds before a WSP session times out is set to 3600 (1 hour).

Configuring a WAP Gateway on a Multifunction Access Server

The WAP gateway can run either on a dedicated router or on a multifunction router. One example of a multifunction router would be to run the Cisco IOS WAP Gateway software on an access server. The ability to enable other Cisco IOS features will depend on the CPU and memory in the router. Feature enablement will affect the performance of the router. To run the WAP gateway on a multifunction access server, note the following configuration tips:

- Refer to the *Cisco IOS Dial Technologies Configuration Guide*, Release 12.2 for configuration scenarios.
- Configure the gateway on a loopback (virtual) interface for enhanced availability.

Specifying How the WAP Gateway Locates Content Servers

When a wireless device requests a web page via the WAP gateway, the Cisco IOS software must determine where to find the requested web page. A number of Cisco IOS commands can be configured to help reduce the time required to access the requested web page.

To specify multiple DNS servers and associated features, use any or all of the following commands in global configuration mode:

Command	Purpose
`Router(config)# ip domain-lookup`	Enables IP DNS host name to IP address translation to help access web content across the Internet. This command is enabled by default.
`Router(config)# ip name-server server-address1 [server-address2...server-address6]`	Specifies the IP addresses of up to six name servers to access DNS information. The first IP address specified becomes the first server that is accessed and the following IP addresses are checked in the order in which they are input.
`Router(config)# ip domain-name domain-name`	Defines a default domain name to complete an unqualified host name. Configuring a default domain name can save keystrokes on the wireless devices.
`Router(config)# ip domain-list domain-name`	Defines a list of default domain names to complete an unqualified host name. Each default domain name in the list is tried in turn until a match is found. Configuring a list of default domain names can save keystrokes on the wireless devices. **Note** If there is a domain list, the domain name defined in an **ip domain-name** command is not used.
`Router(config)# ip host name [tcp-port-number] address1 [address2...address8]`	Defines a static host name to IP address mapping that is saved in the host cache. Defining frequently accessed content servers avoids any dependency on DNS servers and can improve the software performance.

Verifying the Cisco IOS WAP Gateway

Verifying that the WAP gateway is working involves checking the configuration of the wireless devices and web servers. Those tasks may not be possible at the same location as the router that is acting as the WAP gateway. Some Cisco IOS commands, however, can be run on the router to determine if the correct WAP parameters are configured and running. Depending on the level of security required, one or more of the WAP protocol stacks are configured. Each WAP protocol stack is assigned a specific port number from 9200 through 9203. When the router is listening on a port number, use the **show ip sockets** EXEC command to display the port number information.

To verify that the Cisco IOS WAP Gateway with WTLS Class 2 Support feature is running, perform the following steps:

Step 1 Enter the **show wap** EXEC command to display the settings of all the WAP parameters. The values of all the WAP parameters, even those set to their default settings, are displayed.

```
Router# show wap

Cisco IOS Wireless Application Protocol Gateway parameters

WAP Gateway is enabled on interface Loopback0
WAP services available are: secure-cl secure-co

UP browser-specific settings:
        'device:home' is substituted with : 'http://www.company-name.com/wapserver/i'
        'device:base' is substituted with : 'http://www.company-name.com/wapserver'
        'x-up-subno' header is appended with 'gateway.company-name.com'

HTTP headers are wrapped after 80 bytes
User authentication service is 'http://www.company-name.com/auth/login.wml'
User authentication completed is 'http://www.company-name.com/auth/scripts/validate.cgi'
User authentication prefix is 'http://www.company-name.com/auth/scripts'
User authentication timeout is set to 20 minutes
WSP maximum sessions is set to 1000
WSP session timeout is set to 5 minutes
WTLS master key timeout is set to 1 day
WTLS session timeout is set to 1 hour 30 minutes
WTLS connection timeout is set to 1 hour 30 minutes
WTLS handshake timeout is set to 5 minutes

WTLS Encryption Algorithms:
        RC5-CBC-128 - enabled
        RC5-CBC-56  - enabled
        RC5-CBC-40  - enabled

WTLS Hash Algorithms:
        MD5-128    - disabled
        MD5-80     - disabled
        MD5-40     - disabled
        SHA-160    - enabled
        SHA-80     - enabled
        SHA-40     - enabled
        SHA-XOR-40 - disabled
        SHA-0      - disabled

WTLS Key Exchange Algorithms:
        DH-ANON-UNRESTRICTED  - disabled
        DH-ANON-768           - disabled
        DH-ANON-512           - disabled
        RSA-ANON-UNRESTRICTED - disabled
        RSA-ANON-768          - disabled
```

```
        RSA-ANON-512          - disabled
        RSA-UNRESTRICTED      - disabled Certificate: nvram:rsa-un-ca.cer
        RSA-768               - disabled Certificate: nvram:rsa-768-ca.cer
        RSA-512               - enabled  Certificate: nvram:rsa-512-ca.cer

Proxy list is:
        *.company-name.com
        *.company-name.com:*
        *.*->proxy.company-name.com
```

Step 2 Enter the **show ip sockets** EXEC command to display the ports that are being used. Ports 9202 and 9203 are in use, confirming the configuration of the **wap** command.

```
Router# show ip sockets

Proto Remote           Port    Local         Port  In Out Stat TTY OutputIF
 17   0.0.0.0             0 172.20.1.1        67   0   0  489   0
 17   10.1.0.2        49998 172.20.1.1      9203   0   0   B1   0
 17   0.0.0.0             0 172.20.1.1      9202   0   0   B1   0
```

Step 3 Enter the **show wap statistics** EXEC command to show that traffic is being generated. A wireless phone or phone-emulator software on a PC, configured to access the WAP gateway, will generate traffic. Run this command several times while generating the traffic to ensure that the counters are being updated.

```
Router# show wap statistics

errors requests responses sessions sessions-HWM rx-udp tx-udp
0      2614     2614      0        2            5151   2894
 timers: 0
 number of memory pools: 17
```

Troubleshooting Tips

WAP Gateway

The Cisco IOS WAP gateway introduces a new EXEC mode command, **debug wap**, to enable diagnostic output concerning various events relating to the operation of the WAP gateway to be displayed on a console. The **debug wap** command is intended only for troubleshooting purposes because the volume of output generated by the software can result in severe performance degradation on the router. To minimize the impact of using the **debug wap** commands, perform the following steps:

Step 1 Attach a console directly to the router running the WAP gateway.

Step 2 Enter the **no logging console** command in global configuration mode to disable all logging to the console terminal. To reenable logging to the console, use the **logging console** command in global configuration mode.

Step 3 Use Telnet to access a router port. Enter the **enable** command in EXEC configuration mode.

Step 4 Enter the **terminal monitor** command in global configuration mode and enter the necessary **debug wap** commands. Try to enter only specific **debug wap** commands to isolate the output to a certain subcomponent and minimize the load on the processor. Use the **detailed** keyword to generate more detailed debug information on specified subcomponents. To disable logging on the virtual terminal, enter the **no terminal monitor** command.

Step 5 Enter the specific **no debug wap** command when you are finished.

This procedure will minimize the load on the router created by the **debug wap** commands because the console port is no longer generating character-by-character processor interrupts. If you cannot connect to a console directly, you can run this procedure via a terminal server. If you must break the Telnet connection, however, you may not be able to reconnect because the router may be unable to reconnect due to the processor load of generating the **debug wap** output.

WAP Wireless Devices

Ensure that your WAP-enabled wireless device is configured with the appropriate WAP parameters. The IP address that is entered in the WAP browser in the wireless device is the primary IP address of the interface on which the WAP gateway is configured. The UDP port number being used by the wireless device must correspond to the protocol stack configured on the WAP gateway.

Web Content Servers

Ensure that the content server contains the relevant WML files and scripts and can be accessed by the router running the Cisco IOS WAP Gateway with WTLS Class 2 Support feature. The content server software must be configured to register the various WAP MIME types.

Monitoring and Maintaining the Cisco IOS WAP Gateway

To monitor and maintain the Cisco IOS WAP Gateway with WTLS Class 2 Support feature, use the following commands in EXEC mode, as needed:

Command	Purpose
`Router# clear wap statistics`	Resets the WAP gateway counters.
`Router# show wap`	Displays the values of all the WAP gateway parameters. All parameters, even those set to their defaults, are displayed.
`Router# show wap statistics`	Displays the counters maintained by the WAP gateway.

Configuration Examples

This section provides the following configuration examples:

- WAP Gateway Interface Configuration Example
- Customer-Supplied User Authentication Configuration Example
- Proxy List Configuration Example
- WAP Security Features Configuration Examples
- Maximum HTTP Header Length Configuration Example
- UP Browser Support Configuration Example
- WSP Session Configuration Example

- Content Server Location Configuration Example

WAP Gateway Interface Configuration Example

In the following example, the WAP gateway is enabled on a loopback (virtual) interface, the secure connectionless protocol stack is configured, and the secure connection-oriented protocol stack is configured:

```
interface Loopback0
 ip address 172.20.1.1 255.255.0.0
 wap secure-cl secure-co
```

Customer-Supplied User Authentication Configuration Example

In the following example, customer-supplied user authentication is enabled, a file called login.wml prompts the user for a username and password, and a Common Gateway Interface (CGI) script called validate.cgi validates the username and password and displays a page with an HTTP status indicating success or failure:

```
wap authentication service http://www.company-name.com/auth/login.wml
wap authentication completed http://www.company-name.com/auth/scripts/validate.cgi
wap authentication prefix http://www.company-name.com/auth/scripts/
```

The first command causes the gateway to run the customer-supplied authentication procedure in login.wml whenever a new session is started. Instead of serving the first page request on the session, the browser will be redirected to the file called login.wml. This file contains code written in WML, which prompts the user for a username and a password. The username and password parameters that are entered by the user are sent to the CGI script called validate.cgi.

