WebSphere® DataPower XML Integration Appliance XI50

IBM

Version 3.7.2

Command Reference

WebSphere® DataPower XML Integration Appliance XI50

IBM

Version 3.7.2

Command Reference

Note
Before using this information and the product it supports, read the information in "Notices and trademarks" on page 1119.

First Edition (December 2008)

This edition applies to version 3, release 7, modification 2, level 0 of IBM WebSphere DataPower XML Integration Appliance XI50 and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Preface

IBM® WebSphere® DataPower® SOA Appliances are purpose-built, easy-to-deploy network appliances that simplify, help secure, and accelerate your XML and Web services deployments while extending your SOA infrastructure. These appliances offer an innovative, pragmatic approach to harness the power of SOA while simultaneously enabling you to leverage the value of your existing application, security, and networking infrastructure investments.

Who should read this document

This document is intended for administrators of IBM WebSphere DataPower who are responsible for the configuration and maintenance of web services, security, and data communications equipment. These administrators are expected to have familiarity with XML and XSLT.

This document assumes that you have installed and initially configured the appliance as described in the *IBM WebSphere DataPower SOA Appliances: 9003: Installation Guide* or in the *IBM WebSphere DataPower SOA Appliances: Type 9235: Installation Guide*, depending on the model type.

Publications

The IBM WebSphere DataPower library is organized into the following categories:

Installation and upgrade documentation

- *IBM WebSphere DataPower SOA Appliances: 9003: Installation Guide*

  Provides instructions for installing and powering up the Type 7993 (9003) appliance, creating a startup configuration script, and placing the appliance in operation.
- *IBM WebSphere DataPower SOA Appliances: Type 9235: Installation Guide*

  Provides instructions for installing and powering up the Type 9235 appliance, creating a startup configuration script, and placing the appliance in operation.
- *IBM WebSphere DataPower SOA Appliances: Type 9235: Hardware Problem Determination and Service Guide*

  Provides information about diagnosing and troubleshooting hardware problems, ordering consumable replacement parts, and replacing parts.
- *IBM WebSphere DataPower SOA Appliances: Upgrade and Rollback Guide: Generation 2 Firmware*

  Provides instructions for upgrading Generation 2 firmware and for rolling back firmware upgrades.

Administration documentation

- *IBM WebSphere DataPower SOA Appliances: Appliance Overview*

  Provides an introduction and understanding of the IBM Websphere DataPower SOA appliances.
- *IBM WebSphere DataPower SOA Appliances: Administrators Guide*

  Provides instructions for using the DataPower GUI for managing user access, network access, appliance configuration and system configuration of the appliance.
- *IBM WebSphere DataPower SOA Appliances: Hardware Security Module Guide*

  A user guide for using a Hardware Security Module (HSM) installed in the appliance.

Development documentation

- *IBM WebSphere DataPower SOA Appliances: XSL Accelerator Developers Guide*

  Provides instructions for using the WebGUI to configure XSL Proxy and XSL Co-Processor services.
- *IBM WebSphere DataPower SOA Appliances: XML Firewall Developers Guide*

  Provides instructions for using the WebGUI to configure XML Firewall services.
- *IBM WebSphere DataPower SOA Appliances: Web Application Firewall Developers Guide*

  Provides instructions for using the WebGUI to configure Web Application Firewall services.
- *IBM WebSphere DataPower SOA Appliances: Multi-Protocol Gateway Developers Guide*

  Provides instructions for using the WebGUI to configure Multiple-Protocol Gateway services.
- *IBM WebSphere DataPower SOA Appliances: Web Service Proxy Developers Guide*

  Provides instructions for using the WebGUI to configure Web Service Proxy services.
- *IBM WebSphere DataPower SOA Appliances: B2B Gateway Developers Guide*

  Provides instructions for using the WebGUI to configure B2B Gateway services.
- *IBM WebSphere DataPower SOA Appliances: Low Latency Messaging Developers Guide*

  Provides instructions for using the WebGUI to configure a DataPower appliance for low latency messaging.

Reference documentation

- Product-specific documentation for using commands from the command line. The documentation is specific to each of the following products. Each document provides an alphabetical listing of all commands with syntactical and functional descriptions.
  - *IBM WebSphere DataPower XML Accelerator XA35: Command Reference*
  - *IBM WebSphere DataPower XML Security Gateway XS40: Command Reference*
  - *IBM WebSphere DataPower XML Integration Appliance XI50: Command Reference*
  - *IBM WebSphere DataPower B2B Appliance XB60: Command Reference*
  - *IBM WebSphere DataPower Low Latency Messaging Appliance XM70: Command Reference*

- *IBM WebSphere DataPower SOA Appliances: Extension Elements and Functions Catalog*

  Provides programming information about the usage of DataPower XSLT extension elements and extension functions.

Integration documentation

The following documents are available for managing the integration of related products that can be associated with the DataPower appliance:

- *IBM WebSphere DataPower SOA Appliances: Integrating with ITCAM*

  Provides concepts for integrating the DataPower appliance with IBM Tivoli Composite Application Management for SOA.
- *IBM WebSphere DataPower SOA Appliances: Integrating with WebSphere Transformation Extender*

  Provides concepts for integrating the DataPower appliance with WebSphere Transformer Extender.
- *IBM WebSphere DataPower XML Integration Appliance XI50: WebSphere MQ Interoperability*

  Explains the concepts and common use patterns for connecting DataPower services to WebSphere MQ systems.

Problem determination documentation

- *IBM WebSphere DataPower SOA Appliances: Problem Determination Guide*

  Provides troubleshooting and debugging tools.

Supplemental documentations

- *IBM WebSphere DataPower SOA Appliances: Understanding Web Services Policy*

  Provides conceptual information about how the DataPower appliance can use Web Services Policy (WS-Policy).
- *IBM WebSphere DataPower SOA Appliances: Understanding WS-Addressing*

  Provides conceptual information about how the DataPower appliance can use WS-Addressing.
- *IBM WebSphere DataPower SOA Appliances: Understanding LTPA*

  Provides conceptual information about how the DataPower appliance can use Lightweight Third Party Authentication.
- *IBM WebSphere DataPower SOA Appliances: Understanding SPNEGO*

  Provides conceptual information about how the DataPower appliance can use SPNEGO.
- *IBM WebSphere DataPower SOA Appliances: Optimizing through Streaming*

  Provides conceptual information about and procedures for optimizing the DataPower appliance through streaming.
- *IBM WebSphere DataPower SOA Appliances: Securing the Last Mile*

  Provides conceptual information about and procedures for understanding the DataPower appliance while securing the last mile.
- *IBM WebSphere DataPower SOA Appliances: Configuring the DoD PKI*

  Provides conceptual information about and procedures for configuring the DataPower appliance with Department of Defense Public Key Infrastructure.

Reading syntax statements

The reference documentation uses the following special characters to define syntax:

[ ] Identifies optional options. Options not enclosed in brackets are required.

... Indicates that you can specify multiple values for the previous option.

| Indicates mutually exclusive information. You can use the option to the left of the separator or the option to the right of the separator. You cannot use both options in a single use of the command.

{ } Delimits a set of mutually exclusive options when one of the options is required. If the options are optional, they are enclosed in brackets ([ ]).

When the order of the options or parameters must be used in a specific order, the syntax statement shows this order.

Directories on the appliance

The file system contains many examples and critical configuration files. These directories and their contents are as follows:

audit: This directory contains the audit logs. Each appliance contains only one audit: directory. This directory cannot be the destination of a copy. This directory is available from the command line in the `default` domain only.

To view the audit log from the WebGUI, select **Status** → **View Logs** → **Audit Log**.

cert: This encrypted directory contains private key and certificate files that services use in the domain. You can add, delete, and view files, but you cannot modify these files while in the domain. Each application domain contains one cert: directory. This directory is not shared across domains.

chkpoints:
This directory contains the configuration checkpoint files for the appliance. Each application domain contains one chkpoints: directory. This directory is not shared across domains.

config:
This directory contains the configuration files for the appliance. Each application domain contains one config: directory. This directory is not shared across domains.

dpcert:
This encrypted directory contains files that the appliance itself uses. This directory is available from the command line in the `default` domain only.

export:
This directory contains the exported configurations that are created with the Export Configuration utility. Each application domain contains one export: directory. This directory is not shared across domains.

image: This directory contains the firmware images (primary and secondary) for the appliance. This directory is where firmware images are stored typically during an upload or fetch operation. Each appliance contains only one image: directory. This directory is available in the `default` domain only.

local: This directory contains miscellaneous files that are used by the services within the domain, such as XSL, XSD, and WSDL files. Each application domain contains one local: directory. This directory can be made visible to

other domains. When viewed from other domains, the directory name changes from local: to the name of the application domain.

logstore:
This directory contains log files that are stored for future reference. Typically, the logging targets use the logtemp: directory for active logs. You can move log files to the logstore: directory. Each application domain contains one logstore: directory. This directory is not shared across domains.

logtemp:
This directory is the default location of log files, such as the appliance-wide default log. This directory can hold only 13 MB. This directory cannot be the destination of a copy. Each application domain contains one logtemp: directory. This directory is not shared across domains.

pubcert:
This encrypted directory contains the security certificates that are used commonly by Web browsers. These certificates are used to establish security credentials. Each appliance contains only one pubcert: directory. This directory is shared across domains.

sharedcert:
This encrypted directory contains security certificates that are shared with partners. Each appliance contains only one sharedcert: directory. This directory is shared across domains. However, you must be in `default` domain to create or upload keys and certificates.

store: This directory contains example style sheets, default style sheets, and schemas that are used by the local appliance. Do not modify the files in this directory.

Each appliance contains only one store: directory. By default, this directory is visible to all domains. You can make changes to the contents of this directory from the `default` domain only.

The store: directory has the following subdirectories:

meta This encrypted subdirectory contains files that are used by the appliance itself.

msgcat
This subdirectory contains the message catalogs.

policies
This subdirectory contains the following subdirectories. The contents of these subdirectories affect Web services policy.

custom
This subdirectory contains custom style sheets.

mappings
This subdirectory contains mapping style sheets.

templates
This subdirectory contains XML files.

profiles
This subdirectory contains style sheets that are used by DataPower services.

schemas
This subdirectory contains schemas that are used by DataPower services.

dp This encrypted subdirectory contains files that are used by the appliance itself. This subdirectory is available from the command line only.

pubcerts
This encrypted subdirectory contains files that are used by the appliance itself. This subdirectory is available from the command line only.

tasktemplates:
This directory contains the XSL files that define the display of specialized WebGUI screens. Each appliance contains only one tasktemplates: directory. This directory is visible to the `default` domain only.

temporary:
This directory is used as temporary disk space by processing rules. Each application domain contains one temporary: directory. This directory is not shared across domains.

Object name conventions

The name must be unique in this object namespace. The following characters in an object name are valid:
- `a` through `z`
- `A` through `Z`
- `0` through `9`
- `_` (underscore)
- `-` (dash)
- `.` (period)

Typeface conventions

The following typeface conventions are used in the documentation:

bold Identifies commands, programming keywords, and GUI controls.

italics Identifies words and phrases used for emphasis and user-supplied variables.

`monospaced`
Identifies user-supplied input or computer output.

Chapter 1. Initial login and common commands

This chapter provides an alphabetic listing of the commands that are available before entering a specific configuration mode (available at initial login) and commands that are available in most, if not all, configuration modes.

Initial login commands

For a list of the commands that are available after an initial login, refer to Table 1. This table provides a listing of the available commands and their purpose. To determine whether these commands are available to a specific user-type class after an initial login, refer to Table 2.

Table 1. Initial login commands and their general purpose

Command	Purpose
alias[1]	Creates a command macro.
clock[1]	Sets the date or time.
configure terminal	Enters Global configuration mode.
disable[1]	Enters User Mode.
disconnect	Closes a user session.
echo	Echoes text to the console.
enable	Enters Privileged mode.
exec	Calls and runs a target configuration script from another configuration script.
exit	Closes the CLI connection.
help	Displays online help.
login	Logs in to the appliance as a specific user.
ntp[1]	Identifies an NTP server.
ping	Determines if a target host is reachable on the network.
show	Displays configuration or status information
shutdown[2]	Restarts or shuts down the appliance.
switch domain	Moves to a specified domain.
template[1]	Runs an interactive command line script.
test schema[1]	Tests conformity of an XML file against a schema.
test tcp-connection[1]	Tests the TCP connection to a remote host.
top	Returns users to their initial log in mode.
traceroute[1]	Traces the network path to a target host.

[1] Also available in Global mode.

[2] Also available in Flash configuration mode.

Table 2. Commands by type of user that are available after initial login

Command	admin user	Privileged-type user	User-type user
alias	Yes	Yes	No

Table 2. Commands by type of user that are available after initial login (continued)

Command	admin user	Privileged-type user	User-type user
clock	Yes	Yes	No
configure terminal	Yes	Yes	No
disable	Yes	Yes	No
disconnect	Yes	Yes	No
echo	Yes	Yes	Yes
enable	No	No	Yes
exec	Yes	Yes	No
exit	Yes	Yes	Yes
help	Yes	Yes	Yes
login	Yes	Yes	No
ntp	Yes	Yes	No
ping	Yes	Yes	Yes
show	Yes	Yes	Yes
shutdown	Yes	Yes	No
switch	Yes	Yes	Yes
template	Yes	Yes	Yes
test schema	Yes	Yes	Yes
test tcp-connection	Yes	Yes	Yes
top	Yes	Yes	Yes
traceroute	Yes	Yes	Yes

Common commands

For a list of the commands that are available in most configuration modes, refer to Table 3. This table provides a listing of the available commands and their purpose.

Table 3. Common configuration commands and their general purpose

Command	Purpose
admin-state	Sets the administrative state of an object.
cancel	Cancels changes to the current object and returns to the parent configuration mode.
disconnect[1]	Closes a user session.
echo[1]	Echoes text to the console.
exit[1]	Applies changes to the current object and returns to the parent configuration mode.
help[1]	Displays online help.
ping[1]	Determines if a target host is reachable on the network.
reset	Restores default values.
show[1,2]	Displays configuration information
summary	Specifies a brief object-specific comment.
test tcp-connection[1]	Tests the TCP connection to a remote host.
traceroute[1]	Traces the network path to a target host.

Table 3. Common configuration commands and their general purpose (continued)

Command	Purpose
[1] The command is also available after initial log in, which is before you explicitly enter a configuration mode. To determine whether these commands are available to a specific user-type class after an initial login, refer to Table 2 on page 1.	
[2] The output from the command differs when invoked after initial log in and when invoked while in a configuration mode.	

admin-state

Sets the administrative state of an object.

Syntax

admin-state {**enabled** | **disabled**}

Parameters

enabled
(Default) Places an object in the enabled (active) state

disabled
Places an object in the disabled (inactive) state

Guidelines

The **admin-state** command sets the administrative state of an object. Administrative states are not equivalent to operational states. When an object has an administrative state of **enabled**, its operational state might be `up`, `down`, or `pending`. However, when an object has an administrative state of **disabled**, its operational state is always `down`.

Examples

- Disables the object.

```
# admin-state disable
#
```

alias

Creates a command macro.

Syntax

alias *alias command*

no alias *alias*

Parameters

alias Specifies the name of the object.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

command
Specifies a sequence of commands and arguments.

Guidelines

Also available in Global configuration mode.

If creating a macro that uses multiple commands, you can either

- Surround the string in quotes and separate commands with a semicolon. For example:

```
alias eth0 "configure terminal; interface ethernet 0"
```

- Separate commands with an escaped semicolon. For example:

```
alias eth0 configure terminal\;interface ethernet0
```

Use the **no alias** command to delete a command macro.

Related Commands

show alias

Examples

- Creates an alias `eth0`. When invoked, moves to Interface configuration mode (with the `configure terminal` and `interface` commands) for Ethernet Port 0.

```
# alias eth0 configure terminal\;interface eth0
Alias update successful
#
```

- Creates an alias `mgmport`. When invoked, moves to Interface configuration mode with the `configure terminal` and `interface` commands) for Management Port 0.

```
# alias "mgtport configure terminal; interface management 0"
Alias update successful
#
```

- Creates an alias `back2`. When invoked, moves back two configuration modes. If invoked from Validation Credentials configuration mode, moves to Global configuration mode.

```
# alias back2 "exit; exit"
Alias update successful
#
```

- Creates an alias `proxies`. When invoked, displays information about XSL Proxy objects.

```
# alias proxies show xslproxy
Alias update successful
#
```

- Creates an alias `update-cfg`. When invoked, restarts the appliance with an updated configuration script.

```
# alias update-cfg configure terminal\;flash\;del config:runningconfig.cfg\;
copy http://10.10.1.1/configs/39.3.cfg config:///runningconfig.cfg\;
boot config runningconfig.cfg\;shutdown
Alias update successful
#
```

- Deletes the `eth0` alias.

```
# no alias eth0
Alias 'eth0' deleted
#
```

cancel

Cancels changes to the current object and returns to the parent configuration mode.

Syntax

cancel

Guidelines

The **cancel** command cancels all configuration changes to the current object and returns to the parent configure mode. This command is available in all configuration modes except Interface configuration mode.

Related Commands

exit, **reset**

Examples

- Cancels the current configuration, which leaves the objects unchanged.

```
# cancel
#
```

---

clock

Sets the date or time.

Syntax

clock *yyyy-mm-dd*

clock *hh*:*mm*:*ss*

Parameters

yyyy-mm-dd
: Specifies the date in four-digit year, two-digit month, and two-digit day format. When setting the date, separate each value with a hyphen (-).

hh:*mm*:*ss*
: Specifies the time in two-digit hour, two-digit minute, and two-digit second format. When setting the time, separate each value with a colon (:).

Guidelines

Also available in Global configuration mode.

Related Commands

ntp, **show clock**

Examples

- Sets the date to August 8, 2007.

```
# clock 2007-08-08
Clock update successful
#
```

- Sets the time to 8:31 PM.

```
# clock 20:31:00
Clock update successful
#
```

configure terminal

Enters Global configuration mode.

Syntax

configure terminal

Guidelines

You use Global configuration mode to create system-wide resources that are available to various system service, to configure global behaviors, and to enter specialized configuration modes.

Related Commands

disable, **exit**

Examples

- Enters Global configuration mode.

```
# configure terminal
Global configuration mode
(config)#
```

diagnostics

Enters Diagnostics mode.

Syntax

diagnostics

Guidelines

The **diagnostics** command enters Diagnostics mode.

Attention: Use this command only at the explicit direction of IBM Support.

disable

Enters User Mode.

Syntax

disable

Guidelines

Also available in Global configuration mode.

Related Commands

enable, **exit**

Examples

- Exits privileged mode and enters User Mode.

```
# disable
Exiting privileged mode.
>
```

disconnect

Closes a user session.

Syntax

disconnect *session*

Parameters

session Specifies the session ID.

Guidelines

The **disconnect** command closes a user session. Use the **show users** command to display the list of active user sessions.

Related Commands

show users

Examples

- Closes the session that is associated with session ID 36..

```
# disconnnect 36
Session 36 closed.
#
```

echo

Echoes text to the console.

Syntax

echo *text*

Parameters

text Specifies the text to display.

enable

Enters Privileged mode.

Syntax

enable

Guidelines

After entering the **enable** command, the CLI prompts for a user name and password. Only authenticated users are allowed to enter Privileged Mode.

Use the **disable** command to exit Privileged Mode and enter User Mode.

Use the **exit** command to exit Privileged Mode and terminate the CLI connection.

Use Privileged Mode to provide initial access and to start and to shutdown the appliance.

Related Commands

disable, **exit**

Examples

- Exits User Mode and enters Privileged Mode.

```
> enable
Username: admin
Password: ********
#
```

---

exec

Calls and runs a target configuration script.

Syntax

exec *URL*

Parameters

URL Identifies the location of the configuration file.

- If the file resides on the appliance, this parameter takes the form *directory*:///*filename*, where:

  directory
  Identifies a local directory. Generally, the directory is one of the following keywords:
  – `config`
  – `local`

  filename
  Identifies the file in the directory.

- If the file is remote and the transport protocol is HTTP, HTTPS, SCP, or SFTP, this parameter takes one of the following forms:
  – `http://`*user*`:`*password*`@`*host*`/`*file*
  – `https://`*user*`:`*password*`@`*host*`/`*file*
  – `scp://`*user*`:`*password*`@`*host*`/`*file*
  – `sftp://`*user*`:`*password*`@`*host*`/`*file*

  The host name can be specified as an IP address or as a qualified host name when DNS services were previously enabled.

Guidelines

The **exec** command enables the *modularity* of configuration scripts. For example, you can include all service configuration commands in a script called `services.cfg` and all Multi-Protocol Gateway configuration commands in the `gateway.cfg` script.

A *main* configuration script can consist entirely of a series of **exec** commands.

Examples

- Executes the specified configuration scripts.

```
# configure terminal
# exec config:///housekeeping.cfg
# exec config:///interfaces.cfg
# exec config:///crypto.cfg
# exec config:///services.cfg
#
```

exit

Applies changes to the current object and returns to the parent configuration mode.

Syntax

exit

Guidelines

The **exit** command applies all changes made to the object to the running configuration. To save these changes to the startup configuration, use the **write mem** command.

When issued from User Mode or Privileged Mode, the **exit** command closes the CLI connection. In all other modes, the command returns to its parent mode. When issued from the top most parent, the command closes the CLI connection.

Related Commands

cancel, **disable**, **write mem** (Global)

Examples

- Closes the CLI connection from User or Privileged Mode.

  ```
  # exit
  ```

- Applies all changes made to the Crypto Validation Credentials object. Leaves this Crypto Validation Credentials configuration mode, and returns to Crypto configuration mode. Leaves Crypto configuration mode and returns to Global configuration mode. Persists the changes made to all object during this session to the startup configuration. Closes the CLI connection.

  ```
  (config crypto-val-credentials)# exit
  (config crypto)# exit
  (config)# write mem
  (config)# exit
  ```

help

Displays online help.