The second command registers that validate.cgi is the file that must be successfully retrieved to indicate to the gateway that the authentication process succeeded. This CGI script validates the username and password using a user-specified mechanism. The CGI script returns WML, which displays the first customer-supplied authentication screen, along with a message to indicate whether authentication succeeded.

The third command permits pages in the specified directory to be accessed during the authentication process. The specified prefix can be used, for example, to allow the authentication procedure to display a corporate logo on the screen or to operate a more complex authentication procedure using multiple files.

Proxy List Configuration Example

In the following example, a sequence of proxy list records is created. The first record captures all the local domain requests and services them directly. The second record captures local domain requests with a specified port number and services them directly. The third record specifies both an HTTP server and a name of a proxy server to which the requests matching the HTTP server are forwarded. Note that the first two filter records will capture all the local page requests except those requests that do not specify a company-name host. The third filter record captures all other requests containing a dot in the domain name and forwards the requests to the specified proxy server. If the gateway does not find a match in the list, the page request is serviced directly:

```
wap proxy-list *.company-name.com
wap proxy-list *.company-name.com:*
wap proxy-list *.* proxy.company-name.com
```

WAP Security Features Configuration Examples

In the following example, the gateway is being configured to operate all the encryption algorithms, but only a selection of the key-exchange and hash algorithms. WTLS timeout parameters are all set to an interval in seconds that is different from their default values.

```
wap wtls encryption all
wap wtls hash sha-40 sha-80 sha-160
wap wtls key-exchange dh-anon-512 dh-anon-768 dh-anon-unrestricted
wap wtls timeout key 86400
wap wtls timeout session 5400
wap wtls timeout connection 5400
wap wtls timeout handshake 300
```

In the following example, the gateway is being configured to operate WTLS Class 2 security. A key pair and a certificate signing request are generated for the RSA 512 algorithm. The certificate, obtained from a certification authority, is copied to NVRAM and registered with the WAP gateway. Finally, the RSA 512 key-exchange algorithm is enabled.

```
wap wtls key-pair generate rsa-512
wap wtls certificate generate csr rsa-512 common wapserver.company-name.com filespec
cert512.csr
! the certification authority will return a certificate, for example, cert512.cer
! the certificate is then copied to a tftp server before the next command is issued
copy tftp:cert512.cer nvram:cert512.cer
wap wtls certificate gateway rsa-512 nvram:cert512.cer
wap wtls key-exchange rsa-512
```

Maximum HTTP Header Length Configuration Example

In the following example, the WAP gateway is being configured to set the maximum HTTP header length to 80 characters.

```
wap http maximum header-length 80
```

UP Browser Support Configuration Example

In the following example, the WAP gateway is being configured with a substitute URL prefix and a substitute URL to replace the “device-base” and “device-home” requests from a UP browser. A gateway tag, identifying the WAP gateway, is also configured for the gateway to append to the header field of a request from UP browser.

```
wap up-browser device-base http://www.company-name.com/wapserver
wap up-browser device-home http://www.company-name.com/wapserver/index.wml
wap up-browser subno-suffix gateway.company-name.com
```

WSP Session Configuration Example

In the following example, the gateway is being configured to set the maximum number of WSP sessions to 1000, and the number of seconds before an idle WSP session times out is being set to 300 seconds (5 minutes).

```
wap wsp maximum sessions 1000
wap wsp timeout session 300
```

Content Server Location Configuration Example

In the following example, a default domain name, some host names, and two DNS servers are configured:

```
ip domain-name company-name.com
ip host web.company-name.com 10.1.1.10
ip host wapauth.company-name.com 172.21.1.10
ip name-server 172.21.1.100
ip name-server 10.3.1.100
```

Command Reference

This section documents new commands. All other commands used with this feature are documented in the Cisco IOS Release 12.2 command reference publications.

- **clear wap statistics**
- **debug wap**
- **show wap**
- **wap**
- **wap authentication completed**
- **wap authentication prefix**
- **wap authentication service**
- **wap authentication timeout**
- **wap http maximum header-length**
- **wap proxy-list**
- **wap up-browser device-base**
- **wap up-browser device-home**
- **wap up-browser subno-suffix**
- **wap wsp maximum sessions**
- **wap wsp timeout session**
- **wap wtls certificate gateway**
- **wap wtls certificate generate csr**
- **wap wtls encryption**
- **wap wtls hash**
- **wap wtls key-exchange**
- **wap wtls key-pair generate**
- **wap wtls key-pair remove**
- **wap wtls timeout connection**
- **wap wtls timeout handshake**
- **wap wtls timeout key**
- **wap wtls timeout session**

clear wap statistics

To reset the Wireless Application Protocol (WAP) gateway counters, use the **clear wap statistics** command in EXEC mode.

clear wap statistics

Syntax Description

This command has no arguments or keywords.

Command Modes

EXEC

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

WAP gateway counters accumulate during operation and can be useful for monitoring. To establish baseline statistics, use the **clear wap statistics** command at the start of the monitoring period to ensure that the counters are reset.

Examples

The following example shows all the WAP gateway counters being reset:

```
Router# clear wap statistics
```

debug wap

To display debug messages for the Wireless Application Protocol (WAP) gateway, use the **debug wap** command in EXEC mode. To turn off debugging, use the **no** form of this command.

debug wap {**all** | **dns** | **gateway** | **html-to-wml** | **http** | **udp** | **wml** | **wsp** | **wtls** | **wtp**} [**detailed**]

no debug wap {**all** | **dns** | **gateway** | **html-to-wml** | **http** | **udp** | **wml** | **wsp** | **wtls** | **wtp**} [**detailed**]

Syntax Description

all	All active WAP protocols and markup language activity.
dns	Domain Name System (DNS) inquiries.
gateway	Gateway subsystem activity.
html-to-wml	HTML to Wireless Markup Language (WML) activity. **Note** There is no general HTML to WML information conversion. The gateway supports only limited HTML to WML conversion in a situation where web servers have been configured to return a page of HTML with an HTTP error status. If HTML to WML information conversion is required, we recommend running the WAP gateway with a Content Transformation Engine (CTE) such as the Cisco CTE 1400.
http	HTTP activity.
udp	Activity related to the use of User Datagram Protocol (UDP) by the gateway.
wml	WML and Wireless Markup Language Script (WMLS) activity.
wsp	Wireless Session Protocol (WSP) activity.
wtls	Wireless Transport Layer Security (WTLS) activity.
wtp	Wireless Transaction Protocol (WTP) activity.
detailed	(Optional) Detailed debug information for specified subcomponents.

Command Modes

EXEC

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

To minimize the load on the processor, use specific **debug wap** commands. Use the **show debugging** EXEC command to keep track of which debug commands are currently operational. Use the **undebug** EXEC command to turn off all debugging options.

Caution The **debug wap** command is intended for troubleshooting and should not be used during normal operation. The output generated may create severe performance degradation.

Examples

The following is sample output from the following **debug wap** command:

```
Router# debug wap all

Jun 26 09:49:56.683: WAP wtls: rx checksum 68 41 3D CA
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 3, 3
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=3, mac=3
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 3, 2
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=3, mac=2
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 3, 1
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=3, mac=1
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 2, 3
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=2, mac=3
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 2, 2
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=2, mac=2
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 1, 3
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=1, mac=3
Jun 26 09:49:56.683: WAP wtls: Proposed cipher suite 1, 2
Jun 26 09:49:56.683: WAP wtls:  requested cipher suite. bulk=1, mac=2
Jun 26 09:49:56.683: WAP wtls: Proposed Key Exchange 8 (0)
Jun 26 09:49:56.683: WAP wtls: Unsupported Key Exchange algorithm 8 was proposed
Jun 26 09:49:56.683: WAP wtls: Proposed Key Exchange 10 (0)
Jun 26 09:49:56.683: WAP wtls: Unsupported Key Exchange algorithm 10 was proposed
Jun 26 09:49:56.683: WAP wtls: Proposed Key Exchange 9 (0)
Jun 26 09:49:56.683: WAP wtls: Unsupported Key Exchange algorithm 9 was proposed
Jun 26 09:49:56.683: WAP wtls: Proposed Key Exchange 5 (0)
Jun 26 09:49:56.683: WAP wtls: Proposed Key Exchange 7 (0)
Jun 26 09:49:56.683: WAP wtls: Proposed Key Exchange 6 (0)
Jun 26 09:49:56.683: WAP wtls: Selected algorithms bulk cipher algorithm 3
Selected mac algorithm 3
Selected compression mode 0
Selected key exchange 5(0)

Jun 26 09:50:03.431: WAP wtls: rx checksum 96 2C 37 2C
Jun 26 09:50:03.631: WAP wtls: Computing master secret
Jun 26 09:50:03.631: WAP wtls: rx checksum C0 01 02 00
Jun 26 09:50:03.631: WAP wtls: rx checksum 78 CA EA 87
Jun 26 09:50:03.631: WAP wtls: Verify finish succeeded
Jun 26 09:50:03.635: WAP wtls: Going to Opening state...
Jun 26 09:50:04.679: WAP wtls: rx checksum 89 90 CF 20
Jun 26 09:50:04.679: WAP wtp: tr_invoke_res called
Jun 26 09:50:05.679: WAP wtls: rx checksum C7 DF EC A5
Jun 26 09:50:05.803: WAP wtls: rx checksum 4D 02 07 3B
Jun 26 09:50:05.807: WAP wtp: tr_invoke_res called
Jun 26 09:50:05.827: WAP wml: squished 610 into 540
```