Syntax

help [*command*]

? [*command*]

Parameters

command
Specifies the command name.

Examples

- Displays a list of commands available in Privileged Mode.

  ```
  # help
  ```

- Displays help for the **shutdown** command.

  ```
  # help shutdown
  ```

- Displays help for the **shutdown** command.

  ```
  # ? shutdown
  ```

login

Logs in to the appliance as a specific user.

Syntax

login

Guidelines

After entering the **login** command, the CLI prompts for a username and password.

User accounts log in to User Mode, while `admin`, privileged accounts, and group-specific accounts log in to Privileged Mode.

After your initial log in, the CLI prompts you to change your password.

Related Commands

username

Examples

- Logs in as `support` (a privileged account).

  ```
  # login
  Username: support
  Password: ********
  #
  ```

- Logs in as `eugene` (a user account).

  ```
  # login
  Username: eugene
  Password: ********
  >
  ```

ntp

Identifies an NTP server.

Syntax

ntp *server* [*interval*]

no ntp

Parameters

server Specifies the IP address or host name.

interval
Specifies the number of seconds between synchronizations with the NTP server. The default is 900.

Guidelines

Also available in Global configuration mode.

Use the **ntp** command to identify the NTP (Network Time Protocol) server. After identifying an NTP server, the appliance functions as a Simple Network Time Protocol (SNTP) client as described in RFC 2030.

Note: From the CLI, the appliance supports the configuration of only one NTP server. Although the CLI supports only one NTP server, you can use the WebGUI to identify multiple NTP servers. When more than one NTP server is identified, the appliance contacts the first NTP server in the list. If this server does not respond, the appliance contacts the next server in the list. If you used the WebGUI to identify more than one NTP server, do not use the CLI to modify the NTP service. Using the **ntp** command replaces the entire list with the one identified NTP server.

Related Commands

clock, **show ntp-service**, **show ntp-refresh**, **time**

Examples

- Identifies 10.10.12.13 as the NTP server. Uses the default synchronization interval.

```
# ntp 10.10.12.13
Modifying NTP Service configuration
#
```

- Replaces 10.10.12.13 with 10.10.12.14 as an NTP server. Sets the synchronization interval to every 2 minutes.

```
# ntp 10.10.12.13 120
Modifying NTP Service configuration
#
```

- Deletes the configured NTP server.

```
# no ntp
Modifying NTP Service configuration
%     No NTP servers are configured
#
```

ping

Determines if a target host is reachable on the network.

Syntax

ping *host*

Parameters

host Specifies the target host. Use either the IP address or host name.

Guidelines

The **ping** command sends 6 Internet Control Message Protocol (ICMP) echo-request messages to the specified host with a one second interval between each message and reports the results.

Related Commands

ip host, **ip name-server**, **test tcp-connection**, **traceroute**

Examples

- Pings ragnarok.

```
# ping ragnarok
```

- Pings 192.168.77.144.

```
# ping 192.168.77.144
```

reset

Restores default values.

Syntax

reset

Guidelines

The **reset** command sets mode-specific properties to their default values. Properties that lack default values, are unchanged.

Default values assigned by the **reset** command are not applied until the user uses the **exit** command to save changes and exit the current configuration mode.

Related Commands

cancel, **exit**

Examples

- Restores default values for the object and returns to Global configuration model.

```
# reset
# exit
#
```

show

Displays configuration or status information

Syntax

show [*arguments*]

Parameters

arguments
Specifies the specific configuration object or status object.

Guidelines

The **show** command displays configuration information or status information that is relevant to the provided argument. In the absence of an argument, the result differs depending on where you invoked the command.

- Within the initial login, displays a list of available arguments.
- Within a configuration mode, list the currently configured properties of that object.

For information about using the various **show** command, refer to Chapter 129, "Monitoring commands," on page 1053.

shutdown

Restarts or shuts down the appliance.

Syntax

shutdown reboot [*seconds*]

shutdown reload [*seconds*]

shutdown halt [*seconds*]

Parameters

reboot Shuts down and restarts the appliance.

reload Restarts the appliance.

halt Shuts down the appliance.

seconds
Specifies the number of seconds before the appliance starts the shutdown operation. Use an integer in the range of 0 through 65535. The default is 10.

Guidelines

Also available in Flash configuration mode.

The appliance restarts using the startup configuration specified by the **boot config** command and the startup firmware image specified by the **boot image** command. If a startup configuration or firmware image has not been designated, the appliance restarts with the configuration and firmware image that were active when the **shutdown** command was executed.

Related Commands

boot config, boot image

Examples

- Shuts down and restarts the appliance after 10 seconds.

```
# shutdown reboot
Reboot in 10 second(s).
#
```

- Restarts the appliance after 20 seconds.

```
# shutdown reload 20
Reload in 20 second(s).
#
```

- Shuts down the appliance after 60 seconds.

```
# shutdown halt 60
Shutdown in 60 second(s).
#
```

summary

Specifies a brief, object-specific comment.

Syntax

summary *string*

Parameters

string Specifies descriptive text for the object.

Guidelines

The **summary** command specifies a brief, object-specific comment. If the comment contains spaces, enclose the comment in double quotation marks.

Examples

- Adds an object-specific comment.

```
# summary "Amended server list"
```

---

switch domain

Moves to a specified domain.

Syntax

switch domain [*domain*]

Parameters

domain Specifies the name of the target domain.

Guidelines

In the absence of a specified target domain, the command prompts for the domain name.

Related Commands

domain

Examples

- Switches from the default domain to the `application-1` domain.

```
(config)# switch domain application-1
[application-1](config)#
```

- Displays the list of available domains and switches from the `application-1` domain to the `default` domain.

```
[application-1](config)# switch domain
Domain (? for all): ?
application-1
default
Domain (? for all): default
(config)#
```

---

template

Runs an interactive command line script.

Syntax

template *URL*

Parameters

URL Specifies the fully-qualified location of the interactive command line script.

Guidelines

Also available in Global configuration mode.

The **template** command specifies the URL of the interactive command line script. The script is an XML file that can be local or remote to the DataPower appliance. The script must conform to the store:///schemas/dp-cli-template.xsd schema.

To verify whether the script is conformant with the schema, use the **test schema** command.

Related Commands

test schema

Examples

- Verify that `local:///shell-script.xml` conforms to the `store:///schemas/dp-cli-template.xsd` schema.

```
# test local:///shell-script.xml store:///schemas/dp-cli-template.xsd
#
```

- Runs the interactive script as defined in the `local:///shell-script.xml` file.

```
# template local:///shell-script.xml
#
```

---

test schema

Tests conformity of an XML file against a schema.

Syntax

test schema *file schema*

Parameters

file Specifies the URL of the XML file to test.

schema Specifies the URL of the schema.

Guidelines

Also available in Global configuration mode.

The **test schema** command tests the conformity of an XML file against an XSD schema file.

Examples

- Tests conformity of the `xyzbanner.xml` XML file against the `dp-user-interface.xsd` schema.

```
# test schema store:///xyzbanner.xml store:///schemas/dp-user-interface.xsd
Performing validation of document 'store:///xyzbanner.xml' using
  schema 'store:///schemas/dp-user-interface.xsd' ...
Document validation completed: OK.
#
```

test tcp-connection

Tests the TCP connection to a remote appliance.

Syntax

test tcp-connection *host port* [*timeout*]

Parameters

host Specifies the target host. Use either the IP address or host name.

port Specifies the target port.

timeout
Specifies an optional timeout value, the number of seconds that the CLI waits for a response from the target host. The default is 10.

Guidelines

Also available in Global configuration mode.

Related Commands

ip host, **ip name-server**, **ping**, **traceroute**

Examples

- Confirms an available TCP connection to the specified host on port number 80 (the well-known HTTP port), using the default timeout value (10 seconds).

```
# test tcp-connection ragnarok 80
TCP connection successful
#
```

- Confirms an available TCP connection to the specified IP address on port 21 (the well-known FTP control port). The timeout value is 5 seconds.

```
# test tcp-connection 192.168.77.27 21 5
TCP connection successful
#
```

top

Returns users to their initial log in mode.

Syntax

top

Guidelines

Regardless of the current location in the configuration modes, the **top** command immediately returns you to your original login mode.

For custom accounts, **top** returns to the user-group-specific login mode.

Related Commands

usergroup

Examples

- Returns the user, either the `admin` account or a `privileged` account, to Privileged Mode, the user-specific login mode.

```
(config crypto-val-credentials)# top
#
```

traceroute

Traces the network path to a target host.

Syntax

traceroute *host*

Parameters

host Specifies the target host as either the IP address or host name.

Guidelines

Also available in Global configuration mode.

Related Commands

ip host, **ip name-server**, **ping**, **test tcp-connection**

Examples

- Confirms an available TCP connection to `loki` .

```
# traceroute loki
```

Chapter 2. Global configuration mode

You use Global configuration mode to create system-wide resources that are available to various system services, to configure global behaviors, and to enter specialized configuration modes.

This chapter provides an alphabetic listing of commands that are available in Global configuration mode. Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Global configuration mode.

aaapolicy

Enters AAA Policy configuration mode.

Syntax

aaapolicy *name*

no aaapolicy *name*

Parameters

name Specifies the name of the object.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **aaapolicy** command enters AAA (Authentication, Authorization, Audit) configuration mode where you can create or modify an AAA Policy.

Use the **no aaapolicy** command to delete an AAA Policy.

Use the **cancel** or **exit** commands to exit AAA Policy configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

account (Common Criteria)

Defines the lockout behavior for local accounts.

Syntax

account max-login-failure *count*

account lockout-duration *minutes*

Parameters

lockout-duration *minutes*
Specifies the number of minutes to lock out an account after exceeding the maximum number of failed login attempts. A value of 0 indicates that accounts are locked out until reset by a privileged administrator. Use an integer in the range of 0 through 1000. The default is 1.

max-login-failure *count*
Specifies the maximum number of failed login attempts to allow before lockout. A value of 0 disables account lockout. Use an integer in the range of 0 through 64. The default is 3.

Context

Available only when the appliance is in Common Criteria mode.

Guidelines

The **account** command defines whether to lock out a local user account after a specific number of failed login attempts and, if lockout is enabled, the duration to lock out the local account. To enable lockout behavior and define the duration to lock out the account requires two invocations of the **account** command.

- An invocation with the **max-login failure** parameter defines the number of failed login attempts to permit before a successful login. If the value is 3 and the user has failed three consecutive login attempts, the behavior on the next login attempt for this user is as follows:
  - If failure, the account is locked out. The duration of the lockout depends on the value defined by the **lockout-duration** parameter.
  - If successful, the account is not locked out and the count is reset.

  If the value is 0, lockout behavior is disabled. Repeated successive login failures by a user do not cause lockout of that account.
- An invocation with the **lockout-duration** parameter defines the duration to lock out an account after exceeding the permitted number of failed login attempts defined by the invocation with the **max-login failure** command. Instead of locking out an account for a specific duration, the account can be locked out until re-enabled by a privileged administrator. To lock out accounts until reset, set the duration to 0.

  When lockout behavior is enabled and an account is locked out, a privileged administrator can use the Global **reset username** command to re-enable the account. To re-enabled the account

  1. The administrator will change the password on the account with the **reset username** command.
  2. The user will be prompted to again change the password on initial login.

Note: The **account** command applies to all accounts including the `admin` account. The only difference is that the `admin` account cannot be locked out until reset. When the duration is 0, the `admin` account is locked out for 120 minutes or until re-enabled by another administrator.

Related Commands

reset username

Examples

- Enables lockout behavior for accounts that on the fifth login failure, the account is locked out locked out until reset by a privileged administrator:

```
# account lockout-duration 0
# account max-login-failure 4
```

- Disables lockout behavior.

```
# account max-login failure 0
```

---

acl

Enters Access Control List configuration mode for a specified service provider.

Syntax

acl *name*

acl ssh

acl web-mgmt

acl xml-mgmt

no acl *name*

Parameters

name Specifies the name of an object-specific or standalone ACL.

- Can be the name of the service provider (for example, the name of a DataPower service or the name of a CLI Telnet service) in which case the enters Access Control List configuration mode to create an object-specific ACL.
- Can be the name of a standalone ACL, which can later be assigned to a service provider, or to any of the Protocol Handler types.

  The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

ssh Identifies the SSH service. In this case, the command enters ACL configuration mode to create an SSH-specific ACL.

web-mgmt
Identifies the WebGUI Management Interface. In this case, the command enters ACL configuration mode to create a WebGUI Management Interface-specific ACL.

xml-mgmt
Identifies the XML Management Interface. In this case, the command enters ACL configuration mode to create an XML Management Interface-specific ACL.

Guidelines

While in Access Control List configuration mode, you can configure an ACL for a specific service provider or for later assignment to a service provider.

An ACL contains one or more clauses. Each clause consists of an IP address range that is defined by an IP address and net mask and a Boolean value (`ALLOW` or `DENY`). IP addresses are evaluated against each clause in the order in which they are in the

list. A candidate address is denied or granted access to the service provider in accordance with the first matching clause. Consequently, the order of clauses is important in an Access Control List.

Use the **no acl** command to delete a named ACL.

Use the **exit** command to exit Access Control list configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**, **ssh**, **xml-mgmt**

Examples

- Enters Access Control list configuration mode to create the ACL-1 standalone ACL.

```
# acl ACL-1
ACL configuration mode
#
```

- Deletes the standalone ACL-1 ACL.

```
# no acl ACL-1
#
```

- Enters ACL configuration mode for the SSH service.

```
# acl XSLProxy-1
# acl ssh
ACL configuration mode
#
```

- Enters ACL configuration mode for the XML Management Interface.

```
# acl xml-mgmt
ACL configuration mode
#
```

---

action

Enters Processing Action configuration mode.

Syntax

action *name*

no action *name*

Parameters

name Specifies the name of the Processing Action.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no action** command to delete a Processing Action.

Use the **cancel** or **exit** commands to exit Processing Action configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**, **show action**

---

alias

Creates a command macro.

Syntax

alias *aliasName commandString*

no alias *aliasName*

Parameters

aliasName
Specifies the name of the command macro.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

commandString
Defines a sequence of commands.

Guidelines

If creating a macro that uses multiple commands, you can either

- Use quotation marks ("") to surround the command string, and use a semicolon (;) after each commend

  For example:

  ```
  alias eth0 "configure terminal; interface ethernet 0"
  ```

- Separate commands with an escaped semicolon (\;)

  For example:

  ```
  alias eth0 configure terminal\;interface ethernet0
  ```

Use the **no alias** command to delete a command macro.

Also available in Privileged mode.

Related Commands

show alias

Examples

- Creates the `eth0` alias that moves to Interface configuration mode (with the **interface** command) for Ethernet Port 0.

  ```
  # alias eth0 interface eth0
  Alias update successful
  #
  ```

- Creates the `mgmport` alias that moves to Interface configuration mode (with the **interface** command) for Management Port 0.

  ```
  # alias mgtport interface management 0
  Alias update successful
  #
  ```

- Creates the back2 alias that moves back two configuration modes. If invoked while in Validation Credentials configuration mode, moves to Global configuration mode.

```
# alias back2 "exit; exit"
Alias update successful
#
```

- Creates the proxys alias that displays information about XSL Proxy objects.

```
# alias proxys show xslproxy
Alias update successful
#
```

- Creates the update-cfg alias that restarts the appliance with an updated configuration script.

```
# alias update-cfg flash\;
del config:runningconfig.cfg\;copy http://10.10.1.1/configs/39.3.cfg
 config:runningconfig.cfg\; boot config runningconfig.cfg\;
 shutdown
Alias update successful
#
```

- Deletes the eth0 alias.

```
# no alias eth0
Alias 'eth0' deleted
#
```

---

application-security-policy

Enters Application Security Policy configuration mode.

Syntax

application-security-policy *name*

no application-security-policy *name*

Parameters

name Specifies the name of the Application Security Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

The **application-security-policy** command enters Application Security Policy configuration mode to create a named Application Security Policy. A Web Application Firewall can use this Application Security Policy.

Use the **no application-security-policy** command to delete an Application Security Policy.

Use the **cancel** or **exit** commands to exit Application Security Policy configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

audit delete-backup (Common Criteria)

Deletes the archived version of the audit log.

Syntax

audit delete-backup

Context

Available only when the appliance is in Common Criteria mode.

Guidelines

The **audit delete-backup** command deletes the audit:///audit-log.1 file. This file is the archived version of the audit log and is created when the log reaches When the size of the audit log, the audit:///audit-log file, reaches approximately 250 kilobytes, the appliance save this file as the audit:///audit-log.1 file, which overwrites the previous version of the audit:///audit-log.1 file.

After invoking the command, the interface prompts for confirmation.

audit level (Common Criteria)

Sets the audit level of the firmware.

Syntax

audit level {**full** | **standard**}

Parameters

full (Default) Audits the standard set of events and decisions on information flow.

standard
Audits the standard set of events only. Does not audit decisions on information flow.

Context

Available only when the appliance is in Common Criteria mode.

Guidelines

The **audit level** command sets the audit level of the firmware.

- When full auditing is not strictly required, set the level to **standard**.
- When corporate or business security policies require full auditing, set the level to **full**. This audit level impacts performance.

audit reserve (Common Criteria)

Reserves disk space for the audit log.

Syntax

audit reserve *kilobytes*

Parameters

kilobytes
Specifies the amount of disk space in kilobytes to reserve for the audit log. The reserve space must be at least four kilobytes less than the total amount of free space that is currently available on the file system. Use an integer in the range of 0 through 10000. The default is 40.

Context

Available only when the appliance is in Common Criteria mode.

Available only to privileged users in the `default` domain.

Guidelines

The **audit reserve** command specifies the amount of disk space in kilobytes to reserve for the audit log. Use this command to alter the amount of disk space to reserve to prevent the loss of audit events in case of a full disk. This function is disabled if the value is 0.

If the appliance is forced to release the audit reserve:

- All data services will be forced into an operational `down` state and cease to process traffic.
- All administrative services, such as the WebGUI, Telnet, and so forth, will continue to work.

When the appliance forces the release, the log will contain a message that states that the disk space for audit events is low.

Before restoring the appliance to service, a privileged administrator needs to free up disk space. When there is enough available disk space for normal operations, the administration can restart the appliance, which will resume the processing of traffic.

---

cache schema

Loads a compiled schema to the schema cache of a specific XML Manager.

Syntax

cache schema *xmlMgrName schemaURL* [*compilationMode*]

Parameters

xmlMgrName
Specifies the name of an XML manager.

schemaURL
Specifies the URL of the schema that the specific XML Manager caches.

compilationMode
Optionally specifies the schema compilation mode. Use one of the following values:

general
(Default) Performs standard schema compilation

stream
Compiles the schema in streaming mode

If in doubt about whether the target schema lends itself to streaming, retain the default value of **general**.

Related Commands

cache stylesheet, **cache wsdl**

Examples

- Compiles the schema in streaming mode and adds the schema to the schema cache that is maintained by the `mgr1` XML Manager.

```
# cache schema mgr1
 http://www.datapower.com/XSD/partnerProfile-1.xsd stream
#
```

---

cache stylesheet

Loads style sheets to the stylesheet cache for a specific XML manager.

Syntax

cache stylesheet *XML-manager match*

Parameters

XML-manager
Specifies the name of an XML Manager.

match Specifies a shell-style match pattern that selects the URLs of the style sheets to cache.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Related Commands

cache schema, **cache size**, **cache wsdl**, **cache xsl**, **clear xsl**

Examples

- Caches style sheets located at `http://www.datapower.com/XSL/` in the stylesheet cache of the `mgr1` XML manager.

```
# cache stylesheet mgr1
 http://www.datapower.com/XSL/*.xsl
#
```

---

cache wsdl

Loads a compiled WSDL to the WSDL cache of a specific XML Manager.

Syntax

cache wsdl *xmlMgrName wsdlURL*

Parameters

xmlMgrName
Specifies the name of an XML manager.

wsdlURL
Specifies a URL of the schema to cache.

Related Commands

cache schema, **cache stylesheet**

Examples

- Compile and adds the specified WSDL to the WSDL cache of the `mgr1` XML Manager.

```
# cache wsdl mgr1
 http://www.datapower.com/WSDL/quoteNYSE.wsdl
#
```

clear aaa cache

Clears the information caches of a specific AAA Policy.

Syntax

clear aaa *aaaPolicyName*

Parameters

aaaPolicyName
Specifies the name of the AAA Policy.

Guidelines

The **clear aaa cache** command clears both the authentication and authorization information caches of the specified AAA Policy.

Related Commands

cache allow, **cache ttl**

Examples

- Clears the authentication and authorization caches of the `Policy-1` AAA Policy.

```
# clear aaa Policy-1
#
```

clear arp

Clears the ARP table.

Syntax

clear arp

Guidelines

Also available in Interface configuration mode.

Related Commands

arp, **show netarp**

Examples

- Clears the ARP table.

```
# clear arp
#
```

clear dns-cache

Clears the DNS cache.

Syntax

clear dns-cache

Examples

- Clears the DNS cache.

```
# clear dns-cache
Cleared DNS cache
#
```

clear pdp cache

Clears all compiled XACML policies of a specific XACML Policy Decision Point (PDP).

Syntax

clear pdp cache *pdpName*

Parameters

pdpName
Specifies the name of the XACML PDP.

Related Commands

cache-ttl (XACML Policy Decision Point), **clear xsl cache**, **urlrefresh**

Guidelines

In addition to using the **clear pdp cache** command to explicitly clear the PDP-specific XACML policy cache, you can use the following WebGUI properties to control XACML policy cache.

Specify the TTL for the PDP
During PDP configuration, use the **cache-ttl** command to specify a cache lifetime.

Use the XML Manager
When the PDP is for authorization, users can access the XML Manager that

is associated with the AAA Policy with the **clear xsl cache** command. This command clears the compiled XACML policies in the XML Manager that is referenced by the AAA Policy.