The following is sample output from the **debug wap all detailed** command:

```
Router# debug wap all detailed

Jun 26 09:56:43.415: WAP udp: < udp:9203 75 1.1.0.2:49998
 C3 00 00 00 46 01 00 43 01 3B 38 6A 58 4E 27 48
 C6 B0 BF 29 18 EF 41 3E 4D 08 A3 92 DD 1D 92 93
 30 4C 00 12 08 00 00 0A 00 00 09 00 00 05 00 00
 07 00 00 06 00 00 00 00 0E 03 03 03 02 03 01 02
 03 02 02 01 03 01 02 01 00 02 03

Jun 26 09:56:43.415: WAP wtls: rx checksum A1 76 D8 68
Jun 26 09:56:43.415: WAP wtls: Client Hello received 0
Jun 26 09:56:43.415: WAP wtls: SEC_Create.ind received
Jun 26 09:56:43.415: WAP wtls: Proposed cipher suite 3, 3
Jun 26 09:56:43.415: WAP wtls:  requested cipher suite. bulk=3, mac=3
Jun 26 09:56:43.415: WAP wtls: Proposed cipher suite 3, 2
Jun 26 09:56:43.415: WAP wtls:  requested cipher suite. bulk=3, mac=2
```

```
Jun 26 09:56:43.415: WAP wtls: Proposed cipher suite 3, 1
Jun 26 09:56:43.415: WAP wtls:  requested cipher suite. bulk=3, mac=1
Jun 26 09:56:43.415: WAP wtls: Proposed cipher suite 2, 3
Jun 26 09:56:43.415: WAP wtls:  requested cipher suite. bulk=2, mac=3
Jun 26 09:56:43.415: WAP wtls: Proposed cipher suite 2, 2
Jun 26 09:56:43.415: WAP wtls:  requested cipher suite. bulk=2, mac=2
Jun 26 09:56:43.415: WAP wtls: Proposed cipher suite 1, 3
Jun 26 09:56:43.415: WAP wtls: Proposed Key Exchange 9 (0)
Jun 26 09:56:43.415: WAP wtls: Unsupported Key Exchange algorithm 9 was proposed
Jun 26 09:56:43.415: WAP wtls: Proposed Key Exchange 5 (0)
Jun 26 09:56:43.415: WAP wtls: Proposed Key Exchange 7 (0)
Jun 26 09:56:43.415: WAP wtls: Proposed Key Exchange 6 (0)
Jun 26 09:56:43.415: WAP wtls: Selected algorithms bulk cipher algorithm 3
Selected mac algorithm 3
Selected compression mode 0
Selected key exchange 5(0)

Jun 26 09:56:49.611: WAP wtls: Computing master secret
Jun 26 09:56:49.611: WAP wtls: master secret D7 63 E2 E4 17 A4 30 2C 6E 99 27 42
DF 70 74 D5 08 74 61 9C

Jun 26 09:56:49.611: WAP wtls: New receiving hmac key 8B AD 55 F2 1A 19 1A FD DD
F9 E7 17 51 AF 9D 92 B7 50 FD 8A

Jun 26 09:56:49.615: WAP wtls: New receiving key B8 98 91 B9 28 6A F7 D1 18 11
15 E4 F5 C1 C9 BA

Jun 26 09:56:49.615: WAP wtls: New receiving iv EB 99 F6 0A AF ED BA 1C

Jun 26 09:56:49.615: WAP wtls: decrypting with iv EB 99 F6 0A AF ED BA 1C

Jun 26 09:56:49.615: WAP wtls: decrypting with secret B8 98 91 B9 28 6A F7 D1 18
11 15 E4 F5 C1 C9 BA

Jun 26 09:56:49.615: WAP wtls: Finished received
Jun 26 09:56:49.615: WAP wtls: Creating finished on 378 bytes of handshake
Jun 26 09:56:49.615: WAP wtls: Handshake

Jun 26 09:56:51.663: WAP wtp: Rcv Invoke 32770
Jun 26 09:56:51.663: WAP wtp: State
Jun 26 09:56:51.663: WAP wsp: Supplier invoke 2
Jun 26 09:56:51.663: WAP dns: Looking up 172.20.1.1
Jun 26 09:56:51.663: WAP dns: Translated 172.20.1.1 to 172.20.1.1:0
Jun 26 09:56:51.663: WAP http: processing send in state 0
Jun 26 09:56:51.663: WAP http: connecting to 172.20.1.1:80
Jun 26 09:56:51.663: WAP http: waiting to connect...
Jun 26 09:56:51.667: WAP http: HTTP Request:
GET /a.wml HTTP/1.1
user-agent: Nokia7110/1.0 (04.84)
accept-charset: iso-8859-1,utf-8,iso-10646-ucs-2
accept-language: en
accept:
text/vnd.wap.wml,application/vnd.wap.wmlc,text/vnd.wap.wmlscript,application/vnd
.wap.wmlscriptc
accept: image/vnd.wap.wbmp,application/vnd.wap.wtls-ca-
certificate,text/plain,text/html; q=0.1
if-modified-since: Jul 08 2006 09:34:02
X-Network-Info: 10.1.0.2:49998
host: 172.20.1.1
"http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>

<card id="card1" title="Currency" newcontext="true">
```

```
<p>
Amount: <input format="*N" name="amount" title="Amount:"/>

From: <select name="from" value="USD" title="From:">
<option value="DEM">German Mark</option>
<option value="FRF">French Franc</option>
.
.
.

Jun 26 09:56:51.675: WAP http:
.
.
.

Jun 26 09:56:51.683: WAP wml: substituting for "amount"
Jun 26 09:56:51.683: WAP wml: substituting for "German Mark"
Jun 26 09:56:51.683: WAP wml: substituting for "French Franc"
Jun 26 09:56:51.683: WAP wml: substituting for "Finnish Markka"
Jun 26 09:56:51.683: WAP wml: substituting for "US Dollar"
Jun 26 09:56:51.687: WAP wml: substituting for "currency.wmls#"
Jun 26 09:56:51.687: WAP wml: substituting for "conversion"
Jun 26 09:56:51.687: WAP wml: substituting for "card1_help"
Jun 26 09:56:51.687: WAP wml: substituting for "currency.wmls#"
Jun 26 09:56:51.687: WAP wml: squished 610 into 540
```

Related Commands

Command	Description
show debugging	Displays the state of each debugging option.
undebug	Turns off debugging displays.

show wap

To display the values of all the Wireless Application Protocol (WAP) gateway parameters, use the **show wap** command in EXEC mode.

show wap [**statistics**]

Syntax Description

statistics	(Optional) Counters maintained by the WAP gateway.

Defaults

By default, all WAP gateway parameter values are displayed.

Command Modes

EXEC

Command History

Release	Modification
12.2(2)XR	This command was introduced.
12.2(4)XR	The output was extended to display the status of the new rsa-512, rsa-768, and rsa-unrestricted key-exchange algorithms.

Usage Guidelines

If no arguments or keywords are specified, the gateway displays the values of all the WAP parameters, even those that are set to their default. Use the **show running-config** EXEC command to display WAP gateway parameters that have been changed from their default values in the current configuration running on the router.

Examples

The following is sample output from the **show wap** command:

```
Router# show wap

Cisco IOS Wireless Application Protocol Gateway parameters

WAP Gateway is enabled on interface Loopback0
WAP services available are: secure-cl secure-co

UP browser-specific settings:
        'device:home' is substituted with :
'http://www.company-name.com/wapserver/index.wml'
        'device:base' is substituted with : 'http://www.company-name.com/wapserver'
        'x-up-subno' header is appended with 'gateway.company-name.com'

HTTP headers are wrapped after 80 bytes
User authentication service is 'http://www.company-name.com/wapserver/authen_service.wml'
User authentication completed is
'http://www.company-name.com/wapserver/authen_completed.wml'
User authentication prefix is 'http://www.company-name.com/wapserver/pages'
User authentication timeout is set to 20 minutes
WSP maximum sessions is set to 1000
WSP session timeout is set to 5 minutes
WTLS master key timeout is set to 1 day
```

```
WTLS session timeout is set to 1 hour 30 minutes
WTLS connection timeout is set to 1 hour 30 minutes
WTLS handshake timeout is set to 3 minutes

WTLS Encryption Algorithms:
        RC5-CBC-128 - enabled
        RC5-CBC-56  - enabled
        RC5-CBC-40  - enabled

WTLS Hash Algorithms:
        MD5-128    - disabled
        MD5-80     - disabled
        MD5-40     - disabled
        SHA-160    - enabled
        SHA-80     - enabled
        SHA-40     - enabled
        SHA-XOR-40 - disabled
        SHA-0      - disabled

WTLS Key Exchange Algorithms:
        DH-ANON-UNRESTRICTED  - disabled
        DH-ANON-768           - disabled
        DH-ANON-512           - disabled
        RSA-ANON-UNRESTRICTED - disabled
        RSA-ANON-768          - disabled
        RSA-ANON-512          - disabled
        RSA-UNRESTRICTED      - disabled Certificate: nvram:rsa-un-ca.cer
        RSA-768               - disabled Certificate: nvram:rsa-768-ca.cer
        RSA-512               - enabled  Certificate: nvram:rsa-512-ca.cer
Proxy list is:
        *.company-name.com
        *.company-name.com:*
        *.*->proxy.company-name.com
```

Table 1 describes the significant fields shown in the display.

Table 1 show wap Field Descriptions

Field	Description
WAP Gateway is enabled on interface	Interface on which the WAP gateway is enabled.
WAP services available are:	Indicates which protocol stack options the gateway is running.
UP browser-specific settings:	Indicates whether the parameters for each of the three WAP Unwired Planet (UP) browser commands have been set.
HTTP headers are wrapped	Maximum number of characters allowed before the gateway splits up a line of HTTP headers at the end of the next comma-separated value.
User authentication service	URL to start a customer-supplied user authentication service.
User authentication completed	URL to be successfully displayed when the authentication process is completed.
User authentication prefix	URL prefix used to permit access to pages during the WAP customer-supplied user authentication process.
User authentication timeout	Interval after which a user is forced to reauthenticate.
WSP maximum sessions	Maximum number of concurrent Wireless Session Protocol (WSP) sessions.