Use a URL Refresh Policy

You can use a URL Refresh Policy whose match conditions match the internal URL `xacmlpolicy:///`*pdpName* to perform periodic cache refreshes.

- When PDP TTL is 0, the URL Refresh Policy controls cache refresh.
- When the URL Refresh Policy is the **no-cache** type, XACML policies are never cached.
- When the URL Refresh Policy is the **protocol-specified** type, the TTL of the PDP governs cache refresh unless its value is 0.
- When the URL Refresh Policy is the **default** type with a refresh interval setting, the TTL of the PDP is ignored, and the URL Refresh Policy refresh interval governs cache refresh.
- When the URL Refresh Policy is the **no-flush** type with a refresh interval setting, the greater of the URL Refresh Policy refresh interval or the TTL of the PDP governs cache refresh.

Examples

- Clears the XACML policy cache of the `PDP-orderEntry` PDP.

```
# clear pdp cache PDP-orderEntry
Cleared cache of PDP PDP-orderEntry
#
```

clear rbm cache

Clears all cached role-based management (RBM) authentication data.

Syntax

clear rbm cache

Examples

- Clears cached RBM authentication data.

```
# clear rbm cache
Cleared RBM cache
#
```

clear xsl cache

Clears the stylesheet cache of a specific XML Manager.

Syntax

clear xsl cache *XML-manager*

Parameters

XML-manager

Specifies the name of an XML Manager.

Related Commands

cache stylesheet xsl, **cache size**

Examples

- Clears the stylesheet cache of the mgr1 XML Manager.

```
# clear xsl cache mgr1
Cleared cache of xmlmgr mgr1
#
```

---

cli remote open

Establishes a TCP/IP connection to a specific remote host.

Syntax

cli remote open *address port*

Parameters

address Specifies the IP address of the remote host.

port Identifies the port on the remote host that monitors CLI traffic. Use an integer in the range of 0 through 65535.

Guidelines

The **cli remote open** command establishes a TCP/IP session between the appliance and a remote site, but only at explicit initiation of an the admin or a privileged user. This command does not provide a back door to the appliance.

This command provides a command shell to a remote host that allows offsite technicians to access a appliance that is protected by a firewall or other security measures.

This command provides the same function as the **cli telnet** command, but provides the function from a remote host.

Related Commands

cli telnet

Examples

- Establishes an appliance-initiated TCP/IP connection between the DataPower appliance and the remote host (192.168.32.101:64999) and provides the remote host with a command shell.

```
# cli remote open 192.168.32.101 64999
#
```

---

cli telnet

Enters Telnet Service configuration mode, or creates a Telnet service for client-initiated access to the command line.

Syntax

cli telnet *name*

cli telnet *name* [**0** | *telnetServerIP*] *telnetServerPort* [*telnetClientIP clientMask*]

no cli telnet *name*

Parameters

name Specifies the name of the Telnet service.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

telnetServerIP
Specifies the IP address (either primary or secondary) of a DataPower Ethernet interface. In conjunction with the port, identifies the IP address and port that the Telnet service monitors.

0 Indicates a wildcard that specifies all DataPower IP addresses.

telnetServerPort
Identifies a port on one or all IP interfaces. Use an integer in the range of 0 through 65535. In conjunction with IP address of the server, identifies the appliance IP addresses and port that the Telnet service monitors.

telnetClientIP
Optionally identifies the IP address. In conjunction with the client mask, defines a contiguous range of IP addresses that are granted client access to the Telnet service.

clientMask
Identifies the network portion of the client IP address. The client mask can be expressed in CIDR (slash) format or in dotted decimal format.

Guidelines

Without the *telnetClientIP* and *clientMask* arguments, client access to the Telnet service is unrestricted. To restrict access to a noncontiguous IP address range, compile an ACL with the **acl**, **allow**, and **deny** commands.

Note: Telnet is an unsecure protocol and should be used with extreme caution. Telnet should be enabled only on the trusted management port or on a secure network segment.

Use the **no cli telnet** command to delete a Telnet service.

Related Commands

acl, **allow**, **deny**

Examples

- Enters Telnet Service configuration mode to create the `telnet-1` service.

```
# cli telnet telnet-1
Telnet Service configuration mode
#
```

- Creates the `support` Telnet service on 192.168.14.12:23. Access is restricted to the single specified Telnet client (10.10.10.5).

```
# cli telnet support 192.168.14.12 23 10.10.0.5 255.255.255.255
Installed cli telnet handler
#
```

- Creates the `public` Telnet service on Ethernet 192.168.14.12:23. Access is restricted to a range of addresses (10.10.8.0 through 10.10.11.255).

```
# cli telnet public 192.168.14.12 23 10.10.8.0/22
Installed cli telnet handler
#
```

- Deletes the `support` Telnet service.

```
# no cli telnet support
Deleted cli telnet handler
#
```

compact-flash (Type 9235)

Enters Compact Flash configuration mode.

Syntax

compact-flash *name*

Parameters

name Specifies the name of the existing compact flash volume. For appliances that have a compact flash for auxiliary data storage, the name is `cf0`.

Guidelines

The **compact-flash** command enters Compact Flash configuration mode for an existing compact flash enabled appliance. For appliances that have a compact flash for auxiliary data storage, the name is `cf0`.

Examples

- Enters Compact Flash configuration mode for volume `cf0`.

```
# compact-flash cf0
Compact Flash configuration mode
#
```

compact-flash-initialize-filesystem (Type 9235)

Initializes the file system.

Syntax

compact-flash-initialize-filesystem *name*

Parameters

name Specifies the name of the existing compact flash volume. For appliances that have a compact flash for auxiliary data storage, the name is `cf0`.

Guidelines

The **compact-flash-initialize-filesystem** command initializes the file system on the compact flash to allow it to be made active. This action destroys the existing contents of the compact flash storage card.

Examples

- Makes a new file system on the `cf0` compact flash volume.

```
# compact-flash-initialize-filesystem cf0
```

compact-flash-repair-filesystem (Type 9235)

Repairs the file system.

Syntax

compact-flash-repair-filesystem *name*

Parameters

name Specifies the name of the existing compact flash volume. For appliances that have a compact flash for auxiliary data storage, the name is `cf0`.

Guidelines

The **compact-flash-repair-filesystem** command repairs the file system on the compact flash storage card, in case it was corrupted by an abnormal shutdown of the appliance or other error.

Examples

- Repairs the file system on the `cf0` compact flash volume.

```
# compact-flash-repair-filesystem cf0
```

---

compile-options

Enters Compile Options Policy configuration mode.

Syntax

compile-options *name*

no compile-options *name*

Parameters

name Specifies the name of the Compile Options Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Profiling results are available with the **show profile** command, from the WebGUI (**Status** → **Stylesheet Profiles**), or from the XML Management Interface.

Note: After a style sheet is compiled with profiling enabled, it must be flushed from the cache to disable profiling.

Use the **no compile-options** command to delete a Compile Options Policy.

Use the **cancel** or **exit** command to exit Compile Options Policy configuration mode and return to Global configuration mode.

Refer to Appendix D, "Compile Options Policy configuration," on page 1115 for details about creating a Compile Option Policy.

Related Commands

cancel, **exit**, **show profile**, **xslconfig**

conformancepolicy

Enters Conformance Policy configuration mode.

Syntax

conformancepolicy *name*

no conformancepolicy *name*

Parameters

name Specifies the name of the Conformance Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **conformancepolicy** command to enter Conformance Policy configuration mode to create or edit a Conformance Policy. A Conformance Policy is used by a conformance filter or a conformance transform.

- For a conformance filter, define a filter action that uses the `store:///conformance-filter.xsl` style sheet and specifies the named Conformance Policy.
- For a conformance filter, define a transform (`xform`) action that uses the `store:///conformance-xform.xsl` style sheet and specifies the named Conformance Policy.

A Conformance Policy supports the following profiles:

- Web Services Interoperability (WS-I) Basic Profile, version 1.0. The documentation is available at the http://www.ws-i.org/Profiles/BasicProfile-1.0.html site.
- WS-I Basic Profile, version 1.1. The documentation is available at the http://www.ws-i.org/Profiles/BasicProfile-1.1.html site
- WS-I Attachments Profile, version 1.0. The documentation is available at the http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html site.
- WS-I Basic Security Profile, version 1.0. The documentation is available at the http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html site.

Use the **no conformancepolicy** command to delete a Conformance Policy.

Use the **cancel** or **exit** command to exit Conformance Policy configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

copy

Copies a file to or from the DataPower appliance.

Syntax

copy [-f] *source destination*

Parameters

-f Overwrites an existing file, if one of the same name already exists. In the absence of this argument, an attempt to save a file with the same name as an existing file will result in a prompt that requests confirmation to overwrite the existing file.

source and *destination*
Specifies the URLs that identify the source file and target destination, respectively.

- If the source file or target destination reside on the appliance, these arguments take the following form:

  ```
  directory:///filename
  ```

  directory
  Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

  filename
  Specifies the name of a file in the specified directory.

- If the source file or target destination is remote to the DataPower appliance and the transport protocol is SCP or SFTP, these arguments take the form that is compliant with RFC 1738.

  To use an absolute path:

  ```
  scp://user@host:port//file_path
  sftp://user@host:port//file_path
  ```

  To use a path that is relative to the user's home directory:

  ```
  scp://user@host:port/file_path
  sftp://user@host:port/file_path
  ```

  Where:

  host Specifies the fully-qualified host name or IP address of the remote server. If DNS is enabled, the host name.

  port Specifies the listening port on the remote server.

  After issuing the command, the system prompts you for the remote login password.

Guidelines

The **copy** command transfers files to or from the DataPower appliance. You must issue this command from the appliance. When the source file or target destination is remote to the appliance, this command supports only the following protocols:

- HTTP
- HTTPS
- Secure Copy (SCP)
- Secured File Transfer Protocol

To send a file from the appliance as an email, use the Global **send file** command.

When using the **copy** command, be aware of the following restrictions:

- You cannot copy files from the cert: directory
- You cannot copy files to the audit:, logstore:, or logtemp: directory.

Related Commands

delete, **dir**, **move**, **send file** (Global)

Examples

- Uses HTTP to copy a file from the specified URL to the `image:` directory.

```
# copy http://host/image.crypt image:///image.crypt
file copy successful (1534897 bytes transferred)
#
```

- Uses HTTP over SSL to copy a file from the specified URL to the `image:` directory.

```
# copy https://host/image.crypt image:///image.crypt
file copy successful (1534897 bytes transferred)
#
```

- Uses SCP to copy a file from the specified URL to the `store:` directory.

```
# copy scp://jrb@10.10.1.159//XML/stylesheets/InitialConvert.xsl
store:///InitialConvert.xsl
     Password: yetanotherpassword
file copy successful
#
```

- Uses SCP to copy a file from the `logstore:` directory to the specified remote target (identified by a qualified host name).

```
# copy logstore:///Week1.log scp://jrb@ragnarok.datapower.com//LOGS/Week1.log
     Password: yetanotherpassword
file copy successful
#
```

- Uses SFTP to copy a file from the specified URL to the `store:` directory.

```
# copy sftp://jrb@10.10.1.159//XML/stylesheets/InitalConvert.xsl
 store:///InitalConvert.xsl
     Password: yetanotherpassword
file copy successful
#
```

- Uses SFTP to copy a file from the `logstore:` directory to the specified remote target.

```
# copy logstore:///Week1.log sftp://jrb@10.10.1.159//LOGS/x/Week1.log
     Password: yetanotherpassword
file copy successful
#
```

- Copies a file from the `config:` directory to the `local:` directory.

```
# copy config:///startup-config local:///startup-config
file copy successful (2347 bytes transferred)
#
```

---

create-tam-files

Creates TAM configuration files.

Syntax

create-tam-files [*create-copy*] *file admin password tam-domain application host port ssl-key-expiry ssl-timeout ldap-host ldap-port* [*ldap-password*] *ldap-auth-timeout ldap-search-timeout* [*use-ldap-cache*] [*ldap-user-cache-size*] [*ldap-policy-cache-size*]

Parameters

create-copy
The Tivoli® Access Manager key database and key stash files are placed in the cert: directory when created. This directory does not allow files to be moved out of it.

By selecting to create copies of the created files, a copy of the key database and stash files will be placed in the temporary: directory, and can be downloaded off of the appliance.
on Places copies in the temporary: directory.
off (Default) Does not place copies in the temporary: directory.

file Specifies the name to use for the created files. Do not provide a file extension. By default, the configuration files are stored in the local: directory and have the .conf extension. In addition to the configuration files, this file name is the base file name for the TAM key file (.kdb extension) and TAM stash files (.sth extension). The key file and stash file are stored in the cert: directory.

admin Specifies the user name of the TAM administrator. The default is `sec_master`.

password
Specifies the password for the TAM administrator.

tam-domain
Specifies the name of the TAM domain. The specified domain is the TAM domain to which the TAM client authenticate and use at runtime. The default is `Default`.

application
Specifies the name of the TAM application. The specified name is combined with the host name of the appliance to create a unique identifier for objects that are created for the TAM client.

host Specifies the host name or IP address of the TAM policy server.

port Specifies the port on which the TAM policy server listens for requests. The default is 7135.

ssl-key-expiry
Specifies the duration, in days, for which the SSL key file for the TAM client is valid. When the key expires, a new key must be generated for the TAM client. Valid range is 1 through 7200. The default is 183.

ssl-timeout
Specifies the wait period, in seconds, that the TAM client waits for a response to an SSL request from the TAM policy server. Valid range is 1 through 30. The default is 30.

ldap-host
Specifies the host name of the LDAP server that is the user registry for the TAM environment.

ldap-port
Specifies the port on which the LDAP server listens for requests. The default is 389.

ldap-password
Specifies the password for the distinguished name (DN) used to sign on (bind) to the LDAP server.

ldap-auth-timeout
Specifies the timeout, in seconds, that is allowed for LDAP authentication operations. There is no range limit. The default is 30.

ldap-search-timeout
Specifies the timeout, in seconds, that is allowed for LDAP search operations. There is no range limit. The default is 30.

use-ldap-cache
Indicates whether to enable client-side caching. Enabling client-side caching can improve performance for similar LDAP queries.
on Enables client-side caching.
off (Default) Disables client-side caching.

ldap-user-cache-size
When client-side caching is enabled, specifies the number of entries in the LDAP user cache. The default is 256.

ldap-policy-cache-size
When client-side caching is enabled, specifies the number of entries in the LDAP policy cache. The default is 20.

Guidelines

Use the **create-tam-files** command to create the configuration files needed to create a TAM object. The configuration files specify the network and security configuration for the policy server, replica authorization servers, and the LDAP (directory) server.

This command creates the following files:

- Client configuration file
- Key database file
- Key stash file
- Client obfuscation file (TAM version 5.1 and above)

The created files are named using the output file parameter. If TAM files are created with `app1` as the output file name parameter, the created files are `app1.conf`, `app1.kdb`, `app1.sth`, and `appl.conf.obf` (Tivoli Access Manager version 5.1 and above).

The configuration and obfuscation files are written to the local: directory, and the key database and stash files are written to the cert: directory.

Related Commands

cancel, **exit**, **tam**

---

crypto

Enters Crypto configuration mode.

Syntax

crypto

Guidelines

Use the **exit** command to exit Crypto configuration mode and return to Global configuration mode.

Related Commands

exit

delete

Deletes a file from the DataPower appliance.

Syntax

delete *URL*

Parameters

URL

Specifies a URL of the file to delete. This argument take the *directory*:///*filename* form, where:

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

filename
Specifies the name of a file in the specified directory.

Guidelines

The **delete** command deletes a file on the DataPower appliance. The deletion of a file is permanent. After a file is deleted, it cannot be recovered.

Note: The **delete** command does not prompt for confirmation. Be certain that you want to delete the file before issuing this command.

Related Commands

copy, **dir**, **move**

Examples

- Deletes the `startup-config-deprecated` file from the `store:` directory.

```
# delete store:\\\startup-config-deprecated
#
```

- Deletes the `betaImage` file from the `image:` directory.

```
# delete image:\\\betaImage
#
```

deployment-policy

Enters Deployment Policy configuration mode.

Syntax

deployment-policy *name*

no deployment-policy *name*

Parameters

name Specifies the name of the Deployment Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **deployment-policy** command to enter Deployment Policy configuration mode to create or edit a Deployment Policy.

Use the **cancel** or **exit** command to exit Deployment Policy configuration mode and return to Global configuration mode.

Use the **no deployment-policy** command to delete a Deployment Policy.

Related Commands

cancel, **exit**

---

dir

Displays the contents of a directory.

Syntax

dir *directory*

Parameters

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

Related Commands

copy, **delete**, **move**

Examples

- Displays the contents of the `config:` directory.

```
# dir config:

File Name       Last Modified               Size
-----------------------------------------------
unicenter.cfg          Mon Jul  9 11:09:36 2007   3411
autoconfig.cfg         Mon Jul  9 14:20:27 2007   20907

89.2 MB available to config:
#
```

- Displays the contents of the `msgcat` subdirectory of the `store:` directory.

```
# dir store:\\\msgcat

File Name              Last Modified              Size
---------              -------------              ----
crypto.xml             Mon Jul  9 11:09:26 2007   179069
dplane.xml             Mon Jul  9 11:09:26 2007   299644
⋮
xslt.xml               Mon Jul  9 11:09:26 2007   10233

89.2 MB available to store:\msgcat
#
```

disable

Enters User Mode.

Syntax

disable

Guidelines

Use the **disable** command to exit Global configuration mode and enter User mode.

Use the **exit** command to exit Global configuration mode and enter Privileged mode.

Also available in Privileged mode.

Related Commands

enable, **exit**

Examples

- Exits Global configuration mode and enters User Mode.

```
# disable
>
```

- Exits Global configuration mode and enters Privileged Mode.

```
# exit
#
```

dns

Enters DNS Settings configuration mode.

Syntax

dns

no dns

Guidelines

Use the **no dns** command to disable DNS services.

Use the **exit** or **cancel** command to exit DNS Settings configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**, **ip domain**, **ip host**, **ip name-server**

Examples

- Enters DNS Settings configuration mode.

```
# dns
DNS Settings configuration mode
#
```

- Disables DNS services.

```
# no dns
#
```

document-crypto-map

Enters Document Crypto Map configuration mode.

Syntax

document-crypto-map *name*

no document-crypto-map *name*

Parameters

name Specifies the name of the Document Crypto Map.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no document-crypto-map** command to delete a Document Crypto Map.

Use the **exit** or **cancel** command to exit Document Crypto Map Mode and return to Global configuration mode.

Related Commands

cancel, **exit**

documentcache

Enters Document Cache configuration mode for a specific XML Manager

Syntax

documentcache *XML-manager*

Parameters

XML-manager
Specifies the name of an XML Manager.

Guidelines

By default, document caching is disabled. Document caching enables an XML Manager to cache any document that is through HTTP.

In Document Cache configuration mode, you can:

- Enable and specify the size of the document cache
- Design cache policies that determine which documents will be cached and how long they will be retained in the cache
- Delete cache policies
- Clear specific documents or all documents from the document cache.

Use the **exit** command to exit Document Cache configuration mode and enter Global configuration mode.

Related Commands

exit

domain

Enters Application Domain configuration mode.

Syntax

domain *name*

no domain *name*

Parameters

name Specifies the name of the application domain.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **domain** command enters Application Domain configuration mode to create a new Application Domain object or to modify an existing Application Domain object. While in this configuration mode, use the mode-specific commands to define the configuration of the Application Domain object.

To delete an Application Domain object, use the **no domain** command.

To exit this configuration mode without saving configuration changes to the running configuration, use the **cancel** command.

To exit this configuration mode and save configuration changes to the running configuration, use the **exit** command.

Related Commands

cancel, **exit**

failure-notification

Enters Failure Notification configuration mode.

Syntax

failure-notification

no failure-notification

Guidelines

Use the **no failure-notification** command to disable failure reporting. By default, failure reporting is disabled.

Use the **cancel** or **exit** command to exit Failure Notification configuration mode and enter Global configuration mode.

Related Commands

cancel, **exit**, **send error-report**

---

file-capture

Controls the file capture trace utility.

Syntax

file-capture {**always** | **errors** | **off**}

Parameters

always
Enables the file capture trace utility and provides a trace of all appliance traffic.

errors Enables the file capture trace utility and provides a trace for failed transactions only.

off (Default) Disables the file capture trace utility.

Guidelines

The **file-capture** command enables or disables the file capture trace facility. File captures facilitate visibility into erroneous XML or XSLT content as well as provide a record of the sources of erroneous content.

To support file capture, the appliance document trace function creates a RAM-disk to house a WebGUI-accessible virtual file system for tracing all traffic through the appliance. Each transaction appears in a file hierarchy broken down according to the semantics of its URL (that is, a directory for the hostname portion and a directory for each slash portion of the URL) and then further by individual transaction.

Each transaction that represents a transformation stores not only the inputs, but information on style sheets, and disposition of the transformation.

Documents are stored in compressed format to reduce byte count. Should documents need to be removed from the RAM-disk space they will be removed on a FIFO basis.

While browsing the virtual file system repository via the WebGUI, any point in the directory hierarchy can be downloaded either as a tar ball or a zip file.

Note: With file capture enabled (either always or errors), significant performance penalties are imposed. Consequently, file capture should be enabled only in test environments, not in production environments.

Related Commands

packet-capture

Examples

- Enables the file capture trace utility for failed transactions only.

```
# file-capture errors
File capture mode set to errors
#
```

- Disables the file capture trace utility, which restores the default state.

```
# file-capture off
File nature mode set to off
#
```

flash

Enters Flash configuration mode.

Syntax

flash

Guidelines

Use the **exit** command to exit Flash configuration mode and enter Global configuration mode.

Related Commands

exit

ftp-quote-command-list

Enters FTP Quoted Commands List configuration mode.

Syntax

ftp-quote-command-list *name*

no ftp-quote-command-list *name*

Parameters

name Specifies the name of the FTP quoted command list.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no ftp-quote-command-list** command to delete an a FTP quoted commands list.