Table 1 show wap Field Descriptions (continued)

Field	Description
WSP session timeout	Interval after which a WSP session without traffic expires.
WTLS master key timeout	Interval after which an unused Wireless Transport Layer Security (WTLS) master key expires.
WTLS session timeout	Interval after which an inactive WTLS session expires.
WTLS connection timeout	Interval after which an inactive WTLS connection expires.
WTLS handshake timeout	Interval allowed for the WTLS handshake process to be completed.
WTLS Encryption Algorithms:	Indicates whether each encryption algorithm is enabled or disabled.
WTLS Hash Algorithms:	Indicates whether each hash algorithm is enabled or disabled.
WTLS Key Exchange Algorithms:	Indicates whether each key-exchange algorithm is enabled or disabled. For key-exchange algorithms that require a certificate from a certification authority, the output may display more information. If no keys have been generated the output will display the string "Keys: No". If a key has been generated but a certificate has not been registered it will display the string "Keys: Yes", and if a certificate has been registered it will display the string "Certificate:" followed by the path and name of the certificate file.
Proxy list is:	Indicates whether a proxy list is defined.

The following is sample output from the **show wap statistics** command:

```
Router# show wap statistics

errors requests responses sessions sessions-HWM rx-udp tx-udp
0      2614     2614      0        2            5151   2894
 timers: 0
 number of memory pools: 17
```

Table 2 describes the significant fields shown in the display.

Table 2 show wap statistics Field Descriptions

Field	Description
errors	Number of errors.
requests	Number of requests sent by the gateway to the content server.
responses	Number of responses received by the gateway from the content server.
sessions	Number of sessions currently in use by the gateway.
session-HWM	Maximum number of concurrent WSP sessions since last reset of sessions counter—high-water mark (HWM).
rx-udp	Number of messages received by the gateway using User Datagram Protocol (UDP).
tx-udp	Number of messages returned by the gateway using UDP.

Table 2 show wap statistics Field Descriptions (continued)

Field	Description
timers:	Number of timers currently in use by the gateway.
number of memory pools:	Number of memory pools currently in use by the gateway.

Related Commands

Command	Description
clear wap statistics	Resets the WAP gateway counters.
show running-config	Displays the current configuration.

wap

To enable the Wireless Application Protocol (WAP) gateway on an interface, use the **wap** command in interface configuration mode. To disable the WAP gateway, use the **no** form of this command.

wap {**all** | *protocol-stack*}

no wap {**all** | *protocol-stack*}

Syntax Description

all	All the supported WAP protocol stacks.
protocol-stack	One or more of the following protocol stacks: • **cl**—Connectionless Wireless Session Protocol (WSP) using User Datagram Protocol (UDP) port 9200. • **co**—Connection-oriented WSP with Wireless Transaction Protocol (WTP) using UDP port 9201. • **secure-cl**—Secure connectionless WSP with Wireless Transport Layer Security (WTLS) using UDP port 9202. • **secure-co**—Secure connection-oriented WSP with WTP and WTLS using UDP port 9203.

Defaults

By default, the WAP gateway is not enabled.

Command Modes

Interface configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The WAP gateway is configured only on one interface, but it operates over all physical interfaces to take advantage of any redundancy and to maximize availability. The interface on which the WAP gateway is configured can be a physical or loopback (virtual) interface. The gateway uses the primary IP address of this interface as the IP address for all WAP traffic regardless of the actual physical interface over which the packets arrive or depart. Use of a single IP address ensures that the WAP gateway will communicate correctly with WAP browsers, even when return traffic is routed over a different path and leaves the router via a different physical interface.

To reduce the dependence on a physical interface, which may be subject to physical connection issues or network failures, we recommend that the WAP gateway be configured on a loopback interface created for this purpose.

When the WAP gateway is disabled, it stops operating over all the interfaces on the router.

The WAP gateway can be configured to operate all the WAP protocol stacks or a list specifying one or more of the options. Determining which of the possible protocol stacks the WAP gateway will operate depends on the type of service or information that is being accessed via the WAP gateway. Information

that is not confidential need not operate with the overhead of the secure protocol stacks. Confidential corporate information and credit card services require the secure protocol stacks. Each protocol stack operates over a different port.

Use the **show wap** EXEC command to display the interface on which WAP is enabled and to display which WAP services are available.

Note WAP-enabled wireless devices must be configured in advance to use a protocol stack that is being operated by the WAP gateway because there is no method to negotiate between protocol stacks when a session is established. Not all WAP-enabled wireless devices support all the possible WAP protocol stacks.

Examples

The following example shows the WAP gateway being enabled on a loopback interface and configured to operate both types of secure WAP protocol stacks:

```
interface Loopback 0
 ip address 172.20.1.1 255.255.0.0
 wap secure-cl secure-co
```

The following example shows the WAP gateway being disabled:

```
interface Loopback 0
 no wap all
```

Related Commands

Command	Description
show wap	Displays the values of all WAP gateway parameters.

wap authentication completed

To specify the URL that must be successfully accessed to indicate to the Wireless Application Protocol (WAP) gateway that authentication succeeded, use the **wap authentication completed** command in global configuration mode. To restore this parameter to its default value, use the **no** form of this command.

wap authentication completed *url*

no wap authentication completed

Syntax Description

url	URL used to indicate that authentication succeeded.

Defaults

By default, the URL specified by the **wap authentication service** command is used.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The **wap authentication completed** command is used with other authentication commands to enable a customer-supplied user authentication scheme to work with the WAP gateway.

If the **wap authentication service** command is configured, the WAP gateway will redirect the browser to the page configured by the **wap authentication service** command whenever a new session starts. The customer-supplied authentication process will then proceed and a page, specified by the **wap authentication completed** command, will be displayed if the authentication process is successful. Until the authentication process is successful, the WAP gateway will service only URLs specified by the **wap authentication service**, **wap authentication completed**, and **wap authentication prefix** commands, and all other page requests within that session will be denied.

Note The authentication scheme cannot be run when the WAP gateway is operating the connectionless WSP protocol to communicate between the wireless device and the WAP gateway.

Examples

The following example shows the URL http://www.company-name.com/auth/scripts/validate.cgi being configured as the URL to be accessed to indicate to the WAP gateway that authentication succeeded:

```
wap authentication completed http://www.company-name.com/auth/scripts/validate.cgi
```

Related Commands

Command	Description
wap authentication prefix	Specifies a URL prefix used to permit access to pages during the customer-supplied user authentication process.
wap authentication service	Initiates customer-supplied user authentication.
wap authentication timeout	Specifies an interval after which the user is forced to reauthenticate.

wap authentication prefix

To specify a URL prefix that permits access to pages during the Wireless Application Protocol (WAP) customer-supplied user authentication process, use the **wap authentication prefix** command in global configuration mode. To disable use of the prefix, use the **no** form of this command.

wap authentication prefix *url*

no wap authentication prefix

Syntax Description

url	URL prefix used to permit access to pages during the WAP customer-supplied user authentication process. The URL prefix will usually be a directory but can be any valid URL.

Defaults

By default no prefix is defined.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The **wap authentication prefix** command is used with other authentication commands to enable a customer-supplied user authentication scheme to work with the WAP gateway. The prefix URL is normally used to point to a directory from which other pages can be displayed during the authentication process.

If the **wap authentication service** command is configured, the WAP gateway will redirect the browser to the page configured by the **wap authentication service** command whenever a new session starts. The customer-supplied authentication process will then proceed and a page, specified by the **wap authentication completed** command, will be displayed if the authentication process is successful. Until the authentication process is successful, the WAP gateway will service only URLs specified by the **wap authentication service**, **wap authentication completed**, and **wap authentication prefix** commands, and all other page requests within that session will be denied.

Note The authentication scheme cannot be run when the WAP gateway is operating the connectionless WSP protocol to communicate between the wireless device and the WAP gateway.

Examples

The following example shows the directory named scripts being configured as the authentication prefix URL:

```
wap authentication prefix http://www.company-name.com/auth/scripts/
```

Related Commands

Command	Description
wap authentication completed	Specifies a URL that must be successfully accessed to indicate to a WAP gateway that customer-supplied user authentication succeeded.
wap authentication service	Initiates customer-supplied user authentication.
wap authentication timeout	Specifies an interval after which the user is forced to reauthenticate.

wap authentication service

To initiate customer-supplied user authentication, use the **wap authentication service** command in global configuration mode. To disable the customer-supplied user authentication, use the **no** form of this command.

wap authentication service *url*

no wap authentication service

Syntax Description

url	URL used to initiate a customer-supplied user authentication scheme.

Defaults

By default, no URL is defined and no redirection is performed.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The **wap authentication service** command is used to enable a customer-supplied user authentication scheme to work with the Wireless Application Protocol (WAP) gateway.

If the **wap authentication service** command is configured, the WAP gateway will redirect the browser to the page configured by the **wap authentication service** command whenever a new session starts. The customer-supplied authentication process will then proceed and a page, specified by the **wap authentication completed** command, will be displayed if the authentication process is successful. Until the authentication process is successful, the WAP gateway will service only URLs specified by the **wap authentication service**, **wap authentication completed**, and **wap authentication prefix** commands, and all other page requests within that session will be denied.

Note The authentication scheme cannot be run when the WAP gateway is operating the connectionless WSP protocol to communicate between the wireless device and the WAP gateway.

Examples

The following example shows the URL http://www.company-name.com/auth/login.wml being configured as the URL to be accessed to initiate customer-supplied user authentication for the WAP gateway:

```
wap authentication service http://www.company-name.com/auth/login.wml
```

Related Commands

Command	Description
wap authentication completed	Specifies a URL that must be successfully accessed to indicate to a WAP gateway that customer-supplied user authentication succeeded.
wap authentication prefix	Specifies a URL prefix used to permit access to pages during the customer-supplied user authentication process.
wap authentication timeout	Specifies an interval after which the user is forced to reauthenticate.

wap authentication timeout

To specify an interval after which the user must reauthenticate, use the **wap authentication timeout** command in global configuration mode. To disable the timeout interval, use the **no** form of this command.

wap authentication timeout *seconds*

no wap authentication timeout

Syntax Description

seconds	Number of seconds before the user must reauthenticate.

Defaults

By default, no timeout interval is set.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The **wap authentication timeout** command is used with other authentication commands to enable a customer-supplied user authentication scheme to work with the Wireless Application Protocol (WAP) gateway.

If the **wap authentication service** command is configured, the WAP gateway will redirect the browser to the page configured by the **wap authentication service** command whenever a new session starts. The customer-supplied authentication process will then proceed and a page, specified by the **wap authentication completed** command, will be displayed if the authentication process is successful. Until the authentication process is successful, the WAP gateway will service only URLs specified by the **wap authentication service**, **wap authentication completed**, and **wap authentication prefix** commands, and all other page requests within that session will be denied.

Note The authentication scheme cannot be run when the WAP gateway is operating the connectionless WSP protocol to communicate between the wireless device and the WAP gateway.

Examples

The following example shows an authentication timeout interval of 1200 seconds (20 minutes) being set:

```
wap authentication timeout 1200
```

Related Commands

Command	Description
wap authentication completed	Specifies a URL that must be successfully accessed to indicate to a WAP gateway that customer-supplied user authentication succeeded.
wap authentication prefix	Specifies a URL prefix used to permit access to pages during the customer-supplied user authentication process.
wap authentication service	Initiates customer-supplied user authentication.

wap http maximum header-length

To specify how the gateway splits individual HTTP headers into multiple lines, use the **wap http maximum header-length** command in global configuration mode. To restore the maximum header length to the default value, use the **no** form of this command.

wap http maximum header-length *max-characters*

no wap http maximum header-length

Syntax Description

max-characters	Maximum number of characters allowed before the gateway splits a line of HTTP headers at the end of the next comma-separated value.

Defaults

By default, the maximum HTTP header length is set to 100 characters.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The **wap http maximum header-length** command controls how the gateway splits individual HTTP headers into multiple lines. As the gateway builds the HTTP header, if the addition of a comma-separated value causes the line to exceed this value, the header is continued on a new line. Some web content servers cannot handle long lines of text, and the use of line wrapping avoids this problem.

Examples

The following example shows the maximum HTTP header length set to 80 characters:

```
wap http maximum header-length 80
```

wap proxy-list

To specify a list of filter records that the Wireless Application Protocol (WAP) gateway can use to filter out requests that must be routed to a proxy server and not forwarded directly to the server specified in the URL, use the **wap proxy-list** command in global configuration mode. To reset the filter list to the default of a null string, use the **no** form of this command.

wap proxy-list *http-server*[**:***port-number*] [*proxy-server*[**:***port-number*]]

no wap proxy-list

Syntax Description

http-server[**:***port-number*]	Domain Name System (DNS) name or IP address corresponding to an HTTP server. Asterisk (*) wildcard symbols can be used to specify multiple servers. An optional *port-number* argument may be specified separated by a colon. The default port number is 8080.
proxy-server[**:***port-number*]	(Optional) DNS name or IP address of a proxy server. If no proxy server is specified, the page requests matching the filter will be requested directly from the server specified in the page request and not via a proxy server. An optional *port-number* argument may be specified separated by a colon. The default port number is 8080.

Defaults

By default, the filter list is a null string.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

When the WAP gateway receives a page request it searches through the configured proxy list records in the order in which they were entered until it finds a match. If the gateway finds a match, the request is forwarded to the specified proxy server. If no proxy server is specified, or no match is found, the request is serviced directly. Requests to be forwarded to the local domain are configured first, then any other requests that contain a dot in the domain name can be sent to a proxy server.

Examples

The following example shows a list of proxy list records that will enable the WAP gateway to forward page requests that are within the company-name company network directly to its server, while a proxy server will be used for all page requests outside the network:

First Record

```
wap proxy-list *.company-name.com
```

This record shows all requests to the company-name company host being serviced directly and not sent to a proxy server.

Second Record

```
wap proxy-list *.company-name.com:*
```

This record shows all requests to any explicitly specified port number on a company-name host being serviced directly and not passed to a proxy server.

Third Record

```
wap proxy-list *.* proxy.company-name.com:9090
```

This record shows all other requests including a dot in the page request being serviced by the specified company-name proxy server. The first two records capture all the local page requests except those requests that do not specify a company-name host. Requests are usually sent to the default port of 8080, but in this record, a different port number has been specified in the *proxy-server* argument.

Fourth Record

```
wap proxy-list *
```

This record shows all requests without a domain specification being serviced directly. This is the default action.

wap up-browser device-base

To specify a URL prefix to be used as a substitute for the string of characters (known as “device:base”) received in a request from an Unwired Planet (UP) browser, use the **wap up-browser device-base** command in global configuration mode. To restore the URL prefix to its default setting of a null string, use the **no** form of this command.

wap up-browser device-base *url*

no wap up-browser device-base

Syntax Description

url	URL prefix used by the Wireless Application Protocol (WAP) gateway as a substitute for “device:base” when received in requests from a UP browser.

Defaults

By default, the URL is a null string.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

This command works with the **wap up-browser device-home** global configuration command to provide support for UP v4.0 browsers that cannot be configured directly with a conventional URL. When a UP browser on the wireless device starts, it requests “device:home” instead of a URL. The WAP gateway translates the “device:home” request into the URL configured by the **wap up-browser device-home** command before forwarding the request to the content server. Subsequent page requests from the UP browser may specify the “device:base” string followed by a path and filename. The WAP gateway substitutes the URL prefix configured with the **wap up-browser device-base** command for the “device:base” string and then sends the request to the content server.

Examples

The following example shows the URL prefix of http://www.company-name.com/wapserver being set as the substitute URL for the “device:base” request sent by a UP browser:

```
wap up-browser device-base http://www.company-name.com/wapserver
```

Related Commands

Command	Description
wap up-browser device-home	Specifies a URL to be used when a request containing the string of characters (known as “device:home”) is received from a UP browser.

wap up-browser device-home

To specify a URL to be used when a request containing the string of characters (known as "device:home") is received from an Unwired Planet (UP) browser, use the **wap up-browser device-home** command in global configuration mode. To restore the URL prefix to its default setting of a null string, use the **no** form of this command.

wap up-browser device-home *url*

no wap up-browser device-home

Syntax Description

url	URL to be used when a UP browser request for "device:home" is received.

Defaults

By default, the URL is a null string.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

This command works with the **wap up-browser device-base** global configuration command to provide support for UP v4.0 browsers that cannot be configured directly with a conventional URL. When a UP browser on the wireless device starts, it requests the string of characters (known as "device:home") instead of a URL. The Wireless Application Protocol (WAP) gateway translates the "device:home" request into the URL configured by the **wap up-browser device-home** command before forwarding the request to the content server. Subsequent page requests from the UP browser may specify the "device:base" string followed by a path and filename. The WAP gateway substitutes the URL prefix configured with the **wap up-browser device-base** command for the "device:base" string and then sends the request to the content server.

The wireless client device does not know the actual web page as the WAP gateway translates the request for the home page to the actual URL that is downloaded. Relative URLs that may point to a particular place on a page should be avoided.

Examples

The following example shows the URL http://www.company-name.com/wapserver/index.wml being set as the substitute URL for the "device:home" request sent by a UP browser:

```
wap up-browser device-home http://www.company-name.com/wapserver/index.wml
```

Related Commands

Command	Description
wap up-browser device-base	Specifies a URL prefix to be used as a substitute for the string of characters (known as “device:base”) received in a request from a UP browser.

wap up-browser subno-suffix

To specify a gateway tag to be appended to the header field (known as "X-UP-subno") to identify the Wireless Application Protocol (WAP) gateway before transmission to a Wireless Markup Language (WML) content server, use the **wap up-browser subno-suffix** command in global configuration mode. To disable appending a gateway tag to the header field, use the **no** form of this command.

wap up-browser subno-suffix *gateway-tag*

no wap up-browser subno-suffix

Syntax Description

gateway-tag	Gateway tag to be added to the "X-UP-subno" header field to identify the WAP gateway.

Defaults

By default, no gateway tag is appended and the header is unchanged as it passes through the WAP gateway.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The **wap up-browser subno-suffix** command has been implemented to support compatibility with Unwired Planet (UP) browsers.

When a header field (known as "X-UP-subno") is received from UP browsers, it contains user identification information supplied by the browser. The WAP gateway forwards this header to the content server. Where several WAP gateways exist, a gateway tag can be appended to the header field to identify each individual WAP gateway to the content server.

Examples

The following example shows the gateway tag of gateway.company-name.com being added to the "X-UP-subno" header field:

```
wap up-browser subno-suffix gateway.company-name.com
```

wap wsp maximum sessions

To specify the maximum number of concurrent Wireless Session Protocol (WSP) sessions, use the **wap wsp maximum sessions** command in global configuration mode. To disable the restriction on the number of WSP sessions, use the **no** form of this command.

wap wsp maximum sessions *number*

no wap wsp maximum sessions

Syntax Description

number	Maximum number of concurrent WSP sessions.

Defaults

By default, there is no limit to the number of WSP sessions.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

Use the **wap wsp maximum sessions** command to specify a limit to the number of concurrent WSP sessions that the Wireless Application Protocol (WAP) gateway will create. When the maximum number of sessions is reached, a request to create a new WSP session will result in the deletion of the oldest existing session by the gateway, to ensure that the maximum number of WSP sessions is not exceeded. The oldest WSP session is usually in an idle state. A timeout interval can be set for idle WSP sessions by using the **wap wsp timeout session** global configuration command.

Note The practical limit for the maximum number of WSP sessions depends on the type of CPU in the router and the amount of memory.

Examples

The following example shows the maximum number of WSP sessions set to 1000:

```
wap wsp maximum sessions 1000
```

Related Commands

Command	Description
wap wsp timeout session	Specifies the number of seconds that a WSP session will continue to exist when there is no traffic for that session.

wap wsp timeout session

To specify the number of seconds that a Wireless Session Protocol (WSP) session will continue to exist when there is no traffic for that session, use the **wap wsp timeout session** command in global configuration mode. To restore the default timeout interval to 1 hour, use the **no** form of this command.

wap wsp timeout session *seconds*

no wsp timeout session

Syntax Description

seconds	Number of seconds before a WSP session with no traffic is timed out.