Use the **cancel** or **exit** command to exit FTP Quoted Commands List configuration mode and enter Global configuration mode.

Related Commands

cancel, **exit**

host-alias

Enters Host Alias configuration mode to map an IP address to an alias.

Syntax

host-alias *alias*

no host-alias *alias*

Parameters

alias Specifies the alias to assign to the specified IP address.

Guidelines

Use the **no host-alias** command to remove an alias map.

Related Commands

cancel, **exit**

---

httpserv

Enters HTTP Server configuration mode.

Syntax

httpserv *name*

httpserv *name address port*

no httpserv *name*

Parameters

name Specifies the name of the HTTP server.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

address Specifies the IP address of the appliance interface that, in conjunction with the port, identifies the interface-port pair that the HTTP server monitors for incoming HTTP client requests.

port Specifies the port of the appliance interface that, in conjunction with the IP address, identifies the interface-port pair that the HTTP server monitors for incoming HTTP client requests.

Guidelines

You can use either of two forms of the **httpserv** command to create an HTTP server.

- The single-command form, creates a basic HTTP server that serves documents only from the general user storage (store:) area.

  If you wish to restrict access to an HTTP server, you can compile an ACL using the **acl**, **allow**, and **deny** commands.
- The multi-command form, creates an HTTP server capable of serving documents from other local storage areas, and provides the ability to add optional features such as user authentication.

  With only the *name* argument, the command enters HTTP Server configuration mode, a mode that supports HTTP server creation with a series of brief single-purpose commands.

  While in HTTP Server configuration mode, you must use the **ip-address**, **local-directory**, and **port** commands to complete server configuration. Optionally, you can use the **authentication**, **mode**, and **start-page** commands to provide enhanced server functions.

If you wish to restrict access to an HTTP server, you can compile an ACL using the **acl**, **allow**, and **deny** commands.

Use the **no httpserv** command to delete an HTTP server.

Use the **exit** command to exit HTTP Server configuration mode and return to Global configuration mode.

Related Commands

acl, exit, **show services**

Examples

- Enters HTTP Server configuration mode to create the `Serv-1` HTTP server.

```
# httpserv Serv-1
HTTP Server configuration mode
#
```

- Creates the `Serv-2` HTTP server on the specified interface.

```
# httpserv Serv-2 192.168.1.200 64000
Installed HTTP server on port 64000
#
```

- Deletes the `Serv-2` HTTP server.

```
# no httpserv Serv-2
#
```

import-execute

Imports an Import Package object.

Syntax

import-execute *package*

Parameters

package
Specifies the name of the Import Package object.

Guidelines

The **import-execute** command imports an existing Import Package object. The Import Package must have been created with the **import-package** command.

Related Commands

import-package

Examples

- Imports the `Norwood` Import Package.

```
# import-execute Norwood
#
```

import-package

Enters Import Configuration File configuration mode.

Syntax

import-package *name*

no import-package *name*

Parameters

name Specifies the name of the Import Configuration File object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **import-package** command enters Import Configuration File configuration mode to create a new Import Configuration File object or to modify an existing Import Configuration File object. While in this configuration mode, use the mode-specific commands to define the configuration of the Import Configuration File object.

To delete an Import Configuration File object, use the **no import-package** command.

To exit this configuration mode without saving configuration changes to the running configuration, use the **cancel** command.

To exit this configuration mode and save configuration changes to the running configuration, use the **exit** command.

Related Commands

cancel, **exit**

---

ims

Enters IMS™ Connect configuration mode.

Syntax

ims *name*

no ims *name*

Parameters

name Specifies the name of the IMS Connect object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

An IMS Connect object handles IMS protocol communications from the Multi-Protocol Gateway to IMS applications. This object contains settings that affect the behavior of the connection.

Use the **no ims** command to delete an IMS Connect object.

Related Commands

cancel, **exit**

include-config

Enters Include Configuration File configuration mode.

Syntax

include-config *filename*

no include-config *filename*

Parameters

filename

Specifies the name of the include configuration object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

An include configuration object references a local or remote configuration file that can be included in other configuration files.

Use the **no include-config** command to delete an include configuration object.

Related Commands

exec

Examples

- Enters Include Configuration configuration mode to create the `standardServiAceProxies` Include Configuration.

```
# include-config standardServiceProxies
Include Configuration configuration mode
#
```

- Deletes the `standardServiAceProxies` Include Configuration.

```
# no include standardServiceProxies
#
```

input-conversion-map

Enters HTTP Input Conversion Map configuration mode.

Syntax

input-conversion-map *name*

no input-conversion-map *name*

Parameters

name Specifies the name of the Input Conversion Map.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no input-conversion-map** command to delete an Input Conversion Map.

Use the **cancel** or **exit** command to exit HTTP Input Conversion Map configuration mode and enter Global configuration mode.

Related Commands

cancel, **exit**

---

interface

Enters Interface configuration mode for a specified interface.

Syntax

interface {[**ethernet 0** | **eth0**] | [**ethernet 1** | **eth1**] | [**ethernet 2** | **eth2**] | [**management 0** | **mgt0**]}

Guidelines

Depending on model type, the appliance provides three or four Ethernet interfaces. There is one dedicated management port (labelled either `MANAGEMENT` or `MGMT`), and two or three network ports (labelled either `ETHERNET` or `NETWORK`).

Use the **no interface** command to delete an Ethernet interface connections from the appliance.

Note: To disable an Ethernet interface, use the **admin-state** command in Interface configuration mode.

Use the **exit** command to exit Interface configuration mode and enter Global configuration mode.

Related Commands

admin-state (Interface), **exit**, **show interface**

Examples

- Enters Interface configuration mode for Ethernet interface 0.

```
# interface ethernet 0
Interface configuration mode (ethernet 0)
#
```

- Enters Interface configuration mode for Ethernet interface 0.

```
# interface eth0
Interface configuration mode (eth0)
#
```

- Deletes Ethernet interface 0 from the network.

```
# interface eth0 disable
no interface eth0#
#
```

---

ip domain

Adds an entry to the IP domain-suffix search table.

Syntax

ip domain *domain*

no ip domain *domain*

Parameters

domain Specifies the base domain name to which a host name can be prefixed.

Guidelines

This command enables the usage on non-fully qualified domain names (host names) by specifying a list of one or more domain names that can be appended to a host name.

Use multiple **ip domain** commands to add more than one entry to the IP domain name table.

The appliance attempts to resolve a host name in conjunction with any domains identified by the **ip domain** command. The host name is resolved as soon as a match is found.

Use the **no ip domain** command to delete an entry from the table.

Related Commands

search-domain

Examples

- Adds the `datapower.com`, `somewhereelse.com`, and `endoftheearth.com` IP domains to the IP domain table. The appliance attempts to resolve the host name `loki` in following ways:
  ```
  loki.datapower.com
  loki.somewhereelse.com
  loki.endoftheearth.com
  ```
  ```
  # ip domain datapower.com
  # ip domain somewhereelse.com
  # ip domain endoftheearth.com
  # xslproxy Proxy-01
  XSL proxy configuration mode
  # remote-address loki 80
  #
  ```
- Removes `datapower.com` from the IP domain search table. The appliance attempts to resolve the host name `loki` in following ways:
  ```
  loki.somewhereelse.com
  loki.endoftheearth.com
  ```
  ```
  # no ip domain datapower.com
  #
  ```

---

ip host

Maps a host name to an IP address.

Syntax

ip host *hostname address*

no ip host {*hostname* | *

Parameters

hostname
Specifies the name of the host.

address Specifies the IP address of the host.

* Specifies all hosts.

Guidelines

Use the **no ip host** command to remove the host name-IP address mapping.

Related Commands

ip name-server, **show ip hosts**, **show ip name-servers**

Examples

- Maps IP address 10.10.10.168 to host `loki`.

```
# ip host loki 10.10.10.168
#
```

- Deletes the map between IP address 10.10.10.168 and host `loki`.

```
# no ip host loki
#
```

- Deletes all maps from the host mapping table.

```
# no ip host *
#
```

ip name-server

Identifies a local DNS provider.

Syntax

ip name-server *address* [*udpPortNumber*] [*tcpPortNumber*] [*flags*] [*max-retries*]

no ip name-server *address*

no ip name-server *

Parameters

address Specifies the IP address of the DNS server.

udpPortNumber
Optionally identifies the UDP port that the DNS server monitors. Use an integer in the range of 0 through 65535. The default is 53.

tcpPortNumber
Optionally identifies the TCP port that the DNS server monitors. Use an integer in the range of 0 through 65535. The default is 53.

flags Optionally specifies protocol-level DNS behavior. Should be set to 0.

max-retries
Optionally specifies the maximum number of times to retransmit an unacknowledged resolution request to the DNS server. The default is 3.

*
Specifies all DNS servers.

Guidelines

Use the **no ip name-server** command to delete a DNS provider.

Note: Unless specifically requested, do not change that DNS parameter.

Related Commands

ip host, **show ip hosts**, **show ip name-servers**

Examples

- Identifies a DNS server at 10.10.10.240 with the default port.

```
# ip name-server 10.10.10.240
#
```

- Identifies a DNS server at 10.10.10.240 with UDP port 6000.

```
# ip name-server 10.10.10.240 6000
#
```

- Deletes the specified DNS provider.

```
# no ip name-server 10.10.10.240
#
```

- Deletes all DNS providers.

```
# no ip name-server *
#
```

iscsi-chap (Type 9235)

Enters iSCSCI CHAP configuration mode.

Syntax

iscsi-chap *name*

no iscsi-chap *name*

Parameters

name Specifies the name of the iSCSI CHAP.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

The **iscsi-chap** command enters iSCSCI CHAP configuration mode. While in the configuration mode, define the credentials for the challenge handshake. During startup, the firmware uses the CHAP to authenticate the defined user over the network. After authentication, administrators have access to the iSCSI storage on the remote server.

Use the **no iscsi-chap** command to remove an iSCSI CHAP.

Related Commands

cancel, **exit**, **iscsi-hba**

Examples

- Enters iSCSCI CHAP configuration mode to create the `CHAP-1` iSCSI CHAP.

```
# iscsi-chap CHAP-1
New iSCSI CHAP configuration
#
```

- Removes the `CHAP-1` iSCSI CHAP.

```
# no iscsi-chap CHAP-1
iscsi-chap CHAP-1 - Configuration deleted.
#
```

---

iscsi-fs-init (Type 9235)

Initializes the iSCSI volume.

Syntax

iscsi-fs-init *name*

Parameters

name Specifies the name of the iSCSI volume to initialize.

Guidelines

The **iscsi-fs-init** command initializes an existing iSCSI volume. Before the iSCSI volume can be initialized, use the **admin-state** command in iSCSI Volume configuration mode to disable the volume. After the iSCSI volume is initialized, it must be enabled for further use.

Related Commands

admin-state (iSCSI Volume)

Examples

- Disables, initializes, and re-enables the `Georgia` iSCSI volume.

```
# iscsi-volume Georgia
Modify iSCSI Volume configuration
# admin-state disabled
# exit

# iscsi-fs-init Georgia
iSCSI filesystem Georgia initialized

# iscsi-volume Georgia
Modify iSCSI Volume configuration
# admin-state enabled
#
```

---

iscsi-fs-repair (Type 9235)

Repairs an iSCSI volume.

Syntax

iscsi-fs-repair *name*

Parameters

name Specifies the name of the iSCSI volume to repair.

Guidelines

The **iscsi-fs-repair** command repairs the iSCSI volume in case it was corrupted by an abnormal shutdown of the appliance or other error. Before the iSCSI volume can be repaired, use the **admin-state** command in iSCSI Volume configuration mode to disable the volume. After the iSCSI volume is repaired, it must be enabled for further use.

Related Commands

admin-state (iSCSI Volume)

Examples

- Disables, repairs, and re-enables the `Georgia` iSCSI volume.

```
# iscsi-volume Georgia
Modify iSCSI Volume configuration
# admin-state disabled
# exit

# iscsi-fs-repair Georgia
iSCSI filesystem Georgia repaired

# iscsi-volume Georgia
Modify iSCSI Volume configuration
# admin-state enabled
#
# iscsi-fs-repair Georgia
iSCSI filesystem Georgia repaired
#
```

---

iscsi-hba (Type 9235)

Enters iSCSI HBA configuration mode.

Syntax

iscsi-hba {**iscsi1** | **iscsi2**}

Parameters

iscsi1 Identifies the existing iSCSI HBA for the `eth1` Ethernet interface.

iscsi2 Identifies the existing iSCSI HBA for the `eth2` Ethernet interface.

Guidelines

The **iscsi-hba** command enters iSCSI HBA configuration mode for the specified HBA. Each DataPower appliance has **iscsi1** and **iscsi2**. You cannot rename or delete either HBA.

Related Commands

cancel, **exit**

iscsi-target (Type 9235)

Enters iSCSI Target configuration mode.

Syntax

iscsi-target *name*

no iscsi-target *name*

Parameters

name Specifies the name of the iSCSI target.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **iscsi-target** command enters iSCSI Target configuration mode. While in this configuration, define the a logical storage volume, or file system, for remote storage.

Use the **no iscsi-target** command to remove an iSCSI target.

Related Commands

cancel, **exit**

iscsi-volume (Type 9235)

Enters iSCSI Volume configuration mode.

Syntax

iscsi-volume *name*

no iscsi-volume *name*

Parameters

name Specifies the name of the iSCSI volume to configure.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **iscsi-volume** command enters iSCSI Volume configuration mode. While in this configuration mode, create, partition, and name the logical storage volume.

Use the **no iscsi-volume** command to remove an iSCSI volume.

Related Commands

cancel, **exit**

loadbalancer-group

Enters Load Balancer Group configuration mode.

Syntax

loadbalancer-group *name*

no loadbalancer-group *name*

Parameters

name Specifies the name of the Load Balancer Group.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

After completing configuration of the Load Balancer Group, assign the group to a specific XML Manager. Assignment of the Load Balancer Group to an XML Manager makes the group available to the DataPower services that this XML Manager supports.

Use the **no loadbalancer-group** command to delete a Load Balancer Group.

Use the **exit** or **cancel** command to exit Load Balancer Group configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**, **show loadbalancer-group**, **show loadbalancer-status**

---

locate-device (Type 9235)

Controls the locate LED.

Syntax

locate-device {**on** | **off**}

Parameters

on Activates the locate LED light.

off (Default) Deactivates the locate LED light.

Guidelines

The **locate-device** command activates or deactivates the locate LED light on Type 9235 appliances. The locate LED is on the front of the appliance.

- When activated, the locate LED light is illuminated in blue.
- When deactivated, the locate LED light is not illuminated.

Only administrators in the `default` domain with the appropriate permissions can control the locate LED.

Examples

- Activates the locate LED light.

```
# locate-device on
#
```

- Deactivates the locate LED light

```
# locate-device off
#
```

known-host

Adds or removes an SSH peer as an SSH known host.

Syntax

known-host *host* **ssh-rsa** *key*

no known-host *host*

Parameters

host Specifies the fully-qualified host name or IP address for the peer. For example:

```
ragnarok.datapower.com
10.97.111.108
```

ssh-rsa
Identifies RSA as the key type.

key Specifies the host public key for the peer. For example:

```
AAAAB3NzaC1yc2EAAAABIwAAAIEA1J/99rRvdZmVvkaKvcG2a+PeCm25
p8OJl87SA6mtFxudA2ME6n3lcXEakpQ8KFTpPbBXt+yDKNFR9gNHIfRl
UDho1HAN/a0gEsvrnDY5wKrTcRHrqDc/x0buPzbsEmXi0lud5Pl7+BXQ
VpPbyVujoHINCrx0k/z7Qpkozb4qZd8==
```

Guidelines

The **known-host** command adds an SSH peer as an SSH known host.

The **no known-host** command removes an SSH peer as an SSH known host.

Examples

- Adds `ragnarok.datapower.com` by host name as an SSH known host.

```
# known-host ragnarok.datapower.com ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEA1J/99rRvdZmVvkaKvcG2a+PeCm25
p8OJl87SA6mtFxudA2ME6n3lcXEakpQ8KFTpPbBXt+yDKNFR9gNHIfRl
UDho1HAN/a0gEsvrnDY5wKrTcRHrqDc/x0buPzbsEmXi0lud5Pl7+BXQ
VpPbyVujoHINCrx0k/z7Qpkozb4qZd8==
#
```

- Adds `ragnarok.datapower.com` by IP address as an SSH known host.

```
# known-host 10.97.111.108 ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEA1J/99rRvdZmVvkaKvcG2a+PeCm25
p8OJl87SA6mtFxudA2ME6n3lcXEakpQ8KFTpPbBXt+yDKNFR9gNHIfRl
UDho1HAN/a0gEsvrnDY5wKrTcRHrqDc/x0buPzbsEmXi0lud5Pl7+BXQ
VpPbyVujoHINCrx0k/z7Qpkozb4qZd8==
#
```

- Removes `ragnarok.datapower.com` by IP address as an SSH known host.

```
# no known-host 10.97.111.108
#
```

ldap-search-parameters

Enters LDAP Search Parameters configuration mode.

Syntax

ldap-search-parameters *name*

no ldap-search-parameters *name*

Parameters

name Specifies the name of the LDAP Search Parameters object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **ldap-search-parameters** command enters LDAP Search Parameters configuration mode. In this configuration, you can create an LDAP Search Parameters object. This object is a container for that parameters that are used to perform an LDAP search to retrieve the distinguished name (DN) of the user.

Use the **cancel** or **exit** command to leave LDAP Search Parameters configuration mode and enter Global configuration mode.

Use the **no ldap-search-parameters** command to delete a LDAP Search Parameters object.

Related Commands

cancel, **exit**

---

load-interval

Specifies the duration of a measurement interval.

Syntax

load-interval *measurement-interval*

Parameters

measurement-interval
Specifies the measurement interval in milliseconds. Use an integer in the range of 500 through 5000. The default is 1000.

Guidelines

The **load-interval** command specifies the duration of a measurement interval. During this interval, system load is estimated and expressed as a percentage. Use this command in conjunction with the **show load** command to monitor system load. The greater the percentage the greater the use of system resources.

Related Commands

show cpu, **show load**

Examples

- Specifies an measurement interval of 2.5 seconds.

```
# load-interval 2500
#
```

logging category

Enters Log Category configuration mode or delete a custom logging category.

Syntax

logging category *name*

no logging category *name*

Parameters

name Specifies the name for a custom logging category.

Guidelines

Use the **no logging category** command to delete the custom logging category.

Related Commands

cancel, **exit**

logging event

Adds an event class (a set of related events) and a priority to an existing log.

Syntax

logging event *name category priority*

no logging event *name category*

Parameters

name Specifies the name of the existing log to which an event class will be added.

category
Specifies the name of an event-class to add.

priority
Identifies the event priority. The priority indicates that all events that are greater than or equal to this value are logged. Events use the following priority in descending order:

- **emerg** (Emergency)
- **alert** (Alert)
- **critic** (Critical)
- **error** (Error)
- **warn** (Warning)
- **notice** (Notice)
- **info** (Information)
- **debug** (Debug)

Guidelines

Use the **show logging event** command to display a list of event classes.

Use the **show logging priority** command to display a list of event priorities.

Use the **no logging event** command to remove an event class from a log.

Related Commands

show logging event, **show logging priority**

Examples

- Adds all events of critical, alert, or emergency priority to the `Alarms` log.

```
# logging event Alarms all critic
#
```

- Specifies which event classes and which event priorities to add to the `CryptoLog` log.

```
# logging event CryptoLog schema error
# logging event CryptoLog xmlfilter error
# logging event CryptoLog crypto error
# logging event CryptoLog ssl error
# logging event CryptoLog auth warning
#
```

- Removes the schema event class from the `CryptoLog` log.

```
# no logging event CryptoLog schema
#
```

logging eventcode

Adds an event code to the subscription list for a specific log.

Syntax

logging eventcode *target event-code*

no logging eventcode *target event-code*

Parameters

target Specifies the name of an existing log target.

event-code
Specifies the hexadecimal value of the event code.

Guidelines

The **logging eventcode** commands adds an event code to the subscription list for the specified log target. This command is equivalent to using the **event-code** command in Logging configuration mode.

Use the **show logging target** command to display a list of log targets.

Use the **View List of Event Codes** from the WebGUI to view a list of all event codes.

Use the **no** form of the **logging eventcode** command to remove an event code from the inclusion list to the specified log.

Related Commands

logging eventfilter, **logging target**, **event-code** (Logging), **show logging target**

logging eventfilter

Adds an event code to the suppression list for a specific log.

Syntax

logging eventfilter *target event-code*

no logging eventfilter *target event-code*

Parameters

target Specifies the name of an existing log target.

event-code
Specifies the hexadecimal value of the event code.

Guidelines

The **logging eventfilter** commands adds an event code to the suppression list for the specified log target. This command is equivalent to using the **event-filter** command in Logging configuration mode.

Use the **show logging target** command to display a list of log targets.

Use the **View List of Event Codes** from the WebGUI to view a list of all event codes.

Use the **no** form of the **logging eventfilter** command to remove an event code from the exclusion list of the specified log.

Related Commands

logging eventcode, **logging target**, **event-filter** (Logging), **show logging target**

---

logging object

Adds an object filter to a specific log.

Syntax

logging object *name object class*

no logging object *name object class*

Parameters

name Specifies the name of the existing log to which to add an object filter.

object Identifies the object type.

class Identifies a specific instance of the target class.

Guidelines

Use **logging object** to enable a finer granularity in specifying log contents. You can restrict log entries, for example, to those events issued by a specific XSL Proxy or XML Firewall, or to a set of identified service providers.

Refer to Table 4 for specific class identifiers.

Table 4. Logging object identifiers

AAAPolicy	FilterAction	Service
AccessControl	HTTPInputConversionMap	ShellAlias
AccessControlList	HTTPProxyService	SmtpClientHelper
CertMonitor	HTTPService	SNMPSettings
CompileOptionsPolicy	HTTPUserAgent	SSHService
ConfigBase	ImportPackage	SSLProxyProfile
CountMonitor	IncludeConfig	SSLProxyService
CRLFetch	InternalProxy	Statistics
Crypto	IPInterface	StylePolicy
CryptoCertificate	LoadBalancerGroup	StylePolicyAction
CryptoEngine	LogLabel	StylePolicyRule
CryptoFWCred	LogTarget	SystemSettings
CryptoIdentCred	Matching	TAM
CryptoKerborosKDC	MessageFlowControl	TCPProxyService
CryptoKey	MessageMatching	TelnetService
CryptoProfile	MessageMonitor	Throttler
CryptoSSKey	MessageType	TraceTarget
CryptoValCred	MgmtInterface	URLMap
DeviceManagementService	MQConfiguration	URLRefreshPolicy
DeviceSettings	MQGW	URLRewritePolicy
DNSNameService	MQhost	User
DocumentCryptoMap	MQproxy	UserGroup
Domain	MQQM	WebGUI
DurationMonitor	NetworkConfiguration	XMLFirewallService
DynamicSchema	NetworkSettings	XMLManager
DynamicStylesheet	NTPService	xmltrace
DynamicXMLContentMap	RADIUSSettings	XPathRoutingMap
ErrorReportSettings	RBMSettings	XSLCoprocService
EthernetInterface	SchemaExceptionMap	XSLProxyService
EventLog		

Use the **no logging object** command to delete an object filter from an existing log.

Examples

- Adds an object filter to the `Alarms` log. This log will record only events that are issued by the `Proxy-1` XSL Proxy. Event priority uses the existing configuration of the `Alarms` log.

  ```
  # logging object Alarms XSLProxyService Proxy-1
  #
  ```

- Deletes an object filter from the `Alarms` log. This log will record those events set by the original log configuration.

  ```
  # no logging object Alarms XSLProxyService Proxy-1
  #
  ```

logging target

Enters Logging configuration mode.

Syntax

logging target *name*

no logging target *name*

Parameters

name Specifies the name of the system log.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

After entering Logging configuration mode, you should first use the **type** command to identify the log type.

Additional configuration requirements and options are dependent upon the log type.

Use the **no logging target** command to delete an event log.

Related Commands

cancel, **exit**

---

loglevel

Sets the log priority for events to log.

Syntax

loglevel *priority*

Parameters

priority

Specifies the type of events written to the local system log and can be expressed as either keyword or integer. Log events are characterized in descending order of criticality.

- **emerg** or 0
- **alert** or 1
- **critic** or 2
- **error** or 3
- **warn** or 4
- **notice** or 5
- **info** or 6
- **debug** or 7

Guidelines

The **loglevel** command determines which system-generated events to log to the basic event log. The log priority also functions as filter and determines which events to forward to a remote syslog daemon. In contrast, **syslog** specifies the events that will be forwarded to a remote appliance.

In the absence of an argument, **loglevel** displays the current log-level.

The log levels can be expressed as character strings or as integer values, with 0 equating to **emergency** (most critical) and 6 equating to **info** (least critical).

By default the basic log level is set to notice (5).

When issued with an argument, **loglevel** specifies that all events of greater or equal criticality to the argument are logged.

Note: The **loglevel**, **logsize**, and **syslog** commands provide the ability to configure a rudimentary basic logging system.

Users, however, are encouraged to use the logging target command to enter Logging configuration mode. From within this mode, users can exercise more precise control over log formats and contents.

Related Commands

logsize, **show log**, **syslog**

Examples

- Sets the priority to critical, which specifies that critical, alert, and emergency events are logged.

```
# loglevel critical
#
```

- Sets the priority to 2, which specifies that critical, alert, and emergency events are logged.

```
# loglevel 2
#
```

- Sets the priority to debug, which specifies that all events are logged. This setting is not intended for production environments.

```
# loglevel 7
#
```

- Displays the current priority.

```
# loglevel
loglevel is 7 debug
#
```

---

logsize

Sets the size of a basic event log.

Syntax

logsize *size*

Parameters

size Specifies the size of the log in lines. The default is 200.

Guidelines

In the absence of an argument, **logsize** displays the size of the log file in lines.

Note:

The **loglevel**, **logsize**, and **syslog** commands provide the ability to configure a rudimentary basic logging system.

Use the **logging target** command to enter Logging configuration mode. From this mode, define more precise control over log formats and contents.

Related Commands

loglevel, **show log**

Examples

- Sets the log size to 250 lines.

```
# logsize 250
#
```

- Displays the configured log size in lines.

```
# logsize 250
#
```

matching

Enters Matching Rule configuration mode.

Syntax

matching *name*

no matching *name*

Parameters

name Specifies the name of the Matching Rule.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** command to leave Matching Rule configuration mode and enter Global configuration mode.

A Matching Rule contains one or more shell-style match patterns that are used to evaluate candidate HTTP headers and URLs. These rules are used in the implementation of Processing Policy objects. A Processing Policy uses Matching Rule objects to determine whether a candidate XML document is subject to specific processing instructions in the policy.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for procedural details about the creation and implementation of Matching Rule and Processing Policy objects.

Use the **no matching** command to delete a Matching Rule.

Related Commands

cancel, **exit**

memoization

Enables the optimization of XPath expressions for a specific XML Manager.

Syntax

memoization *XML-manager*

no memoization *XML-manager*

Parameters

XML-manager
Specifies the name of an XML manager.

Guidelines

Memoizing an XPath expression adds a transparent caching wrapper to the expression, so that expression values that have already been calculated are returned from a cache rather than being recomputed each time. Memoization can provide significant performance gains for computing-intensive calls.

Memoization is enabled by default, and should rarely, if ever, be disabled. It is possible, however, that with certain style sheets, memoization could inflict a performance penalty. The identification of such style sheets is largely a matter of trial and error.

Use the **no memoization** command to disable XPath expression optimization.

Examples

- Disables XPath optimizations for the `mgr1` XML Manager.

```
# no memoization mgr1
XML memoization successfully disabled
XML memoization successfully updated
#
```

- Restores the default condition by enabling XPath optimizations for the `mgr1` XML Manager.

```
# memoization
XML memoization successfully enabled
XML memoization successfully updated
#
```

---

message-matching

Enters Message Matching configuration mode.

Syntax

message-matching *name*

no message-matching *name*

Parameters

name Specifies the name of the traffic-flow definition.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

The **message-matching** command create a traffic-flow definition that describes a traffic stream to be subject to administrative monitoring and control.

When in Message Matching configuration mode, you can specify traffic stream characteristics in terms of traffic origin (IP address), HTTP header content, SSL identity, or requested documents.

Use the **cancel** or **exit** command to leave Message Matching configuration mode and enter Global configuration mode.

Use the **no message-matching** command to delete a traffic-flow definition.

Related Commands

cancel, **exit**, **reset**

message-type

Enters Message Type configuration mode.

Syntax

message-type *name*

no message-type *name*

Parameters

name Specifies the name of the message class.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **message-type** command creates a message class consists of one or more traffic-flow definitions that were created previously with the **message-matching** command. It identifies a set of traffic streams that are subject to specific, rules-based administrative monitoring and control.

Use the **cancel** or **exit** command to leave Message Type configuration mode and enter Global configuration mode.

Use the **no message-type** command to delete a message class.

Related Commands

cancel, **exit**

metadata

Enters Processing Metadata configuration mode.

Syntax

metadata *name*

no metadata *name*

Parameters

name Specifies the name of the Processing Metadata object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

While in Processing Metadata configuration mode you define the contents of the Metadata Processing object, a list or manifest, of metadata items that are returned in an XML nodeset to the object using the Metadata. This is typically an AAA Policy.

Use the **cancel** or **exit** command to leave Processing Metadata configuration mode and enter Global configuration mode.

Use the **no metadata** command to delete a Processing Metadata object.

Related Commands

cancel, **exit**

---

mkdir

Creates a subdirectory.

Syntax

mkdir local:///*subdirectory*

Parameters

local:///*subdirectory*
The subdirectory to create in the local: directory.

Guidelines

The **mkdir** command creates subdirectories in the local: directory on the DataPower appliance. You can create subdirectories for application-specific files such as style sheets and schemas.

Use the **rmdir** command to delete subdirectories.

Related Commands

rmdir

Examples

- Creates the `stylesheets` subdirectory of the `local:` directory.

```
# mkdir local:///stylesheets
Directory 'local:///stylesheets' successfully created.
#
```

- Creates the `C-1` subdirectory in the `stylesheets` subdirectory of the `local:` directory.

```
# mkdir local:///stylesheets/C-1
Directory 'local:///stylesheets/C-1' successfully created.
#
```

---

monitor-action

Enters Message Filter Action configuration mode.

Syntax

monitor-action *name*

no monitor-action *name*

Parameters

name Specifies the name of the control procedure.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A monitor action is a control procedure that specifies an action or set of actions to take when a monitored message class exceeds a configured threshold.

Use the **cancel** or **exit** command to leave Message Filter Action configuration mode and enter Global configuration mode.

Use the **no monitor-action** command to delete a control procedure.

Related Commands

cancel, **exit**, **monitor-count**, **monitor-duration**

---

monitor-count

Enters Message Count Monitor configuration mode.

Syntax

monitor-count *name*

no monitor-count *name*

Parameters

name Specifies the name of the monitor.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A monitor count is an incremental, or counter-based, monitor that consists of a target message class, a configured threshold, and a control procedure that is triggered when the threshold is exceeded.

Use the **cancel** or **exit** command to leave Message Count Monitor configuration mode and enter Global configuration mode.

Use the **no monitor-count** command to delete an incremental monitor.

Related Commands

cancel, **exit**, **show message-count-filters**

monitor-duration

Enters Message Duration Monitor configuration mode.

Syntax

monitor-duration *name*

no monitor-duration *name*

Parameters

name Specifies the name of the duration monitor.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A duration, or time-based, monitor consists of a target message class, two thresholds, and a control procedure that is triggered when either threshold is exceeded.

Use the **cancel** or **exit** command to leave Message Duration Monitor configuration mode and enter Global configuration mode.

Use the **no monitor-duration** command to delete a duration monitor.

Related Commands

cancel, **exit**, **monitor-action**, **monitor-count**, **show message-durations**, **show message-duration-filters**

move

Moves a file from one directory to another.

Syntax

move [**-f**] *source-URL destination-URL*

Parameters

-f Overwrites an existing file, if one of the same name already exists.

In the absence of this argument, an attempt to save a file with the same name as an existing file results in a prompt that requests confirmation to overwrite the existing file.

source-URL and *destination-URL*
Specifies the URLs that identify the source file and target destination, respectively. These arguments take the following form:

```
directory:///filename
```

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

filename
Specifies the name of a file in the specified directory.

Guidelines

You can use the **move** command to transfer a file to or from a directory. However, you cannot use the **move** command to copy a file from the private cryptographic area (such as the cert: directory).

Related Commands

copy, **delete**, **dir**

Examples

- Moves a file from the `config:` directory to the `store:` directory.

```
# move config:///startup-config store:///archiveConfig-10
#
```

- Renames a file.

```
# move config:///startup-config config:///archiveConfig-10
#
```

mpgw

Enters Multi-Protocol Gateway configuration mode.

Syntax

mpgw *name*

no mpgw *name*

Parameters

name Specifies the name of the Multi-Protocol Gateway.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no mpgw** command to delete a Multi-Protocol Gateway.

Related Commands

cancel, **exit**

mq-qm

Enters MQ Queue Manager configuration mode.

Syntax

mq-qm *name*

no mq-qm *name*

Parameters

name Specifies the name of the queue manager.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In WebSphere MQ, distributed send and receive queues are managed by a component called a *queue manager*. The queue manager provides messaging services for communicating applications by periodically monitoring or polling queues, by ensuring that sent messages are directed to the correct receive queue, or that messages are routed to another queue manager.

Use the **no mq-qm** command to delete an MQ queue manager object.

Related Commands

cancel, **exit**

---

mq-qm-group

Enters MQ Queue Manager Group configuration mode.

Syntax

mq-qm-group *name*

no mq-qm-group *name*

Parameters

name Specifies the name of the queue manager.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

You can create a MQ Queue Manager Group to implement a failover configuration that provides connection redundancy in the event of a critical queue manager or bus error resulting in loss of connectivity between clients and backend servers.

Use the **no mq-qm-group** command to delete an MQ queue manager group object.

Related Commands

cancel, **exit**

---

mtom

Enters MTOM Policy configuration mode.

Syntax

mtom *name*

no mtom *name*

Parameters

name Specifies the name of the MTOM Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

While in MTOM (SOAP Message Transmission Optimization Mechanism) Policy configuration mode you define an MTOM Policy, which provides a mechanism for optimizing the transmission and wire format of an XML/SOAP message. Optimization is performed by selecting elements with base 64 encoded character data. The selected elements are decoded and attached as MIME attachment parts before transmission. Decoding before transmission reduces the overhead that is associated with base 64 encoded data.

Use the **cancel** or **exit** command to leave MTOM Policy configuration mode and enter Global configuration mode.

Use the **no mtom** command to delete an MTOM Policy.

Related Commands

cancel, **exit**

---

network

Enters Network Settings configuration mode.

Syntax

network

no network

Guidelines

While in Network Settings configuration mode, you can enable or disable the generation of certain Internet Control Message Protocol (ICMP) replies and control the retry and intervals of these messages. By default the appliance replies to the corresponding ICMP requests.

You can also control routing behavior, interface isolation and ECN settings.

Use the **cancel** or **exit** command to leave Network Settings configuration mode and enter Global configuration mode.

Use the **no network** command to reset network settings to their defaults.

Related Commands

cancel, **exit**

---

nfs-client

Enters NFS Client Settings configuration mode.

Syntax

nfs-client

no nfs-client

Guidelines

While in NFS Client configuration mode, you configure NFS client global settings, which are employed in all application domains. By default, the NFS Client is disabled.

Use the **cancel** or **exit** command to leave NFS Client configuration mode and enter Global configuration mode.

Use the **no nfs-client** command to disable the NFS client.

Related Commands

cancel, **exit**

---

nfs-dynamic-mounts

Enters NFS Dynamic Mounts configuration mode.

Syntax

nfs-dynamic-mounts

no nfs-dynamic-mounts

Guidelines

While in NFS Dynamic Mounts configuration mode, you configure NFS dynamic mounts settings, which are employed within the current application domain. By default, the NFS dynamic mounts are disabled; once in NFS Dynamic Mounts configuration mode, use the **admin-state** command to enable dynamic mounts and other commands to specify operational properties.

Use the **cancel** or **exit** command to leave NFS Dynamic Mounts configuration mode and enter Global configuration mode.

Use the **no nfs-dynamic-mounts** command to restore the NFS dynamic mount default settings.

Related Commands

cancel, **exit**

---

nfs-static-mount

Enters NFS Static Mounts configuration mode.

Syntax

nfs-static-mount *name*

no nfs-static-mount *name*

Parameters

name Specifies the name of the NFS static mount object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

While in NFS Static Mounts configuration mode, you configure NFS static mounts settings, which are employed within the current application domain.

Use the **cancel** or **exit** command to leave NFS Static Mounts configuration mode and enter Global configuration mode.

Use the **no nfs-static-mount** command to delete an NFS static mount.

Related Commands

cancel, **exit**

---

ntp

Identifies an NTP (Network Time Protocol) server.

Syntax

ntp *address* [*interval*]

no ntp *address*

Parameters

address Specifies the IP address of an NTP server.

interval
Optionally specifies the number of seconds between NTP updates.

Guidelines

After an NTP server has been identified by the **ntp** command, the appliance functions as a Simple Network Time Protocol (SNTP) client as described in RFC 2030.

By default, while functioning as an NTP client, the appliance issues time-of-day requests to the specified NTP server every 15 minutes (900 seconds).

The appliance supports one NTP server at a time. To designate a new NTP server, use the **no ntp** command to delete the current server, and then use the **ntp** command to designate the new server.

Also available in Privileged mode.

Related Commands

clock, **ntp-service**, **show ntp time**

Examples

- Identifies 10.10.12.13 as an NTP server. The appliance issues NTP updates every 15 minutes (900 seconds).

```
# ntp 10.10.12.13
Enabling NTP service
#
```

ntp-service

Enters NTP Service configuration mode.

Syntax

ntp-service

no ntp-service

Guidelines

After an NTP server is identified by the **remote-server** command (NTP Service), the appliance functions as a Simple Network Time Protocol (SNTP) client as described in RFC 2030. While functioning as an NTP client, the appliance issues time-of-day requests to the specified NTP server every 15 minutes (900 seconds).

The appliance supports one NTP server at a time. To designate a new NTP server, use the **no ntp-service** command to delete the current server, and then use the **remote-server** command to designate the new server.

Related Commands

clock, **ntp**, **show ntp time**

Examples

- Enters NTP Service configuration mode.

```
# ntp-service
NTP Service configuration mode
#
```

peer-group

Enters Peer Group configuration mode.

Syntax

peer-group *name*

no peer-group *name*

Parameters

name Specifies the name of the peer group.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

While in Peer Group configuration mode, you identify members of an SLM Monitoring Peer Group. Group members run identical SLM policies and aggregate data providing enforcement of SLM policies across multiple appliances.

Use the **cancel** or **exit** command to leave Peer Group configuration mode and enter Global configuration mode.

Use the **no peer-group** command to delete a Peer Group.

Related Commands

cancel, **exit**

---

policy-attachments

Enters Policy Attachment configuration mode.

Syntax

policy-attachments *name*

no policy-attachments [*name*]

Parameters

name Specifies the name of the object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** command to exit Policy Attachment configuration mode and return to Global configuration mode.

Use the **no policy-attachments** command to delete an Policy Attachment object.

Related Commands

cancel, **exit**

---

policy-parameters

Enters Policy Parameters configuration mode.

Syntax

policy-parameters *name*

no policy-parameters [*name*]

Parameters

name Specifies the name of the object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** command to exit Policy Parameters configuration mode and return to Global configuration mode.

Use the **no policy-parameters** command to delete an Policy Parameters object.

Related Commands

cancel, **exit**

radius

Enters RADIUS Settings configuration mode.

Syntax

radius

no radius

Guidelines

While in RADIUS configuration mode, you configure RADIUS (Remote Authentication Dial-In User Service) settings.

Use the **cancel** or **exit** command to exit RADIUS configuration mode and return to Global configuration mode.

Use the **no radius** command to disable RADIUS service.

Related Commands

cancel, **exit**

raid-activate (Type 9235)

Activates an existing array volume.

Syntax

raid-activate *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-activate** command activates a hard disk array volume that is in the `inactive` state, typically with the `foreign volume` inactive state.

Examples

- Activates the RAID Volume in the disks as the active RAID volume.

```
# raid-activate raid0
```

raid-delete (Type 9235)

Deletes an array volume.

Syntax

raid-delete *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-delete** command makes the disks that are presently a hard disk array volume on the appliance no longer an array volume, removing all metadata. This action destroys the content of the array volume.

Examples

- Deletes the hard disk array volume on the disks.

```
# raid-delete raid0
```

raid-initialize (Type 9235)

Initializes an array volume.

Syntax

raid-initialize *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-initialize** command makes the two disks into a hard disk array volume. This action destroys any prior content of the array volume.

Examples

- Builds a RAID volume on the disks on the system.

```
# raid-initialize raid0
```

raid-rebuild (Type 9235)

Forces a rebuild of an array volume.

Syntax

raid-rebuild *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-rebuild** command forces a rebuild of a hard disk array volume. The contents of the primary disk in the array volume are copied to the secondary disk.

Examples

- Rebuilds the array volume `raid0`.

```
# raid-rebuild raid0
```

raid-volume (Type 9235)

Enters Hard Disk Array configuration mode for an existing array volume.

Syntax

raid-volume *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-volume** command enters Hard Disk Array configuration mode for an existing hard disk array enabled appliance. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Related Commands

cancel, **exit**

Examples

- Enters Hard Disk Array configuration mode for volume `raid0`.

```
# raid-volume raid0
Hard Disk Array configuration mode
#
```

raid-volume-initialize-filesystem (Type 9235)

Initializes the filesystem.

Syntax

raid-volume-initialize-filesystem *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-volume-initialize-filesystem** command initializes the filesystem on the hard disk array to allow it to be made active. This action destroys the existing contents of the hard disk array.

Examples

- Makes a new file system on the `raid0` hard disk array volume.

```
# raid-volume-initialize-filesystem raid0
#
```

raid-volume-repair-filesystem (Type 9235)

Repairs the file system.

Syntax

raid-volume-repair-filesystem *name*

Parameters

name Specifies the name of the existing hard disk array volume. For appliances that have a hard disk array for auxiliary data storage, the name is `raid0`.

Guidelines

The **raid-volume-repair-filesystem** command repairs the file system on the hard disk array, in case it was corrupted by an abnormal shutdown of the appliance or other error.

Examples

- Repairs the file system on the `raid0` hard disk array volume.

```
# raid-volume-repair-filesystem raid0
#
```

rbm

Enters RBM Settings configuration mode.

Syntax

rbm

no rbm

Guidelines

While in RBM configuration mode, you configure role-based management (RBM) settings.

Use the **cancel** or **exit** command to exit RBM configuration mode and return to Global configuration mode.

Use the **no rbm** command to disable RBM service. Note that this command can disable WebGUI access.

Related Commands

cancel, **exit**

refresh stylesheet

Forces a reload of a specified style sheets by an XML Manager.

Syntax

refresh stylesheet {* | *XML-manager*} *match*

Parameters

XML-manager
Specifies the name of a specific XML Manager.

* Specifies all XML Manager objects.

match Defines a shell-style match pattern that defines the style sheets to refresh.

Guidelines

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

The **refresh stylesheet** command forces a one time reload. In contrast, the **xslrefresh** command enables a periodic policy-based refresh cycle.