Defaults

By default, the number of seconds before the session times out is set to 3600 (1 hour).

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

Use the **wap wsp timeout session** command to manage the timeout intervals for WSP sessions with no traffic. Keeping the WSP session timeout interval short may improve software performance because terminating an idle WSP session frees the associated memory resources. A short timeout interval, however, may result in the gateway dropping existing sessions sooner than expected.

Examples

The following example shows the WSP session timeout set to 300 seconds (5 minutes):

```
wap wsp timeout session 300
```

Related Commands

Command	Description
wap wsp maximum sessions	Specifies the maximum number of concurrent WSP sessions.

wap wtls certificate gateway

To register a signed certificate to be used with the specified key-exchange algorithm that the Wireless Application Protocol (WAP) gateway will operate, use the **wap wtls certificate gateway** command in global configuration mode. To disable the use of a certificate with the specified key-exchange algorithm, use the **no** form of this command.

wap wtls certificate gateway {**rsa-512** | **rsa-768** | **rsa-unrestricted**} *filespec*

no wap wtls certificate gateway {**rsa-512** | **rsa-768** | **rsa-unrestricted**}

Syntax Description

rsa-512	Rivest, Shamir, and Adelman (RSA) key-exchange algorithm using a key length of 512 bits.
rsa-768	RSA key-exchange algorithm using a key length of 768 bits.
rsa-unrestricted	RSA key-exchange algorithm using a key length of 1024 bits.
filespec	Name, including the path, of the certificate file obtained from a certification authority (CA).

Defaults

By default, the WAP gateway does not associate any certificates with any RSA key-exchange algorithms.

Command Modes

Global configuration

Command History

Release	Modification
12.2(4)XR	This command was introduced.

Usage Guidelines

Use the **wap wtls key-pair generate** and the **wap wtls certificate generate csr** commands to generate an RSA key pair and an associated certificate signing request that is sent to a CA. The CA will return a certificate that can be registered using the **wap wtls certificate gateway** command. The certificate file must be copied to a file system that can always be accessed by the WAP gateway. We recommend that certificates be stored in NVRAM. The software checks that the specified certificate matches the associated RSA key pair before enabling use of the associated key-exchange algorithm.

Use the **wap wtls certificate gateway** command to establish a certificate to be associated with a key-exchange algorithm, or to replace an existing certificate that is expiring, for example.

When disabling the use of a certificate with a specified key-exchange algorithm, a warning may be displayed if the associated key-exchange algorithm is currently enabled. The warning notes that the associated key-exchange algorithm will also be disabled because it cannot operate without a certificate.

Note The certificate must remain in the specified location because it will be read every time the router is restarted. If a certificate file cannot be read, the associated key-exchange algorithm will not be enabled.

Examples

The following example shows the WAP gateway being configured to register a certificate file named cert512.cer to be used with the RSA key-exchange algorithm with the key length limited to 512 bits:

```
wap wtls certificate gateway rsa-512 nvram:cert512.cer
```

Related Commands

Command	Description
wap wtls certificate generate csr	Generates a CSR to be used to request a signed certificate from a CA.
wap wtls key-pair generate	Generates an RSA key pair to use with a specified key-exchange algorithm operated by the WAP gateway.

wap wtls certificate generate csr

To generate a certificate signing request (CSR) to be used to request a signed certificate from a certification authority (CA), use the **wap wtls certificate generate csr** command in global configuration mode.

wap wtls certificate generate csr {**rsa-512** | **rsa-768** | **rsa-unrestricted**} [**common** *common-name*] [**country** *country-code*] [**state** *state*] [**locality** *city*] [**organization** *org-name*] [**orgunit** *org-unit-name*] **filespec** *filespec*

Syntax Description

rsa-512	Rivest, Shamir, and Adelman (RSA) key-exchange algorithm using a key length of 512 bits.
rsa-768	RSA key-exchange algorithm using a key length of 768 bits.
rsa-unrestricted	RSA key-exchange algorithm using a key length of 1024 bits.
common *common-name*	Name of the gateway. If the name is not specified, the host name and domain name of the router, if configured, will be used. The *common-name* argument corresponds to the X.509 attribute of CN and may contain only characters from the PrintableString character set defined by Abstract Syntax Notation One (ASN.1). Permitted characters for the *common-name* argument are shown in Table 3.
country *country-code*	(Optional) A two-character country code. The *country-code* argument corresponds to the X.509 attribute of C and may contain only characters from the PrintableString character set defined by ASN.1. Permitted characters for the *country-code* argument are shown in Table 3.
state *state*	(Optional) State or province name. The *state* argument corresponds to the X.509 attribute of S and may contain only characters from the PrintableString character set defined by ASN.1. Permitted characters for the *state* argument are shown in Table 3.
locality *city*	(Optional) Name of the nearest large city. The *city* argument corresponds to the X.509 attribute of L and may contain only characters from the PrintableString character set defined by ASN.1. Permitted characters for the *city* argument are shown in Table 3.
organization *org-name*	(Optional) Organization name. The *org-name* argument corresponds to the X.509 attribute of O and may contain only characters from the PrintableString character set defined by ASN.1. Permitted characters for the *org-name* argument are shown in Table 3.
orgunit *org-unit-name*	(Optional) Organization unit name. The *org-unit-name* argument corresponds to the X.509 attribute of OU and may contain only characters from the PrintableString character set defined by ASN.1 . Permitted characters for the *org-unit-name* argument are shown in Table 3.
filespec *filespec*	Name, including path, of the file to which the CSR should be written. If the name is not specified, the gateway will send the output to the console.

Defaults

By default, the Wireless Application Protocol (WAP) gateway does not generate a CSR.

Command Modes

Global configuration

Command History

Release	Modification
12.2(4)XR	This command was introduced.

Usage Guidelines

Use the **wap wtls key-pair generate** command to create an appropriate RSA key pair before generating a CSR using the **wap wtls certificate generate csr** command because the RSA public key is used as part of the CSR and the resulting signed certificate. The generated CSR is encoded using Public-Key Cryptography Standard #10 (PKCS #10), a standard syntax for certificate signing requests. The CSR that is generated can be written to the specified file or, if no filename is specified, output to the screen. The text of the CSR in the file or on the console must be sent to a CA in the exact format that it was output by the router. For screen output, select the entire block of text. The CSR is usually submitted to a CA via e-mail or a web application. The CSR and the submitting company are validated by the CA and a signed certificate, authenticating the public key from the gateway, is returned. The signed certificate must be registered with a key-exchange algorithm that the WAP gateway will operate, using the **wap wtls certificate gateway** command.

The CSR includes the name of the gateway that will appear on the certificate. The name must correspond to an X.509 name and can consist of several components, referred to as “attributes.” The CN attribute, which is generated using the **common** keyword and *common-name* argument, or defaults to the host name and domain name, is mandatory. Some CAs require that additional attributes be supplied. These attributes may be included in the CSR using the other, optional keywords and their associated arguments.

The decision over which CA to use depends on the wireless devices that you are operating. Some handsets are preconfigured with the root certificates of the main certification authorities, and you can choose a CA that is already trusted by the handset. If the handset allows a certificate to be downloaded, you must contact a CA to obtain a Wireless Transport Layer Security (WTLS) root certificate for the key-exchange algorithm that the WAP gateway is to operate. The root certificate can be downloaded by the handsets from a web server.

Table 3 describes the PrintableString character set.

Table 3 PrintableString Character Set

Character	Description
a–z	Lowercase and uppercase letters (mixed use of case is acceptable).
0–9	Numerals
"	Quotation mark
(	Opening parenthesis
)	Closing parenthesis
+	Plus sign
,	Comma
-	Minus sign
.	Period

Table 3 PrintableString Character Set

Character	Description
/	Slash mark
:	Colon
?	Question mark
“ ”	Blank space

Note The **wap wtls certificate generate csr** command is never saved in the router configuration files.

Examples

The following example shows the WAP gateway being configured to generate an RSA key pair for the **rsa-512** form of the RSA key-exchange encryption algorithm, and then to generate a CSR for use with the generated key pair:

```
wap wtls key-pair generate rsa-512
wap wtls certificate generate csr rsa-512 common wap.company-name.com country us filespec
nvram:cert512.csr
```

Related Commands

Command	Description
wap wtls certificate gateway	Registers a signed certificate from a CA with the specified key-exchange algorithm.
wap wtls key-pair generate	Generates an RSA key pair to use with a specified key-exchange algorithm operated by the WAP gateway.

wap wtls encryption

To specify the encryption algorithms that the Wireless Application Protocol (WAP) gateway will operate, use the **wap wtls encryption** command in global configuration mode. To disable the use of an encryption algorithm, use the **no** form of this command.

wap wtls encryption {**all** | *cipher-block*}

no wap wtls encryption {**all** | *cipher-block*}

Syntax Description

all	All the supported encryption algorithms.
cipher-block	One or more of the following Rivest Cipher 5 (RC5) cipher block chaining algorithm keywords: • **rc5-cbc-40**—Key length of 40 bits. • **rc5-cbc-56**—Key length of 56 bits. • **rc5-cbc-128**—Key length of 128 bits.

Defaults

By default, all the RC5-based algorithms are enabled.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

Block cipher encryption techniques are designed to disguise plain text patterns that otherwise might generate patterns of encrypted cipher text. Any repeated sequences can facilitate cracking of the algorithm.

The WAP gateway can be configured to operate all the encryption algorithms or a list specifying one or more of the options. To decide which encryption algorithms to configure, you must consider several factors. A shorter key length is easier to compute and will impose less overhead on the processor than a longer key length, but a shorter key length offers weaker security. The level of security you need to configure will also be determined by the type of information that can be accessed through the gateway. Confidential corporate information often requires a high level of security.

Use the **wap wtls encryption** command in conjunction with the **wap wtls hash** and **wap wtls key-exchange** global configuration commands to help establish and operate a secure WAP session.