Related Commands

xslrefresh

Examples

- Refreshes the `OrderEntry.xsl` style sheet that is cached by the `mgr1` XML Manager.

```
# refresh stylesheet mgr1
 http://www/somecompany/XML/stylesheets/OrderEntry.xsl
#
```

- Refreshes any style sheet with a URL that contains `datapower` and that is cached by the `mgr1` XML Manager.

```
# refresh stylesheet mgr1 *datapower*
#
```

- Refreshes all style sheets in all XML Manager objects.

```
# refresh stylesheet * *
#
```

remove chkpoint

Deletes a checkpoint configuration file.

Syntax

remove chkpoint *name*

Parameters

name Specifies the name of the checkpoint configuration file.

Guidelines

The **remove chkpoint** command deletes the named checkpoint configuration file from the domain-specific chkpoint: directory.

The command is equivalent to using the **delete** command to remove the file from a specified directory.

Related Commands

delete, **rollback chkpoint**, **save chkpoint**, **show chkpoints**

Examples

- Deletes the `foo` checkpoint configuration file.

```
# remove chkpoint foo
Configuration Checkpoint foo deleted
#
```

or

```
# delete chkpoints:\\\foo.zip
File deletion successful
#
```

---

reset domain

Deletes the configuration for a domain.

Syntax

Resets an application domain from the `default` domain.

reset domain *name*

Resets an current domain.

reset domain

Parameters

name Specifies the name of the domain to reset.

Guidelines

The **reset domain** command resets the configuration for the domain.

- When invoked from the `default` domain without an explicit domain, the command resets the configuration for the `default` domain.
- When invoked from the `default` domain with an explicit domain, the command resets the configuration for the specified domain.
- When invoked from an application domain, the command resets the configuration of this domain.

The **reset domain** command is different from the **no domain** command.

- The **reset domain** command deletes all configured objects in the domain but retains the configuration of the domain and all files in the local: directory.
- The **no domain** command deletes all configured objects in the domain, deletes all files in the domain, and deletes the configuration of the domain itself.

Related Commands

domain

Examples

- Resets the `Test` domain while in the `default` domain.

```
# reset domain Test
Resetting 'Test' will delete all services configured within the domain!
Do you want to continue? [y/n]:y
Domain reset successfully.
#
```

- Resets the `Test` domain while in the `Test` domain.

```
[Test]# reset domain
reset domain
Resetting 'Test' will delete all services configured within the domain!
Do you want to continue? [y/n]:y
Domain reset successfully.
[Test]#
```

reset username

Re-enables a locked out account.

Syntax

reset username *account* [*password*]

Parameters

account
Specifies the name of the user account to reset.

password
Optional. Specifies the new, temporary password for the account.

Guidelines

The **reset username** command allows a privileged administrator to re-enable an account after lockout. If the invocation does not include the password, the interface prompts for the password. In either case, the interface prompts for confirmation of the password.

After an administrator re-enables the account, the administrator needs to send the owner of the account the new password. The next time the owner of the account logs in, the interface prompts for a new password. This password must comply with the corporate password policy.

When the appliance is in Common Criteria mode, the password set by the administrator and the new password provided by the user on initial login after the account is re-enabled must meet the following criteria:

- Be at least six characters in length
- Contain at least one nonalphanumeric character
- Not be one of the past five passwords

Examples

- Re-enables the `suehill` account by changing the password for the account (without the administrator specifying the password).

```
# configure terminal
(config)# reset username suehill
Enter new password: ********
Re-enter new password: ********
Password for user 'suehill' is reset.
(config)#
```

restart domain

Restores a domain to the state defined by the startup configuration file.

Syntax

restart domain [*domain*]

Parameters

domain Optionally identifies the domain to be restarted.

Guidelines

The **restart domain** command restarts the named domain. Without an explicit domain, the command restarts the current domain.

Domain restart is accomplished by discarding the existing domain running configuration and re-initializing the domain with its startup configuration file, either autoconfig.cfg or the startup file that is identified by the **boot config** command.

Related Commands

boot config, **save-config overwrite**, **write memory**

Examples

- Restarts the `AcceptanceCriteria` domain from the `default` domain.

```
# restart domain AcceptanceCriteria
Restarting 'AcceptanceCriteria' will affect all services configured
within the domain!
Do you want to continue? [y/n]:y
Domain 'AcceptanceCriteria' restarted.
#
```

- Restarts the `AcceptanceCriteria` domain from the `AcceptanceCriteria` domain.

```
[AcceptanceCriteria]# restart domain
Restarting 'AcceptanceCriteria' will affect all services configured
within the domain!
Do you want to continue? [y/n]:y
Domain 'AcceptanceCriteria' restarted.
[AcceptanceCriteria]#
```

---

rmdir

Removes a subdirectory.

Syntax

rmdir local:///*subdirectory*

Parameters

local:///*subdirectory*
The subdirectory to remove from the local: directory.

Guidelines

The **rmdir** command removes subdirectories from the local: directory.

Related Commands

mkdir

Examples

- Deletes the `stylesheets` subdirectory and all its contents from the `local:` directory.

```
# rmdir local:///stylesheets
Removing 'local:///stylesheets' will delete all files
including subdirectories!
Do you want to continue? [y/n]:y
Directory 'local:///stylesheets' successfully deleted.
#
```

rollback chkpoint

Loads a checkpoint configuration file as the running configuration.

Syntax

rollback chkpoint *name*

Parameters

name Specifies the name of the checkpoint configuration file.

Guidelines

The **rollback chkpoint** command reverts the running configuration to the configuration that is defined in the named checkpoint configuration file.

Note: If you use the **rollback chkpoint** command after a configuration was persisted with the **write memory** command, the appliance uses the configuration in the checkpoint configuration file, not the configuration in the startup configuration file. Before reverting to a checkpoint configuration, you might want to compare the timestamp on the checkpoint configuration file the startup configuration file. If the startup configuration file is more recent than the checkpoint configuration file, you might want to validate the differences to ensure that a necessary configuration change is forgotten.

Related Commands

rollback chkpoint, **save chkpoint**, **show chkpoints**, **write memory**

Examples

- Reverts to the configuration in the `foo` checkpoint configuration file.

```
# rollback chkpoint foo
Rollback Chkpoint foo is initiated (may take a few
minutes to complete)
#
```

rule

Enters Stylesheet Policy Rule configuration mode.

Syntax

rule *name*

rule *name* {**request** | **response**}

no rule *name*

Parameters

name Specifies the name of the global processing rule.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

request
Identifies the global rule as a request rule. This type of rule applies to client requests only.

response
Identifies the global rule as a response rule. This type of rule applies to server responses only.

Guidelines

The **rule** command (Global), in conjunction with the **match** command (Stylesheet Policy), provides one method of rule creation. An alternative method is provided by the **request-rule**, **response-rule**, and **rule** commands (all available in Stylesheet Policy).

Rules initiated from Global configuration mode are global named objects that are available for assignment to one or more Processing Policies. Rules initiated from Stylesheet Policy configuration mode are internal to a specific Processing Policy and cannot be reused by other policies.

In the absence of a **request** or **response** keyword, the global rule is bidirectional and is applied to both client requests and server responses.

Global rules can be used in the implementation of a Processing Policy. A Processing Policy enables an XML Firewall or XSL Proxy to select an appropriate style sheet with which to filter or transform an input document. The selected style sheet can be used in conjunction with, or instead of, processing instructions contained within the input document.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for procedural details regarding the creation and implementation global rules and Processing Policies.

Use the **no rule** command to delete a global processing rule.

Related Commands

cancel, **exit**, **match**, **matching**, **response-rule**, **request-rule**, **rule** (Stylesheet Policy), **show rule**, **stylepolicy**

Examples

- Creates the `star` matching rule to use for matching all URLs.

```
# matching star
Matching Rule configuration mode
# urlmatch *
# exit
```

- Creates the `valClientServer` global bidirectional rule that validates client and server input against the specified schema.

```
# rule valClientServer
Stylesheet Policy Rule configuration mode
# validate INPUT schema store:///soap-envelope-1.1.xsd
# xform INPUT store:///identify.xsl OUTPUT
# exit
```

- Creates the `bidirectional-schema-val` Stylesheet Policy and adds the matching rule and associated global rule to the policy.

```
# stylepolicy bidirectional-schema-val
Stylesheet Policy configuration mode
# match star valClientServer
```

- Deletes the `valClientServer` global rule.

```
# no rule valClientServer
#
```

save chkpoint

Creates a checkpoint configuration file.

Syntax

save chkpoint *name*

Parameters

name Specifies the name of the checkpoint configuration file. Do not specify a file extension.

Guidelines

The **save chkpoint** command creates the named checkpoint configuration file in the domain-specific chkpoint: directory. The created archive file has the .zip extension and contains the configuration data for the objects in that domain. Basically, a checkpoint configuration file is equivalent to a ZIP bundle created with the **backup** command.

A checkpoint configuration file remains on the appliance until you explicitly delete it with the **remove chkpoint** command. You can revert the running configuration to the configuration that is contained in a checkpoint configuration file with the **rollback chkpoint** command.

You can save up to the number of checkpoint configuration files are defined by the **maxchkpoints** command.

Related Commands

backup, **maxchkpoints** (Application Domain), **remove chkpoint**, **rollback chkpoint**, **show chkpoints**, **write memory**

Examples

- Creates the `foo` checkpoint configuration file.

```
# save chkpoint foo
Save Configuration Checkpoint foo scheduled (may take a few
minutes to complete)
#
```

save error-report

Creates an error report.

Syntax

save error-report

Guidelines

The **save error-report** command creates an error report in the local file temporary:///error-report.txt. This file can then be used for debugging purposes. The file can be copied off the appliance with the **send error-report** command.

This command takes no arguments. It automatically creates a file that contains any backtrace or watchdog error report, the contents of the audit log, and the running configuration.

If there is insufficient space to write the file, the following error message indicates this condition:

```
Could not write error report to {FILE}
```

If you receive this message, delete files to make sufficient space.

Related Commands

failure-notification, **send error-report**, **send file**

Examples

- Generates an error report file in `temporary:///error-report.txt`. This file is then sent to `supportteam@customer.com` via the specified mail server.

```
# save error-report
# send error-report smtp.customer.com
 "Danger Will Robinson" supportteam@customer.com
#
```

---

save internal-state

Saves the internal state as a text file.

Syntax

save internal-state

Guidelines

The **save internal-state** command writes the internal state to the temporary:///internal-state.txt file

Examples

- Saves the internal state of the appliance.

```
# save internal-state
Internal state written to temporary:///internal-state.txt
#
```

---

save-config overwrite

Specifies system behavior after a running configuration is saved.

Syntax

save-config overwrite

no save-config overwrite

Guidelines

By default the **Save Config** button and the **write memory** command write the current running configuration to config:///autoconfig.cfg and designate this file as the startup configuration.

Use **no save-config overwrite** command to override the default behavior and designate the file defined with the **boot config** command as the startup configuration.

Related Commands

boot config, **write memory**

Examples

- Uses `config:///autoconfig.cfg` as the startup configuration after saving the running configuration.

```
# save-config overwrite
 Save Config will overwrite startup config.
#
```

- Uses the file that is defined with the **boot config** command as the startup configuration after saving the running configuration.

```
# no save-config overwrite
 Save Config will not overwrite startup config.
#
```

---

schema-exception-map

Enters Schema Exception Map configuration mode.

Syntax

schema-exception-map *name*

no schema-exception-map *name*

Parameters

name Specifies the name of the Schema Exception Map

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** command to exit Schema Exception Map configuration mode and return to Global configuration mode.

Use the **no schema-exception-map** command to delete a Schema Exception Map.

Related Commands

cancel, **exit**

Examples

- Enters Schema Exception Map configuration mode to create the SEM-1 Schema Exception Map.

```
# schema-exception-map SEM-1
Schema Exception Map configuration mode
#
```

- Deletes the SEM-1 Schema Exception Map.

```
# no schema-exception-map SEM-1
#
```

search results

Enables the search results optimization algorithm for an XML Manager.

Syntax

search results *XML-manager*

no search results *XML-manager*

Parameters

XML-manager
Specifies the name of an XML manager.

Guidelines

The search results algorithm provides more efficient processing of style sheets that contain '//' (all element) expressions. If the style sheet contains an all element expression (for example, //inning), the XML manager caches all occurrences of the inning element.

Use the **no search results** command to disable the search results optimization algorithm.

Examples

- Disables the search results optimization algorithm for the mgr1 XML Manager.

```
# no search results mgr1
Configuration successfully updated
#
```

- Enables the search results algorithm for the mgr1 XML Manager, which restores the default condition.

```
# search results mgr1
Configuration successfully updated
#
```

send error-report

Sends an error report as e-mail.

Syntax

send error-report *mail-server subject email-address* [*email-address* ...]

Parameters

mail-server
Identifies a local SMTP server by IP address or by host name.

subject Specifies the text string for the subject field of the outgoing message.

email-address
Specifies the list of fully-qualified email addresses that identifies one or more recipients.

Guidelines

When using **send error-report**, keep the following points in mind.

- With automated notification enabled, the appliance searches for a backtrace file (that contains failure related data) at each reboot. On finding a backtrace file, the appliance sends the file to the appliance does not issue any notifications.
- Place this command within the configuration file to ensure that it will be executed upon each reboot.
- If identifying the SMTP server by host name, you must place this command after the **ip name-server** command in the configuration file — otherwise the appliance will be unable to perform host name: IP address resolution.
- This command can be issued to immediately send an error report that was generated with the **save error-report** command.

Related Commands

failure-notification, **save error-report**, **send file**

Examples

- On system restart, transmits a notification to `supportteam@customer.com` and `weekendcoverage@attbb.com` via the specified mail server.

```
# send error-report
 smtp.customer.com ALERT! supportteam@customer.com weekendceverage@attbb.com
```

send file

Enables SMTP-based file transmission.

Syntax

send file *URL mail-server email-address*

Parameters

URL Identifies the target file and takes one of the following forms:
- `audit:///`*`filename`*
- `pubcert:///`*`filename`*
- `config:///`*`filename`*
- `store:///`*`filename`*
- `image:///`*`filename`*
- `tasktemplates:///`*`filename`*
- `logstore:///`*`filename`*
- `temporary:///`*`filename`*
- `logtemp:///`*`filename`*

mail-server
Identifies a local SMTP server by IP address or by host name.

email-address
Specifies the fully-qualified Email addresses of the file recipient.

Guidelines

You cannot send a file from the cert:, dp:, or dpcert: directory.

Related Commands

failure-notification, **send error-report**

Examples

- Sends the `config:///autoconfig.cfg` file to `supportteam@customer.com` through the `smtp.customer.com` mail server.

```
# send file config:///autoconfig.cfg
 smtp.customer.com supportteam@customer.com
#
```

service battery-installed

Notifies the firmware that the battery was changed.

Syntax

service battery-installed

Guidelines

The **service battery-installed** command resets the installation date of the battery. Use this command after changing the battery.

After the expiration period of 2 years, the firmware generates log messages that the battery should be changed.

service nagle

Enables or disables the Nagle algorithm.

Syntax

service nagle

Guidelines

The **service nagle** command enables or disables the Nagle slow packet avoidance algorithm. By default, the algorithm is enabled.

Examples

- Disables the Nagle algorithm.

```
# service nagle
disabled service nagle algorithm.
```

- Enables the Nagle algorithm.

```
# service nagle
enabled service nagle algorithm.
```

service-monitor

Enters Web Services Monitor configuration mode.

Syntax

service-monitor *name*

no service-monitor *name*

Parameters

name Specifies the name of the Web Services Monitor.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In XML Manager configuration mode, you can configure the target Web Service Monitor to perform WSDL-aware monitoring.

Use the **no service-monitor** command to delete a Web Services Monitor.

Examples

- Enters Web Services Monitor Configuration mode to create the `WSMonitor-2` Web Services Monitor.

```
# service-monitor WSMonitor-2
Web Services Monitor configuration mode
#
```

set-system-var

Creates a system variable.

Syntax

set-system-var *name value*

Parameters

name Specifies the name of the variable to be created, and takes the form:

`var://system/`*`contextName`*`/`*`name`*

`var://system`
Specifies the required prefix that identifies a global variable.

contextName
Specifies the required name of the context within which the global variable resides.

value Specifies the value to assign.

Guidelines

The **set-system-var** command creates a new system variable that actions or style sheets can access with the **dp:variable()** function. System variables, while globally accessible, must be defined within a specified context.

For example `var://system/Notes/jim` is correct as it sets the variable `jim` in the context `Notes` within the `system` scope.

However, `var://system/Notes` and `var://system/Notes/` are incorrect as they specify only a scope and context, but not a variable.

Examples

- Creates the `counter` system variable in the `signerID` context.

```
# set-system-var var://system/signerID/counter 0
#
```

simple-rate-limiter

Enters Simple Rate Limiter configuration mode.

Syntax

simple-rate-limiter *name*

no simple-rate-limiter *name*

Parameters

name Specifies the name of the Simple Rate Limiter.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **simple-rate-limiter** command enters Simple Rate Limiter configuration mode. In this mode, you can create a Simple Rate Limiter. A Simple Rate Limiter is used by a Web Application Firewall.

Use the **cancel** or **exit** commands to exit Simple Rate Limiter configuration mode and return to Global configuration mode.

Use the **no simple-rate-limiter** command to delete a Simple Rate Limiter.

Related Commands

cancel, **exit**

slm-action

Enters SLM Action configuration mode.

Syntax

slm-action *name*

no slm-action *name*

Parameters

name Specifies the name of the SLM Action.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In SLM (Service Level Monitor) Action configuration mode, define an administrative response by defining an action type (log, reject, or shape traffic) and specifying a log priority.

Use the **no slm-action** command to delete an SLM Action.

Related Commands

cancel, **exit**

---

slm-cred

Enters SLM Credential Class configuration mode.

Syntax

slm-cred *name*

no slm-cred *name*

Parameters

name Specifies the name of the SLM Credential Class.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In SLM (Service Level Monitor) Credential Class configuration mode, define a group of credentials subject to an SLM policy and optional match conditions used in conjunction with the defined credential group.

Use the **no slm-cred** command to delete an SLM Credential Class.

Related Commands

cancel, **exit**

---

slm-policy

Enters SLM Policy configuration mode.

Syntax

slm-policy *name*

no slm-policy *name*

Parameters

name Specifies the name of the SLM Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In SLM Policy configuration mode, define an SLM policy by specifying an evaluation method, noting peer groups and assigning statements to the policy.

Use the **no slm-policy** command to delete an SLM Policy.

Related Commands

cancel, **exit**

---

slm-rsrc

Enters SLM Resource Class configuration mode.

Syntax

slm-rsrc *name*

no slm-rsrc *name*

Parameters

name Specifies the name of the SLM Resource Class.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In SLM Resource Class configuration mode, define a set of resources that are subject to an SLM Policy.

Use the **no slm-rsrc** command to delete an SLM Resource Class.

Related Commands

cancel, **exit**

---

slm-sched

Enters SLM Schedule configuration mode.

Syntax

slm-sched *name*

no slm-sched *name*

Parameters

name Specifies the name of the SLM Schedule.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In SLM Schedule configuration mode, define an SLM Schedule by specifying the days and hours when the schedule is in effect.

Use the **no slm-sched** command to delete an SLM Schedule.

Related Commands

cancel, **exit**

---

snmp

Enables or disables SNMP.

Syntax

snmp

no snmp

Guidelines

While in SNMP Settings configuration mode, you configure Simple Network Management Protocol settings.

Use the **cancel** or **exit** command to exit SNMP configuration mode and return to Global configuration mode.

Use the **no snmp** command to disable SNMP.

Related Commands

cancel, **exit**, **system**

Examples

- Enters SNMP configuration mode.

```
# snmp
SNMP Settings configuration mode
#
```

- Disables SNMP.

```
# no snmp
#
```

---

soap-disposition

Enters SOAP Header Disposition Table configuration mode.

Syntax

soap-disposition *name*

no soap-disposition *name*

Parameters

name Specifies the name of the disposition table.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **soap-disposition** command enters SOAP Header Disposition Table configuration mode and creates the named object if it does not exist. If the object exists, enters this configuration mode to allow for editing.

A SOAP Header Disposition Table object contains a list of instructions that controls how to handle SOAP headers, child elements, or both SOAP headers and child elements. This object is used by an `xform` action that uses the `store:///soap-refine.xsl` style sheet.

Use the **no soap-disposition** command to delete the named object.

Use the **cancel** or **exit** command to exit this configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

---

source-ftp-poller

Enters FTP Poller Front Side Handler configuration mode.

Syntax

source-ftp-poller *handler*

no source-ftp-poller *handler*

Parameters

handler

Specifies the name of the FTP Poller Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

Use the **no source-ftp-poller** command to delete an FTP Poller Front Side Handler object.

Related Commands

cancel, **exit**

---

source-ftp-server

Enters FTP Server Front Side Handler configuration mode.

Syntax

source-ftp-server *handler*

no source-ftp-server *handler*

Parameters

handler

Specifies the name of the FTP Server Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-ftp-server** command to delete an FTP Server Front Side Handler object.

Related Commands

cancel, **exit**

---

source-http

Enters HTTP Front Side Handler configuration mode.

Syntax

source-http *handler*

no source-http *handler*

Parameters

handler

Specifies the name of the HTTP Front Side Handler.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-http** command to delete an HTTP Front Side Handler object.

Related Commands

cancel, **exit**

---

source-https

Enters HTTPS (SSL) Handler configuration mode.

Syntax

source-https *handler*

no source-https *handler*

Parameters

handler

Specifies the name of the Secure HTTP Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-https** command to delete a Secure HTTP Front Side Handler object.

Related Commands

cancel, **exit**

source-imsconnect

Enters IMS Connect Handler configuration mode.

Syntax

source-imsconnect *name*

no source-imsconnect *name*

Parameters

name Specifies the name of the IMS Connect Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

An IMS Connect object handles IMS protocol communications with Multi-Protocol Gateway clients.

Use the **no source-imsconnect** command to delete an IMS Connect Handler object.

Related Commands

cancel, **exit**

source-mq

Enters MQ Front Side Handler configuration mode.