Note Many wireless devices support only a subset of the available algorithms. Configuring the gateway to operate only one or two algorithms may prevent some wireless devices from accessing the gateway.

Examples

The following example shows the WAP gateway being configured to disable the use of the RC5 cipher block chaining encryption algorithm with an effective key length of 128 bits. All the RC5-based algorithms are enabled by default.

```
no wap wtls encryption rc5-cbc-128
```

Related Commands

Command	Description
wap wtls hash	Specifies the hash algorithms operated by the WAP gateway.
wap wtls key-exchange	Specifies the key-exchange algorithms operated by the WAP gateway.

wap wtls hash

To specify the hash algorithms that the Wireless Application Protocol (WAP) gateway will operate, use the **wap wtls hash** command in global configuration mode. To disable the use of a hash algorithm, use the **no** form of this command.

wap wtls hash {**all** | *algorithm*}

no wap wtls hash {**all** | *algorithm*}

Syntax Description

all	All the supported hash algorithms.
algorithm	One or more of the following hash algorithm keywords: • **md5-40**—Message Digest 5 (MD5) algorithm using the first 5 bytes of the output. • **md5-80**—MD5 algorithm using the first 10 bytes of the output. • **md5-128**—MD5 algorithm using all the output. • **sha-0**—Secure Hash Algorithm 1 (SHA-1) with no keyed MAC being calculated. • **sha-40**—SHA-1 algorithm using the first 5 bytes of the output. • **sha-80**—SHA-1 algorithm using the first 10 bytes of the output. • **sha-160**—SHA-1 algorithm using all the output. • **sha-xor-40**—A 5-byte exclusive OR (XOR) checksum is generated. The input data is first divided into multiple blocks of 5 bytes. If the final block is fewer than 5 bytes, it is padded with zeros. All blocks are then grouped together, and the XOR checksum is applied to obtain the result.

Defaults

By default, the WAP gateway operates the sha-40, sha-80, and sha-160 hash algorithms.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

Hash algorithms are used to construct a digital signature for encrypted text to prevent attempts to modify the original encrypted text.

The WAP gateway can be configured to operate all the hash algorithms or a list specifying one or more of the options. The level of security you need to configure will be determined by the type of information that can be accessed through the gateway. Confidential corporate information often requires a high level of security.

Use the **wap wtls hash** command in conjunction with the **wap wtls encryption** and **wap wtls key-exchange** global configuration commands to help establish and operate a secure WAP session.

Note Many wireless devices support only a subset of the available algorithms. Configuring the gateway to operate only one or two algorithms may prevent some wireless devices from accessing the gateway.

Examples

The following example shows the WAP gateway being configured to disable the use of the SHA-1 hash algorithm where the complete output is used:

```
no wap wtls hash sha-160
```

Related Commands

Command	Description
wap wtls encryption	Specifies the encryption algorithms operated by the WAP gateway.
wap wtls key-exchange	Specifies the key-exchange algorithms operated by the WAP gateway.

wap wtls key-exchange

To specify a key-exchange algorithm that the Wireless Application Protocol (WAP) gateway will operate, use the **wap wtls key-exchange** command in global configuration mode. To disable the use of a key-exchange algorithm, use the **no** form of this command.

wap wtls key-exchange {**all** | *algorithm*}

no wap wtls key-exchange {**all** | *algorithm*}

Syntax Description

all	All the supported key-exchange algorithms. **Note** For the algorithms that perform authentication, this keyword will not enable the algorithm unless associated certificates are registered.
algorithm	One or more of the following hash algorithm keywords: • **dh-anon-512**—Diffie-Hellman (DH) key-exchange algorithm without authentication. Key length is limited to 512 bits. • **dh-anon-768**—DH key-exchange algorithm without authentication. Key length is limited to 768 bits. • **dh-anon-unrestricted—**DH key-exchange algorithm without authentication. Key length is unrestricted. • **rsa-anon-512**—Rivest, Shamir, and Adelman (RSA) key-exchange algorithm without authentication. Key length is limited to 512 bits. • **rsa-anon-768**—RSA key-exchange algorithm without authentication. Key length is limited to 768 bits. • **rsa-anon-unrestricted**—RSA key-exchange algorithm without authentication. Key length is unrestricted. (The gateway currently uses a key length of 1024 bits.) • **rsa-512**—RSA key-exchange algorithm with authentication using RSA certificates. Key length is limited to 512 bits. • **rsa-768**—RSA key-exchange algorithm with authentication using RSA certificates. Key length is limited to 768 bits. • **rsa-unrestricted**—RSA key-exchange algorithm with authentication using RSA certificates. Key length is unrestricted. (The gateway currently uses a key length of 1024 bits.)

Defaults

By default, the WAP gateway operates all the key-exchange algorithms that do not perform authentication.

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.
12.2(4)XR	The **rsa-512**, **rsa-768**, and **rsa-unrestricted** keywords were added.

Usage Guidelines

Key-exchange algorithms allow cryptographic keys to be securely generated and exchanged between devices over an unsecured connection.

The WAP gateway can be configured to operate all the key-exchange algorithms or a list specifying one or more of the options. To decide which key-exchange algorithms to configure, you must consider several factors. A shorter key length is easier to compute and will impose less overhead on the processor than a longer key length, but a shorter key length offers weaker security. The level of security you need to configure will also be determined by the type of information that can be accessed through the gateway. Confidential corporate information often requires a high level of security.

Use the **wap wtls key-pair generate** to generate an RSA key pair and the **wap wtls certificate generate csr** command to generate an associated certificate signing request that is sent to a certification authority (CA). The CA will return a certificate that may be registered using the **wap wtls certificate gateway** command before enabling the **rsa-512**, **rsa-768**, and **rsa-unrestricted** algorithms. The software checks that a certificate and matching RSA key pair exist before enabling the **rsa-512**, **rsa-768**, and **rsa-unrestricted** algorithms.

The WAP gateway can operate key-exchange algorithms that do not require authentication, such as **rsa-anon-512**, and algorithms that do require authentication, such as **rsa-512**. If you want the WAP gateway to operate only those algorithms that perform authentication, we recommend that you disable all the other algorithms to prevent a WAP client selecting an algorithm that does not require authentication during the negotiation process.

Use the **wap wtls key-exchange** command in conjunction with the **wap wtls encryption** and **wap wtls hash** global configuration commands to help establish and operate a secure WAP gateway.

Note Many wireless devices support only a subset of the available algorithms. Configuring the gateway to operate only one or two algorithms may prevent some wireless devices from accessing the gateway.

Examples

The following example shows the WAP gateway being configured to disable use of the DH key-exchange encryption algorithm without authentication and with the key length limited to 512 bits:

```
no wap wtls key-exchange dh-anon-512
```

The following example shows the WAP gateway being configured to use the RSA key-exchange algorithm with authentication using RSA certificates and with the key length limited to 512 bits:

```
wap wtls key-exchange rsa-512
```

Related Commands

Command	Description
wap wtls certificate gateway	Registers a signed certificate from a CA with the specified key-exchange algorithm.
wap wtls certificate generate csr	Generates a CSR to be used to request a signed certificate from a CA.

Command	Description
wap wtls hash	Specifies the hash algorithms operated by the WAP gateway.
wap wtls key-pair generate	Generates an RSA key pair to use with a specified key-exchange algorithm operated by the WAP gateway.

wap wtls key-pair generate

To generate a Rivest, Shamir, and Adelman (RSA) key pair for use with the specified key-exchange algorithm that the Wireless Application Protocol (WAP) gateway will operate, use the **wap wtls key-pair generate** command in global configuration mode.

wap wtls key-pair generate {**rsa-512** | **rsa-768** | **rsa-unrestricted**}

Syntax Description

rsa-512	RSA key pair with a key length of 512 bits.
rsa-768	RSA key pair with a key length of 768 bits.
rsa-unrestricted	RSA key pair with a key length of 1024 bits.

Defaults

By default, the WAP gateway automatically generates a new RSA key pair for use with the anonymous forms of the RSA key-exchange algorithm when the gateway is restarted unless an RSA key pair has previously been manually generated by the user.

Command Modes

Global configuration

Command History

Release	Modification
12.2(4)XR	This command was introduced.

Usage Guidelines

RSA keys are always generated in pairs with one public RSA key and one private RSA key. The gateway uses the same key pair for operating both the anonymous and non anonymous forms of the RSA key-exchange algorithm. If RSA keys for the specified key-exchange algorithm and an associated certificate already exist, a message is displayed warning that the gateway will disassociate the certificate from the key pair and disable the key-exchange algorithm before the new key pair is generated.

To prevent a Wireless Transport Layer Security (WTLS) key exchange occurring using RSA key data that is being modified by the key generation software, the WAP gateway does not allow RSA key pairs to be generated while it is running. Use the **no wap all** command to disable the WAP gateway before using the **wap wtls key-pair generate** command.

Use the **wap wtls key-pair generate** command if the gateway is to operate a WTLS Class II key-exchange algorithm. After the key pair is generated, use the **wap wtls certificate generate csr** and the **wap wtls certificate gateway** commands to obtain and register an associated certificate.

If an RSA key pair has been generated using the **wap wtls key-pair generate** command, it will be maintained across router restarts in NVRAM.

Note The **wap wtls key-pair generate** command is never saved in the normal router configuration files. The keys that are generated are saved in the private configuration in NVRAM, which is never displayed to the user or backed up to another device.