Syntax

source-mq *handler*

no source-mq *handler*

Parameters

handler

Specifies the name of the MQ Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-mq** command to delete an MQ Front Side Handler object.

Related Commands

cancel, **exit**

---

source-nfs-poller

Enters NFS Poller Front Side Handler configuration mode.

Syntax

source-nfs-poller *handler*

no source-nfs-poller *handler*

Parameters

handler

Specifies the name of the NFS Poller Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-nfs-poller** command to delete an NFS Poller Front Side Handler object.

Related Commands

cancel, **exit**

---

source-raw

Enters Stateless Raw XML Handler configuration mode.

Syntax

source-raw *handler*

no source-raw *handler*

Parameters

handler

Specifies the name of the Stateless Raw XML Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-raw** command to delete a Stateless Raw XML Handler object.

Related Commands

cancel, **exit**

---

source-ssh-server

Enters SFTP Server Front Side Handler configuration mode.

Syntax

source-ssh-server *handler*

no source-ssh-server *handler*

Parameters

handler

Specifies the name of the SFTP Server Front Side Handler.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-ssh-server** command to delete an SFTP Server Front Side Handler object.

---

source-stateful-tcp

Enters Stateful Raw XML Handler configuration mode.

Syntax

source-stateful-tcp *handler*

no source-stateful-tcp *handler*

Parameters

handler

Specifies the name of the Stateful Raw XML Handler.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no source-stateful-tcp** command to delete a Stateful Raw XML Handler object.

Related Commands

cancel, **exit**

---

source-tibems

Enters TIBCO EMS Front Side Handler configuration mode.

Syntax

source-tibems *handler*

no source-tibems *handler*

Parameters

handler

Specifies the name of the TIBCO EMS Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

Use the **no source-tibems** command to delete a TIBCO Front Side Handler object.

Related Commands

cancel, **exit**

---

source-wasjms

Enters WebSphere JMS Front Side Handler configuration mode.

Syntax

source-wasjms *handler*

no source-wasjms *handler*

Parameters

handler

Specifies the name of the WebSphere JMS Front Side Handler object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

Use the **no source-wasjms** command to delete a WebSphere JMS Front Side Handler object.

Related Commands

cancel, **exit**

---

sql-source

Enters SQL Data Source configuration mode.

Syntax

sql-source *name*

no sql-source *name*

Parameters

name Specifies the name of the object. Supply an alphanumeric string that is unique within the current application domain.

Guidelines

This feature is an optional feature and might not be available on your appliance.

The **sql-source** command enter SQL Data Source configuration mode. In this mode, you can create or edit a SQL Data Source object. When a data source object exists,

the appliance can interact (insert, query, and so forth.) with the databases hosted by the object through the `sql` processing action.

Use the **no sql-source** command to delete an SQL Data Source object.

Related Commands

cancel, **exit**

---

ssh

Enables SSH on appliance interfaces.

Syntax

ssh

ssh *address port*

no ssh [*address*]

Parameters

address Specifies the IP address of a local interface.

port Identifies the port of a local interface that services SSH traffic. Use an integer in the range of 0 through 65535.

Guidelines

SSH is disabled by default. You can use the optional arguments to explicitly bind SSH to a specified interface. If you explicitly bind SSH to an interface, you must have previously configured that interface.

In the absence of an explicit address assignment, SSH first attempts to bind to the management port. If the management port has not been previously configured, SSH binds to all configured interfaces.

You can compile an ACL to restrict access to SSH.

Use the **no ssh** command to disable SSH.

Related Commands

xml-mgmt

Examples

- Enables SSH on port 22 (the default port) of the specified interface.

```
# ssh 10.10.13.4
SSH service listener enabled
#
```

- Enables SSH on port 2200 of the specified interface.

```
# ssh 10.10.13.4 2200
SSH service listener enabled
#
```

- Disables SSH on all interfaces, which restores the default state.

```
# no ssh
SSH service listener disabled
#
```

sslforwarder

Creates an SSL Proxy (forwarder) service.

Syntax

sslforwarder *name* [*local-address* | **0**] *local-port remote-address remote-port sslproxy* [*priority*]

no sslforwarder *name*

Parameters

name Specifies the name of the SSL forwarding service.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

local-address
Specifies the IP address (primary or secondary) of an appliance interface. In conjunction with the local port, identifies a specific IP address and port that the SSL Proxy service monitors.

0 Indicates all appliance interfaces. In conjunction with the local port, identifies a specific port on all IP interfaces that the SSL Proxy service monitors.

local-port
Identifies the local port. Use an integer in the range of 0 through 65535. In conjunction with the IP address, identifies the IP addresses and ports that the SSL Proxy service monitors.

remote-address
Specifies the IP address of the remote SSL peer. In conjunction with the remote port number, identifies a specific destination.

remote-port
Identifies the destination port. Use an integer in the range of 0 through 65535. In conjunction with the remote IP address, identifies a specific destination.

sslproxy
Specifies the name of a forward (client) SSL Proxy Profile.

priority
Indicates the priority to assign for scheduling or for resource allocation. Use one of the following values:

low Receives below normal priority.

normal
(Default) Receives normal priority.

high Receives above normal priority.

Guidelines

An SSL Proxy service is most commonly used to enable the transmission of syslog-ng messages with an SSL wrapper (for example, Stunnel).

Before you can create an SSL Proxy service, you must create an identification credentials set, an SSL profile, and an SSL Proxy Profile to be used by the SSL Proxy service.

Use the **no sslforwarder** command to delete an SSL Proxy service.

Related Commands

idcred, **logging target**, **profile sslproxy**, **type** (Logging)

Examples

- Creates the `syslog-ng-stunnel` SSL Proxy service. Monitors port *7777* on local interface 10.10.13.21 and forwards traffic, via an SSL connection, from this interface-port pair to the destination address 192.168.1.159:7777. The `sslClient` SSL Proxy Profile provides the credentials to establish the SSL connection.

  Creates the `syslog-1` log that directs log events to the local address 10.10.13.21:7777. The `syslog-ng-stunnel` SSL Proxy service monitors that port, forwards log events, via an SSL connection, to 10.10.1.159:1724.

```
# sslforwarder syslog-ng-stunnel 10.10.13.21 7777
192.168.1.159 7777 sslClient
New SSL Proxy Service configuration
#

# logging target syslog-1
New Log Target configuration
# type syslog-ng
# remote-address 10.10.13.21 7777
# event cli error
# exit
Logging configuration successful
#
```

- Deletes the `syslog-ng-stunnel` SSL Proxy service.

```
# no sslforwarder syslog-ng-stunnel
sslforwarder syslog-ng-stunnel - configuration deleted.
#
```

---

sslproxy

Creates an SSL Proxy Profile that defines an SSL service type.

Syntax

Create an SSL proxy profile for a client

sslproxy *name* **client** *client-profile* [**client-cache** {**on** | **off**}]

sslproxy *name* **forward** *client-profile* [**client-cache** {**on** | **off**}]

Create an SSL proxy profile for a server

sslproxy *name* **server** *server-profile* [**sess-timeout** *timer-value*] [**cache-size** *entries*] [**client-auth-optional** {**on** | **off**}] [**client-auth-always-request** {**on** | **off**}]

sslproxy *name* **reverse** *server-profile* [**sess-timeout** *timer-value*] [**cache-size** *entries*] [**client-auth-optional** {**on** | **off**}] [**client-auth-always-request** {**on** | **off**}]

Create an SSL proxy profile for both client and server

sslproxy *name* **two-way** *server-profile client-profile* [**sess-timeout** *timer-value*] [**cache-size** *entries*] [**client-cache** {**on** | **off**}] [**client-auth-optional** {**on** | **off**}] [**client-auth-always-request** {**on** | **off**}]

sslproxy *name* **both** *server-profile client-profile* [**sess-timeout** *timer-value*] [**cache-size** *entries*] [**client-cache** {**on** | **off**}] [**client-auth-optional** {**on** | **off**}] [**client-auth-always-request** {**on** | **off**}]

Deletes an SSL proxy profile
no sslproxy *name*

Parameters

name Specifies the name of the SSL Proxy Profile.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

client (or **forward**)
Indicates that the SSL proxy functions as an SSL client (or functions in the forward direction). In client proxy mode, SSL is used over the appliance-to-server connection. Client proxy mode requires a client cryptographic profile.

server (or **reverse**)
Indicates that the SSL proxy functions as an SSL server (or functions in the reverse direction). In server proxy mode, SSL is used over the appliance-to-client connection. Server proxy mode requires a server cryptographic profile.

two-way (or **both**)
Indicates that the SSL proxy functions as both an SSL client and SSL server (or functions in both directions). In two-way mode, SSL is used over both the appliance-to-server connection and over the appliance-to-client connection. Two-way mode requires both a client and server cryptographic profile.

server-profile
When the operational mode is either **client** or **two-way**, identifies the Crypto Profile that is used by the SSL client to authenticate itself to the SSL server.

client-profile
When the operational mode is **server** or **two-way**, identifies the Crypto Profile that is used by the SSL server to authenticate itself to SSL clients.

sess-timeout *timer-value*
Sets the session timeout value for the server-side session cache. Use an integer in the range of 1 through 86400 to define the time, in seconds, that session-specific state data is maintained in the cache.

By default, an SSL server caches SSL session-specific state data for 5 minutes (300 seconds). A value of 0 disables server-side caching.

cache-size *entries*
Optionally sets the maximum size of the session cache. Use an integer in the range of 1 through 500 to define the cache size in kilo entries (1024 entries). For example, a value of 10 defines a maximum cache size of 10,240 entries. By default, the maximum cache size is 20 (20,480 entries).

client-cache {**on** | **off**}
Optionally disables client-side caching of session state data.

on (Default) Enables client-side caching.

off Disables client-side caching.

client-auth-optional {**on** | **off**}
When acting as an SSL server, controls when SSL client authentication is optional.

on Requests but does not require client authentication. When there is no client certificate, the request does not fail.

off (Default) Requires client authentication only when the server cryptographic profile has an assigned Validation Credentials.

client-auth-always-request {**on** | **off**}
When acting as an SSL server, controls when to request SSL client authentication.

on Always requests client authentication.

off (Default) Requires client authentication only when the server cryptographic profile has an assigned Validation Credentials.

Guidelines

The **sslproxy** command creates an SSL Proxy Profile that defines an SSL service type (client, server, two-way). Before creating an SSL Proxy Profile, one or more Crypto Profile objects must exist. Without the referenced cryptographic profiles, the SSL proxy profile is created but in the `down` operational state.

Use the **profile** command in Crypto mode to create a cryptographic profile.

Use the **no sslproxy** command to delete an SSL Proxy Profile.

Related Commands

profile (Crypto)

Examples

- Creates the `SSL-1` server SSL Proxy Profile using the `Low` Crypto Profile on the appliance-to-client connections. Default values are used for the other properties.
  ```
  # sslproxy SSL-1 server Low
  ```
- Creates the `SSL-2` client SSL Proxy Profile using the `High` Crypto Profile on appliance-to-server connections. Default values are used for the other properties.
  ```
  # sslproxy SSL-2 client High
  ```
- Creates the `SSL-3` client SSL Proxy Profile using the `ClientIDs` Crypto Profile on the appliance-to-server connections. Client-side caching is disabled.
  ```
  # sslproxy SSL-3 client ClientIDs client-cache off
  ```
- Creates the `SSL-4` two-way SSL Proxy Profile using the `NoMD` Crypto Profile on the appliance-to-client connections and the `High` Crypto Profile on appliance-to-server connections. Default values are used for the other properties.
  ```
  # sslproxy SSL-4 two-way NoMD High
  ```
- Creates the `SSL-5` server SSL Proxy Profile using the `Low` Crypto Profile on the appliance-to-client connection. The session-specific state data times out after 15 minutes (900 seconds), and maximum cache size is allocated for 102,400 entries. Default values are used for the other properties.
  ```
  # sslproxy SSL-5 reverse Low sess-timeout 900 cache-size 100
  ```
- Creates the `SSL-6` two-way SSL Proxy Profile using the `NoMD` Crypto Profile on the appliance-to-client connections and the `High` Crypto Profile on the appliance-to-server connections. The session-specific state data times out after 15

minutes (900 seconds), the maximum cache size is allocated for 102,400 entries, client-side caching is disabled, and SSL client authentication by the backend server is optional.

```
# sslproxy SSL-6 both NoMD High sess-timeout 900 cache-size 100
client-cache off client-auth-optional on
```

- Deletes the SSL-6 SSL Proxy Profile.

```
# no sslproxy SSL-6
```

ssltrace

Enables an SSL trace of a specified SSL Proxy Profile.

Syntax

ssltrace *name*

Parameters

name Specifies the name of the target SSL Proxy Profile.

Guidelines

This command is available only during Telnet and SSH command sessions.

Press the ENTER key to stop the trace. If the SSL connection terminates before you press ENTER, the firmware displays the following message and ends the SSL trace:

```
SSL connection completed
```

The trace is not specific to a port, but rather to an SSL Proxy Profile. Consequently, the traced object is the first connection using the target SSL Proxy Profile.

Keep in mind that a single SSL Proxy Profile can be used by multiple DataPower services. If the target SSL proxy is two-way, the first connection in either direction (that is, client-to-server or server-to-client) is traced.

A CLI shell supports only a single trace session. However, simultaneous multiple traces (up to a maximum of five sessions) using different shells are allowed.

Note: Tracing severely diminishes performance of the appliance, multiple simultaneous sessions even more so.

Tracing is intended for debugging purposes only and should not be implemented in operational environments.

Examples

- Initiates an SSL trace for the SSL-1 SSL Proxy.

```
# ssltrace SSL-1
Press ENTER to stop tracing
Waiting for connection to begin
⋮
#
```

startup

Starts the DataPower installation wizard.

Syntax

startup

Guidelines

The **startup** command is available when you initially log in to the appliance. You can invoke this command after accepting the DataPower license and changing the password to the `admin` account.

The installation wizard prompts for information about the basic configuration for the appliance. This information that you provide became the running configuration that the appliance uses.

The installation wizard prompts for the following information, which you should gather before invoking the installation wizard:

- Interface IP addresses and masks
- Default gateways
- Interface mode
- IP name servers

When the installation wizard completes, you can review the configuration. After reviewing the configuration, you can save the configuration or discard it.

Related Commands

show startup-config (Global), **show startup-errors** (Global)

Examples

- Starts the installation wizard.

```
# startup
#
```

---

statistics

Initiates statistical data collection.

Syntax

statistics

no statistics

Guidelines

Statistical data collection is disabled by default.

Statistical display (with the **show statistics** command) is not available if statistical data collection is suspended.

Use the **no statistics** command to clear all data collection counters and to suspend statistical data collection.

Related Commands

show statistics

Examples

- Initiates statistical data collection.

```
# statistics
#
```

- Clears all data collection counters and suspends statistical data collection.

```
# no statistics
#
```

---

stylepolicy

Enters Processing Policy configuration mode.

Syntax

stylepolicy *name* [**xsldefault** *URL*] [**filter** *URL*]

no stylepolicy *name*

Parameters

name Specifies the name of the Processing Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

xsldefault *URL*
Identifies a default XSL style sheet used for document transformation. This default style sheet performs transformation only if a candidate XML document fails to match any of the processing rules defined within the named Processing Policy, and if the candidate document does not contain internal transformation instructions.

filter *URL*
Identifies a default XSL filtering style sheet. This default XSL filtering style sheet performs XML filtering only if a candidate XML document fails to match any of the filter rules defined within the named Processing Policy.

Guidelines

A Processing Policy enables a service such as a Web Service Proxy, XML Firewall or XSL Proxy to select an appropriate style sheet with which to filter or transform an input document. The selected style sheet can be used in conjunction with, or instead of, processing instructions contained within the input document.

Use the **no stylepolicy** command to delete a Processing Policy.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for procedural details regarding the creation and implementation of Processing Policies.

Related Commands

cancel, **exit**

Examples

- Enters Processing Policy configuration mode to create the FW-1 Processing Policy.

```
# stylepolicy FW-1
Processing Policy configuration mode
#
```

- Enters Processing Policy configuration mode to create the FW-1 Processing Policy. Designates identity.xsl as default document transformation, and designates soapfilter.xsl as the filtering style sheets.

```
# stylepolicy FW-1
 xsldefault store:///identity.xsl filter
 store:///soapfilter.xsl
Processing Policy configuration mode
#
```

- Deletes the FW-1 Processing Policy.

```
# no stylepolicy FW-1
#
```

no stylesheet

Deletes style sheets from the cache of an XML Manager.

Syntax

no stylesheet *XML-manager match*

Parameters

XML-manager
Specifies the name of an XML manager.

match Defines a shell-style match pattern that defines the style sheets to delete.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

[1-5] Matches 1, 2, 3, 4, or 5

[xy] Matches x or y

Related Commands

refresh stylesheet

Examples

- Deletes the specified style sheet cached by the mgr1 XML Manager.

```
# no stylesheet mgr1
 http://www/somecompany/XML/stylesheets/OrderEntry.xsl
#
```

- Deletes any style sheet cached by the mgr1 XML Manager whose URL contains datapower.

```
# no stylesheet mgr1 *datapower*
#
```

- Deletes all style sheets cached by the mgr1 XML Manager.

```
# no stylesheet mgr1 *
#
```

switch domain

Moves to the specified application domain.

Syntax

switch domain [*domain*]

Parameters

domain Identifies the target domain.

Guidelines

Issue the **switch domain** command without arguments to display the list of available application domains.

Related Commands

domain

Examples

- Moves to the `Domain-1` application domain.

```
# switch domain Domain-1
[Domain-1] #
```

---

syslog

Designates where to forward log messages.

Syntax

syslog *address log-level*

Parameters

address Specifies the IP address of the target workstation.

log-level

Specifies the type of messages to forward to the target workstation. The log level can be a keyword or an integer.

- **emerg** or 0
- **alert** or 1
- **critic** or 2
- **error** or 3
- **warn** or 4
- **notice** or 5
- **info** or 6
- **debug** or 7

Guidelines

The **loglevel**, **logsize**, and **syslog** commands provide the ability to configure a basic logging system. However, you are encouraged to use the **logging target** command to enter Logging configuration mode. In this mode, you can exercise more precise control over log formats and contents.

Log events are characterized in descending order of importance.

Log levels can be expressed as character strings or as integer values, with 0 equating to **emerg** (most important) and 6 equating to **info** (least important).

The log level specifies that all events of greater or equal importance to the argument are logged.

The **loglevel** command specifies the event types that will be logged locally. The **syslog** command specifies a subset of locally logged events that will be forwarded to a remote appliance.

Reception of log events by the remote appliance requires configuration of that appliance or the installation of a syslog daemon.

Note: Events that are forwarded to the syslog facility are filtered through the local event log. Consequently, events that are forwarded to a remote syslog daemon must be either the same as the set of events that are locally logged, or a subset of the events that are locally logged. For example, you can use the **loglevel** command to specify a local log level of 3 (or error), meaning that emergency, alert, critical, and error events are written to the local log facility. With this log level, you can specify a syslog level of 3 (warning), 2 (critical), 1 (alert), or 0 (emergency). You can not specify a syslog level of 4 (warning), 5 (notice), or 6 (info).

Related Commands

loglevel

Examples

- Identifies appliance 10.10.100.17 as the recipient of emergency events only.

```
# syslog 10.10.100.17 0
#
```

- Identifies appliance 10.10.100.17 as the recipient of emergency events only.

```
# syslog 10.10.100.17 emerg
#
```

- Identifies appliance 10.10.100.17 as the recipient of error, critical, alert, and emergency events.

```
# syslog 10.10.100.17 3
#
```

system

Enters System Settings configuration mode.

Syntax

system

Guidelines

Use the **cancel** or **exit** command to exit System Settings configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

tam

Enters TAM (IBM Tivoli Access Manager) configuration mode.

Syntax

tam *name*

Parameters

name Optionally identifies the TAM object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

In the absence of a *name* argument, the appliance firmware creates a TAM object named `default`.

Guidelines

While in TAM configuration mode, you can configure a TAM object. During object configuration, the TAM configuration files must reside on the appliance. To create TAM client configuration files, use the **create-tam-files** command.

Use the **cancel** or **exit** command to exit TAM configuration mode and return to Global configuration mode.

Although native TAM supports both local and remote clients, the appliance supports only remote client operations. The TAM configuration supports only one policy server, and supports only LDAP directories. Although the configuration files allow the specification of URAF (User Registry Adapter Framework) directories, the appliance does not support these directory servers, which include Microsoft® Active Directory and Lotus® Domino®.

TAM is a licensed feature, and requires the presence of a TAM license on the DataPower appliance. Contact your IBM representative, to obtain the needed license.

Related Commands

cancel, **create-tam-files**, **exit**

---

tcpproxy

Creates a TCP proxy that redirects an incoming TCP packet stream to a remote address.

Syntax

tcpproxy *name local-address local-port destination-address destination-port* [*priority*]

no tcpproxy *name*

Parameters

name Specifies the name of the TCP proxy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

local-address
Specifies an IP address (primary or secondary) of an interface. With the local port, identifies the incoming TCP traffic stream that will be redirected.

local-port
Identifies a specific port on the local host. Use an integer in the range of 0 through 65535. With local host, identifies the incoming TCP traffic stream that will be redirected.

destination-address
Identifies the destination host. With the destination port, identifies the recipient of the redirected TCP packet stream. The value can be an IP address or host name.

destination-port
Identifies a port on destination host, Use an integer in the range of 0 through 65535. With the destination host, identifies the recipient of the redirected packet stream.

priority
Indicates the priority to assign for scheduling or for resource allocation. Use one of the following values:

low Receives below normal priority.

normal
(Default) Receives normal priority.

high Receives above normal priority.

Guidelines

The TCP Proxy service terminates the inbound TCP connection, and initiates an outbound TCP connection to the destination address.

Use the **no tcpproxy** command to delete a TCP proxy.