Examples

The following example shows the WAP gateway being configured to generate an RSA key pair for use with the RSA key-exchange algorithm with authentication using RSA certificates and with the key length limited to 512 bits:

```
wap wtls key-pair generate rsa-512
```

Related Commands

Command	Description
wap wtls certificate gateway	Registers a signed certificate from a CA with the specified key-exchange algorithm.
wap wtls certificate generate csr	Generates a CSR to be used to request a signed certificate from a CA.

wap wtls key-pair remove

To delete a Rivest, Shamir, and Adelman (RSA) key pair being used with the specified key-exchange algorithm that the Wireless Application Protocol (WAP) gateway will operate, use the **wap wtls key-pair remove** command in global configuration mode.

wap wtls key-pair remove {**rsa-512** | **rsa-768** | **rsa-unrestricted**}

Syntax Description

rsa-512	RSA key pair with a key length of 512 bits.
rsa-768	RSA key pair with a key length of 768 bits.
rsa-unrestricted	RSA key pair with a key length of 1024 bits.

Defaults

No RSA key pairs are deleted.

Command Modes

Global configuration

Command History

Release	Modification
12.2(4)XR	This command was introduced.

Usage Guidelines

Use the **wap wtls key-pair remove** command to delete a key pair previously generated by the **wap wtls key-pair generate** command. Before deleting a key pair, use the **wap wtls key-exchange** command to disable the use of the associated key-exchange algorithm. If the **wap wtls key-pair remove** command is used to delete a key pair corresponding to a key-exchange algorithm that is enabled, a warning is displayed to note that the algorithm will be disabled if the key pair is deleted. In addition, if a registered certificate corresponding to the deleted key pair exists, it will become invalid if the key pair is deleted because the certificate is useless after the corresponding key pair has been removed. Use the **no wap wtls certificate gateway** global configuration command to disable a certificate.

Examples

The following example shows the WAP gateway being configured to disable the **rsa-unrestricted** form of the RSA key-exchange algorithm, and the associated certificate, before the RSA key pair is deleted:

```
no wap wtls key-exchange rsa-unrestricted
no wap wtls certificate gateway rsa-unrestricted
wap wtls key-pair remove rsa-unrestricted
```

Related Commands

Command	Description
wap wtls certificate gateway	Registers a signed certificate from a CA with the specified key-exchange algorithm.
wap wtls key-exchange	Specifies the key-exchange algorithms operated by the WAP gateway.
wap wtls key-pair generate	Generates an RSA key pair to use with a specified key-exchange algorithm operated by the WAP gateway.

wap wtls timeout connection

To specify an interval after which an inactive Wireless Transport Layer Security (WTLS) connection will be closed by the gateway, use the **wap wtls timeout connection** command in global configuration mode. To restore the connection timeout interval to its default setting, use the **no** form of this command.

wap wtls timeout connection *seconds*

no wap wtls timeout connection

Syntax Description

seconds	Number of seconds before an inactive WTLS connection is closed.

Defaults

By default, the WTLS connection timeout interval is 1800 seconds (30 minutes).

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

The Wireless Application Protocol (WAP) gateway has no reliable method for determining when a user is finished with a session if the session is not explicitly closed by the browser. Use the **wap wtls timeout connection** command to ensure a balance between closing an inactive WTLS connection before the user is finished with the connection and using router memory with a series of inactive connections.

Other global configuration commands that you can use to control the behavior of the WTLS layer are **wap wtls timeout handshake**, **wap wtls timeout key**, and **wap wtls timeout session**.

Examples

The following example shows the WTLS connection timeout interval set to 20 minutes:

```
wap wtls timeout connection 1200
```

Related Commands

Command	Description
wap wtls timeout handshake	Specifies the interval that the gateway allows for the WTLS handshake to complete.
wap wtls timeout key	Specifies an interval during which the gateway will retain an unused WTLS master key.
wap wtls timeout session	Specifies an interval after which an inactive WTLS session will expire.

wap wtls timeout handshake

To specify the interval that the gateway allows for the Wireless Transport Layer Security (WTLS) handshake to complete, use the **wap wtls timeout handshake** command in global configuration mode. To restore the handshake timeout interval to its default setting, use the **no** form of this command.

wap wtls timeout handshake *seconds*

no wap wtls timeout handshake

Syntax Description

seconds	Number of seconds allowed for the WTLS handshake process to complete.

Defaults

By default, the WTLS handshake timeout interval is 120 seconds (2 minutes).

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

Use the **wap wtls timeout handshake** command to ensure that an incomplete WTLS handshake process is terminated after an appropriate interval. The WTLS handshake process may fail for a variety of reasons but if the client does not respond during the timeout interval, the process is terminated.

Other global configuration commands that you can use to control the behavior of the WTLS layer are **wap wtls timeout connection**, **wap wtls timeout key**, and **wap wtls timeout session**.

Examples

The following example shows the WTLS handshake timeout set to 3 minutes:

```
wap wtls timeout handshake 180
```

Related Commands

Command	Description
wap wtls timeout connection	Specifies an interval after which an inactive WTLS connection will expire.
wap wtls timeout key	Specifies an interval during which the gateway will retain an unused WTLS master key.
wap wtls timeout session	Specifies an interval after which an inactive WTLS session will expire.

wap wtls timeout key

To specify an interval during which the gateway will retain a Wireless Transport Layer Security (WTLS) master key when the session is unused, use the **wap wtls timeout key** command in global configuration mode. To restore the master key timeout interval to its default setting, use the **no** form of this command.

wap wtls timeout key *seconds*

no wap wtls timeout key

Syntax Description

seconds	Number of seconds the gateway will retain an unused WTLS session and master key.

Defaults

By default, the WTLS session key timeout interval is 2,678,400 seconds (31 days).

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

After the timeout interval, the wireless device will need to renegotiate a new key when it contacts the Wireless Application Protocol (WAP) gateway.

Note The WTLS master key is not maintained across router reboots. After the router is rebooted, a new master key must be established when the wireless device connects to the WAP gateway.

Other global configuration commands that you can use to control the behavior of the WTLS layer are **wap wtls timeout connection**, **wap wtls timeout handshake**, and **wap wtls timeout session**.

Examples

The following example shows the timeout interval for an unused WTLS master key set to 1 day:

```
wap wtls timeout key 86400
```

Related Commands

Command	Description
wap wtls timeout connection	Specifies an interval after which an inactive WTLS connection will expire.
wap wtls timeout handshake	Specifies the interval that the gateway allows for the WTLS handshake to complete.
wap wtls timeout session	Specifies an interval after which an inactive WTLS session will expire.

wap wtls timeout session

To specify an interval after which an inactive Wireless Transport Layer Security (WTLS) session will be closed by the gateway, use the **wap wtls timeout session** command in global configuration mode. To restore the session timeout interval to its default setting, use the **no** form of this command.

wap wtls timeout session *seconds*

no wap wtls timeout session

Syntax Description

seconds	Number of seconds before an inactive WTLS session is closed.

Defaults

By default, the WTLS session timeout interval is 3600 seconds (1 hour).

Command Modes

Global configuration

Command History

Release	Modification
12.2(2)XR	This command was introduced.

Usage Guidelines

An inactive WTLS session is defined as a session that has no associated WTLS connections.

The Wireless Application Protocol (WAP) gateway has no reliable method for determining when a user is finished with a session if the session is not explicitly closed by the browser. Use the **wap wtls timeout session** command to ensure a balance between closing an inactive WTLS session before the user is finished with the session and using router memory with a series of inactive sessions.

Other global configuration commands that you can use to control the behavior of the WTLS layer are **wap wtls timeout connection**, **wap wtls timeout handshake**, and **wap wtls timeout key** commands.

Examples

The following example shows the WTLS session timeout interval set to 90 minutes:

```
wap wtls timeout session 5400
```

Related Commands

Command	Description
wap wtls timeout connection	Specifies an interval after which an inactive WTLS connection will expire.
wap wtls timeout handshake	Specifies the interval that the gateway allows for the WTLS handshake to complete.
wap wtls timeout key	Specifies an interval during which the gateway will retain an unused WTLS master key.

Glossary

2G—second generation. Generic name for second-generation digital mobile networks such as GSM. See GSM.

CDMA—code division multiple access. A technology for digital transmission used in North America, Japan, and Korea.

CA—certification authority. An organization that validates certificate signing requests and the submitter before returning a signed certificate to be used with a corresponding authentication algorithm.

CSR—certificate signing request. Text in a set format such as PKCS #10, and including a public key, is sent to a certification authority, which validates the request and returns a signed certificate file.

GPRS—general packet radio service. An upgrade to the existing GSM digital mobile networks to provide higher-speed data services.

GSM—global system for mobile communication. Popular 2G digital cellular mobile network standard, widely deployed in Europe.

PKCS—public key cryptography standard. A standard syntax for certificate signing requests.

UDP—User Datagram Protocol. Connectionless transport layer protocol in the IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery, requiring that error processing and retransmission be handled by other protocols.

UP browser—WAP browser developed by Unwired Planet (now Openwave).

WAP—Wireless Application Protocol. Protocol suite to enable the delivery and presentation of data on mobile phones and other wireless devices.

WDP—Wireless Datagram Protocol. Protocol that is similar to UDP but has segmentation and reassembly capabilities. Used where bearers can support only small packet sizes.

WML—Wireless Markup Language. An XML-based markup language used to describe content for delivery to WAP devices. WML is optimized for mobile wireless devices and uses the concept of a deck of cards where a series of related cards (often screens) are downloaded from the server at the same time. The downloaded deck is navigated locally, saving many separate, and potentially expensive, requests to the remote server.

WMLC—Wireless Markup Language Compressed. Compressed WML content is WML content received from a web server and compressed by a WAP gateway, using a process called tokenization. WMLC content is then forwarded to the WAP-enabled device.

WMLS—Wireless Markup Language Script. A scripting language derived from ECMAscript and used with WML.

WSP—Wireless Session Protocol. WAP protocol that is broadly equivalent to HTTP.

WTAI—Wireless Telephony Application Interface. An API to the telephony capabilities in a mobile phone.

WTLS—Wireless Transport Layer Security. A protocol to ensure data integrity, privacy, and authentication to transactions on a wireless device using WAP. WTLS is derived from the Transport Layer Security (TLS) protocol but has been optimized for use over narrowband communication channels.

WTP—Wireless Transaction Protocol. A lightweight protocol that provides reliable transport for WSP. See also WSP.