Examples

- Creates a `ForwardHTTP` TCP proxy that redirects incoming traffic received on appliance interface 192.68.14.12:80 to host 10.10.20.100:80.

```
# tcpproxy ForwardHTTP 192.168.14.12 80 10.10.20.100 80
#
```

- Creates the `TCPRelay` TCP proxy that redirects incoming traffic received on appliance interface 192.168.14.12: 6130 to port 49000 on host `ragnarok`.

```
# tcpproxy TCPRelay 192.168.14.12 6130 ragnarok 49000
#
```

- Deletes the `ForwardHTTP` TCP proxy.

```
# no tcpproxy ForwardHTTP
#
```

template

Runs an interactive command line script.

Syntax

template *URL*

Parameters

URL Specifies the fully-qualified location of the interactive command line script.

Guidelines

The **template** command specifies the URL of the interactive command line script. The script is an XML file that can be local or remote to the DataPower appliance. The script must conform to the store:///schemas/dp-cli-template.xsd schema.

To verify whether the script is conformant with the schema, use the **test schema** command.

Related Commands

test schema

Examples

- Verify that `local:///shell-script.xml` conforms to the `store:///schemas/dp-cli-template.xsd` schema.

```
# test local:///shell-script.xml store:///schemas/dp-cli-template.xsd
#
```

- Runs the interactive script as defined in the `local:///shell-script.xml` file.

```
# template local:///shell-script.xml
#
```

---

test hardware

Tests the hardware.

Syntax

test hardware

Guidelines

The **test hardware** command tests the hardware. Depending on the state of the hardware, the command produces output that states the status for each component:

- `success`
- `warning`
- `failure`

The components are broken down into the following categories:

- Backtrace availability
- Interface diagnostics
- Fan diagnostics
- Cryptographic card diagnostics

Samples of success statements are as follows:

- `[success] Backtrace file does not exist`
- `[success] 4 interface expected - 4 interfaces found`
- `[success] MAC address of interface 'eth0' is 00:11:25:27:bf:e7`
- `[success] Statistics for interface 'eth0' show no errors`
- `[success] 6 fans expected - 6 fans found`
- `[success] fan 1 operating within expected range`
- `[success] Status of the crypto 'standard' is fully operational`

Samples of warning statements are as follows:

- `[warning] Backtrace file exists.`
- `[warning] Physical link on interface 'eth0' is down.`
- `[warning] eth1 has invalid MAC (ff:ff:ff:ff:ff)`

Samples of failure statements are as follows:

- `[failure] Expected number of interfaces: 4 - Found: 1`
- `[failure] fan 2 operating outside expected range (rpm too low)`
- `[failure] Status of crypto 'not detected' is unknown.`

The output of the test hardware command is included in any generated error report.

---

test logging

Test the configuration of a Log Category.

Syntax

test logging *category priority message*

Parameters

category
Specifies the name of an existing Log Category.

priority
Identifies the event priority. The priority indicates that all events that are greater than or equal to this value are logged. Events use the following priority in descending order:

- **emerg** (Emergency)
- **alert** (Alert)
- **critic** (Critical)
- **error** (Error)
- **warn** (Warning)
- **notice** (Notice)
- **info** (Information)
- **debug** (Debug)

message
Specifies the text for the message.

Guidelines

The **test logging** command generates an event at the specified level and sends it to the specified log target. Use this command to test the configuration of a Logging Target.

To create a Logging Target, use the Global **logging target** command.

Related Commands

logging target (global)

Examples

- Tests two candidate URLs against the `URLmap-1` URL map.

```
# urlmap URLmap-1
URL map mode
#
# match https://www.company.com/XML/stylesheets/*
# match https://www.distributer.com/*xsl
# exit

# test urlmap URLmap-1 https://www.company.com/XML/stylesheets/style1.xsl match
# test urlmap URLmap-1 https://www.distributer.com/Renditions/XML2HTML.xsl match
#
```

test schema

Tests conformity of an XML file against a schema.

Syntax

test schema *file schema*

Parameters

file Specifies the URL of the XML file to test.

schema Specifies the URL of the schema.

Guidelines

The **test schema** command tests the conformity of an XML file against an XSD schema file.

Examples

- Tests conformity of the `xyzbanner.xml` XML file against the `dp-user-interface.xsd` schema.

```
# test schema store:///xyzbanner.xml store:///schemas/dp-user-interface.xsd
Performing validation of document 'store:///xyzbanner.xml' using
  schema 'store:///schemas/dp-user-interface.xsd' ...
Document validation completed: OK.
#
```

test urlmap

Tests a candidate URL against a specific URL map.

Syntax

test urlmap *URL-map URL*

Parameters

URL-map
Specifies the name of the URL map.

URL Specifies the candidate URL string.

Guidelines

Use **test urlmap** to verify or troubleshoot a URL map that is used by a Stylesheet Refresh Policy. The command returns "`match`" if the candidate URL matches a pattern in the URL map or returns "`no match`" if the URL does not match a pattern.

The **test urlmap** command tests a candidate URL against a single URL map. The **test urlrefresh** command tests a candidate URL against all URL maps used by a Stylesheet Refresh Policy.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of URL maps and Stylesheet Refresh Policies.

Related Commands

interval urlmap, **match**, **test urlrefresh**, **urlmap**, **urlrefresh**, **xslrefresh**

Examples

- Tests two candidate URLs against the URLmap-1 URL map.

```
# urlmap URLmap-1
URL map mode
#
# match https://www.company.com/XML/stylesheets/*
# match https://www.distributer.com/*xsl
# exit

# test urlmap URLmap-1 https://www.company.com/XML/stylesheets/style1.xsl match
# test urlmap URLmap-1 https://www.distributer.com/Renditions/XML2HTML.xsl match
#
```

test tcp-connection

Tests the TCP connection to a remote appliance.

Syntax

test tcp-connection *host port* [*timeout*]

Parameters

host Specifies the target host. Use either the IP address or host name.

port Specifies the target port.

timeout
Specifies an optional timeout value, the number of seconds that the CLI waits for a response from the target host. The default is 10.

Related Commands

ip host, **ip name-server**, **ping**, **traceroute**

Examples

- Confirms an available TCP connection to the specified host on port number 80 (the well-known HTTP port), using the default timeout value (10 seconds).

```
# test tcp-connection ragnarok 80
TCP connection successful
#
```

- Confirms an available TCP connection to the specified IP address on port 21 (the well-known FTP control port). The timeout value is 5 seconds.

```
# test tcp-connection 192.168.77.27 21 5
TCP connection successful
#
```

test urlrefresh

Tests a given URL against a specific Stylesheet Refresh Policy.

Syntax

test urlrefresh *name URL*

Parameters

name Specifies the name of a Stylesheet Refresh Policy.

URL Specifies the candidate URL.

Guidelines

Use the **test urlrefresh** command to verify or troubleshoot a Stylesheet Refresh Policy. The command returns "`match`" if the candidate URL matches a pattern in any URL map that is used by the policy or returns "`no match`" if the URL does not match a pattern.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of URL maps and Stylesheet Refresh Policies.

Related Commands

interval urlmap, **match**, **test urlmap**, **urlmap**, **urlrefresh**, **xslrefresh**

Examples

- Tests two candidate matches against the `2aday` Stylesheet Refresh Policy. Output confirms the matches and displays the refresh interval and the match pattern.

```
# configure terminal
Global configuration mode
# urlrefresh 2aday
URL refresh mode
# interval urlmap ShipmentQuery 43200
# interval urlmap PriceQuery 4300
# exit

# test urlrefresh 2aday https://www.distributer.com/XML/xforms/2HTML.xsl
match: 43200 seconds. rule:https://www.company.com/XML/stylesheets/*

# test urlrefresh 2aday https://www.amajoraccount.com/Zeus/RenderHtml.xsl
match: 43200 seconds. rule:https://www.amajoraccount.com/Zeus/*xsl
#
```

test urlrewrite

Tests a URL against a URL Rewrite Policy.

Syntax

test urlrewrite *URL-rewrite-policy URL*

Parameters

URL-rewrite-policy
Specifies the name of the URL Rewrite Policy to test.

URL Specifies the candidate URL.

Guidelines

The **test urlrewrite** command tests a URL against a URL Rewrite Policy. Use this command to verify or troubleshoot a URL Rewrite Policy. The command evaluates the candidate URL in terms of the PCRE specified in the URL Rewrite Policy rules, and returns the rewritten URL and the stylesheet URL.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

absolute-rewrite, **urlrewrite**

Examples

- Enter the URL Rewrite Policy configuration mode to create the `RW-1` URL Rewrite Policy. Adds a rule to the URL Rewrite Policy. Applies the changes and returns to Global configuration mode.

```
# urlrewrite RW-1
URL rewrite policy configuration mode
# rewrite  (.*)xsl=(.*)\?(.*) $1xsl=ident.xsl?$3 http://mantis:8000/$2
# exit
```

- Tests a candidate URL against the URL Rewrite Policy.

```
# test urlrewrite RW-1
 http://mantis:8000/foo/bar/my.cgi?x=y&xsl=style.xsl?/input.xml

input replace -> http://mantis:8000/foo/bar/my.cgi?x=y&xsl=ident.xsl?/input.xml
style replace -> http://mantis:8000/style.xsl
#
```

---

tfim

Enters TFIM configuration mode.

Syntax

tfim *name*

Parameters

name is a required string that identifies the TFIM object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In TFIM (IBM Tivoli Federated Identity Manager) configuration mode, you configure a TFIM object that provides the information needed to locate and access a TFIM server.

Use the **cancel** or **exit** command to exit TFIM configuration mode and return to Global configuration mode.

TFIM is a licensed feature, and requires the presence of a TAM license on the DataPower appliance. Contact your IBM representative, to obtain the needed license.

Related Commands

cancel, **exit**

---

throttle

Specifies appliance behavior when faced with a low memory condition.

Syntax

throttle *throttle-threshold kill-threshold timeout*

no throttle

Parameters

throttle-threshold
: Specifies the free memory threshold (expressed as a percentage of total memory) at which the appliance starts to implement a memory conservation algorithm. Use an integer in the range of 1 through 100. The default is 20.

kill-threshold
: Specifies the free memory threshold (expressed as a percentage of total memory) at which the appliance restarts itself. This value must be less than the throttle threshold. Use an integer in the range of 0 through 100. The default is 5.

timeout
: Specifies the duration of the memory conservation algorithm. Use a positive integer. The default is 30.

Guidelines

By default, the appliance monitors its memory usage and reacts to low memory conditions by first refusing to accept new connections. If the refusal to accept new connections does not free sufficient memory, the appliance then responds by restarting itself.

When free memory falls below the throttle threshold (a measure of free memory expressed as a percentage of total memory), the appliance refuses to accept new connections. If the amount of free memory does not rise above the throttle threshold in the specified timeout (expressed in seconds), the appliance restarts. If free memory falls below the kill threshold (also a measure of free memory expressed as a percentage of total memory), the appliance restarts immediately.

Use the **no throttle** command to turn off throttling.

Related Commands

show throttle

Examples

- Customizes appliance behavior when faced with a low memory condition.
  - The appliance stops accepting new connections when the amount of free memory falls below 15% of total memory.
  - The appliance restarts if the amount of free memory does not rise above 15% of total memory within 40 seconds of its refusal to accept new connections.

– The appliance restarts immediately if the amount of free memory falls below 10% of total memory.

```
# throttle 15 10 40
#
```

- Restores the default throttle settings.
  – The appliance stops accepting new connections when the amount of free memory falls below 20% of total memory.
  – The appliance restarts if the amount of free memory does not rise above 20% of total memory within 30 seconds of its refusal to accept new connections.
  – The appliance restarts immediately if the amount of free memory falls below 5% of total memory.

```
# throttle 20 5 30
#
```

- Disables throttling.

```
# no throttle
#
```

- Disables throttling.

```
# throttle 0 0 0
#
```

tibems-server

Enters TIBCO EMS configuration mode.

Syntax

tibems-server *name*

no tibems-server *name*

Parameters

name Specifies the name of the TIBCO EMS server object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

While in TIBCO EMS configuration mode, you specify the properties required to locate and access one or more TIBCO EMS servers used in conjunction with Multi-Protocol Gateways.

Use the **cancel** or **exit** command to exit TIBCO EMS configuration mode and return to Global configuration mode.

Use the **no tibems-server** command to delete a TIBCO EMS server object.

Related Commands

cancel, **exit**

timezone

Enters Timezone configuration mode.

Syntax

timezone

Guidelines

While in Timezone configuration mode, you configure the time zone settings for the appliance. The time zone alters the display of time to the user.

Use the **cancel** or **exit** command to exit Timezone configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

---

traceroute

Traces the network path to a target host.

Syntax

traceroute *host*

Parameters

host Specifies the target host as either the IP address or host name.

Related Commands

ip host, **ip name-server**, **ping**, **test tcp-connection**

Examples

- Confirms an available TCP connection to `loki` .

```
# traceroute loki
```

---

uddi-registry

Enters UDDI Registry configuration mode.

Syntax

uddi-registry *name*

Parameters

name Specifies the name of the UDDI Registry object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In UDDI (Universal Description Discovery and Integration) Registry configuration mode, you configure a UDDI Registry object that provides the information needed to locate and access a UDDI Registry.

Use the **cancel** or **exit** command to exit UDDI Registry configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

---

uddi-subscription

Enters UDDI Subscription configuration mode.

Syntax

uddi-subscription *name*

Parameters

name Specifies the name of the UDDI Subscription object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In UDDI (Universal Description Discovery and Integration) Subscription configuration mode, you configure a UDDI Subscription object that provides the username and password information used to access a specified UDDI Registry (via reference to a previously created UDDI Registry object) and the key used to identify the subscribed-to registry resource.

UDDI Subscription objects provide a means to retrieve Web Service endpoint data stored in a UDDI Registry. The endpoint information can then be used by the appliance to proxy services.

Use the **cancel** or **exit** command to exit UDDI Subscription configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

---

undo

Reverts a modified object to its previously saved configuration.

Syntax

undo *object-type name*

Parameters

object-type
Specifies the type of object. For a complete list of object types, use the **show** command

name Specifies the name of the object.

Guidelines

The **undo** command reverts a modified object to its last persisted state. The persisted state is the configuration in the startup configuration.

If invoked against a modified object in the startup configuration, you will receive the following message:

```
objectType name - Configuration reverted.
```

If invoked against an new object that has yet to be saved to the startup configuration, you will receive the following message:

```
% Cannot undo new configuration
```

If invoked against a object that has not been modified, you will receive the following message:

```
% Cannot undo - configuration has not been modified
```

If invoked against a nonexistent named-object, you will receive the following message:

```
% Cannot undo last configuration change
```

If invoked against a nonexistent object type, you will receive the following message:

```
% Invalid class
```

Related Commands

show, **write memory**

Examples

- Reverts the modified `gateway-1` Multi-Protocol Gateway object to its last saved configuration.

```
# undo mpgw gateway-1
mpgw gateway-1 - Configuration reverted.
#
```

urlmap

Enters URL Map configuration mode.

Syntax

urlmap *name*

Parameters

name Specifies the name of the URL map.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

URL maps are used in the implementation of Stylesheet Refresh Policies that enable the periodic update of the stylesheet cache maintained by an XML manager.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of URL maps and Stylesheet Refresh Policies.

Related Commands

cancel, **exit**

urlrefresh

Enters URL Refresh Mode.

Syntax

urlrefresh *name*

no urlrefresh *name*

Parameters

name Specifies the name of the Stylesheet Refresh Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

With a Stylesheet Refresh Policy in place, the appliance refreshes specified style sheets at regular intervals. Update eligibility is determined by match criteria contained with URL maps assigned to the Stylesheet Refresh Policy.

When implementing a Stylesheet Refresh Policy, keep in mind that frequent updates of cached style sheets can increase the processing load on the appliance and on external network appliances. Well-designed policies provide for the more frequent updates of style sheets that are subject to change, while providing less frequent updates for stable style sheets.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of URL maps and Stylesheet Refresh Policies.

Use the **no urlrefresh** command to delete a Stylesheet Refresh Policy.

Related Commands

cancel, **exit**, **refresh stylesheet**

urlrewrite

Enters URL Rewrite Policy configuration mode.

Syntax

urlrewrite *name*

no urlrewrite *name*

Parameters

name Specifies the name of the URL Rewrite Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In this configuration, you can define rewrite rules that perform the following types of replacements:

- Rewrite an entire URL or a portion of a URL based on a URL match.
- Replace the value of the `Content-Type` header based on a URL match.
- Replace the value of an arbitrary header based on its value.
- Rewrite the body of an HTTP POST request.

Rewrite rules that are defined in the URL Rewrite Policy occur before document processing.

- Any Matching Rule must match the rewritten URL.
- Any action in the Processing Policy can change the URI that is sent to the backend server.

Use the **no urlrewrite** command to delete a URL Rewrite Policy

Related Commands

cancel, **exit**

---

user

Enters User configuration mode.

Syntax

user *name*

no user *name*

Parameters

name Specifies the name of the user.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

The **user** command is available in Global configuration mode. The **user** command enters User configuration mode. While in User configuration mode, you can create or modify User objects.

To exit the configuration mode and not apply the changes, use the **cancel** command. To apply the changes and exit the configuration mode, use the **exit** command. Using either command returns you to Global configuration mode.

Use the **no user** command to delete a user account.

To confirm the account, use the **show usernames** command.

Related Commands

cancel, **exit**, **show usernames**, **usergroup**

user-agent

Enters User Agent configuration mode.

Syntax

user-agent [*name*]

no user-agent [*name*]

Parameters

name Optionally specifies the name of the HTTP user agent.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

In the absence of an explicitly identified HTTP User Agent, the command enters User Agent Configuration mode for the `default` HTTP User Agent.

Guidelines

Use the **cancel** or **exit** command to exit HTTP User Agent Settings configuration mode and return to Global configuration mode.

Use the **no user-agent** command to delete an HTTP User Agent.

Related Commands

cancel, **exit**

user-expire-password

Forces a specified user to change the account password at the next login.

Syntax

user-expire-password *account*

Parameters

account

Identifies the target user account.

Examples

- Forces password change for the `josephb` account on the next login.

```
# user-expire-password josephb
Expire password for user 'josephb' succeeded
#
```

user-password

Changes the password of the current user.

Syntax

user-password

Examples

- Enters an interactive session to change a password.

```
# user-password
```

usergroup

Enters User Group configuration mode.

Syntax

usergroup *name*

no usergroup *name*

Parameters

name Specifies the name of the User Group.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A user group consists of a set of access privileges that are subsequently assigned to individual user accounts.

Use the **no usergroup** command to delete a User Group.

Related Commands

cancel, **exit**, **show usergroup**, **username**

vlan-sub-interface

Enters VLAN Sub-Interface configuration mode.

Syntax

Enter the configuration mode to create or modify VLAN objects
vlan-sub-interface *name*

Delete VLAN objects
no vlan-sub-interface *name*

Disable VLAN objects
disable vlan-sub-interface *name*

Note: The Admin State of Ethernet interfaces can be set from **enabled** to **disabled** while Ethernet cables are still physically connected to the appliance. Use this method of removing the appliance from the network without physically removing network cables.

Parameters

name Specifies the name of the VLAN object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

The **vlan-sub-interface** command is available in Global configuration mode. The **vlan-sub-interface** command enters Virtual LAN (VLAN) configuration mode. While in VLAN configuration mode, you can create or modify VLAN objects. Use the **no vlan-sub-interface** command to delete a VLAN object.

A VLAN object is generally used to provide a trunk between adjacent DataPower appliances. The VLAN object creates a virtual IP interface (VIP) on one of the physical Ethernet interfaces on the appliance. VLAN packets are identified by the IEEE 802.1Q tagging protocol.

You can create multiple VLAN objects on a single Ethernet interface.

To exit the configuration mode and not apply the changes, use the **cancel** command. To apply the changes and exit the configuration mode, use the **exit** command. Using either command returns you to Global configuration mode.

Related Commands

cancel, **exit**, **show vlan-interface**, **show vlan-sub-interface**

---

wasjms-server

Enters WebSphere JMS configuration mode.

Syntax

wasjms-server *name*

no wasjms-server *name*

Parameters

name Specifies the name of the WebSphere Application Server object.

The name can contain a maximum of 128 characters. For restrictions, refer to “Object name conventions” on page xxvi.

Guidelines

The **wasjms-server** command to enter WebSphere JMS configuration mode. In this mode, you can create a WebSphere Application Server (WAS) version 6.*x* default JMS message provider object

This command implements support for IBM JFAP (JetStream Formats and Protocols) on a target Multi-Protocol Gateway.

Use the **no wasjms-server** command to delete a WAS JMS default JMS message provider object.

Related Commands

cancel, **exit**

watchdog

Sets watchdog timeout values.

Syntax

watchdog *soft-timeout hard-timeout*

Parameters

soft-timeout
Specifies the soft timer value.

hard-timeout
Specifies the hard timer value.

Guidelines

The **watchdog** sets watchdog timeout values.

Watchdog timer values are set to default values. These default values should rarely, if ever, require a change.

Before changing these values, contact DataPower Customer Support.

web-application-firewall

Enters Web Application Firewall configuration mode.

Syntax

web-application-firewall *name*

no web-application-firewall *name*

Parameters

name Specifies the name of the Web Application Firewall.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** commands to exit Web Application Firewall configuration mode and return to Global configuration mode.

Use the **no web-application-firewall** command to delete a Web Application Firewall.

Related Commands

cancel, **exit**

web-mgmt

Creates a specialized HTTP server that supports WebGUI access.

Syntax

web-mgmt

web-mgmt *address port* [**on** [*timeout*]]

web-mgmt *address port* [**off**]

no web-mgmt

Parameters

address Specifies the IP address of the appliance interface that, with port, identifies the interface monitored by the WebGUI server for incoming HTTP client requests.

port Identifies the interface monitored by the WebGUI server for incoming HTTP client requests. Use an integer in the range of 0 through 65535).

on *timeout*
Sets the idle-session logout timer in seconds. Use an integer in the range of 0 to 65535. The default is 600 (10 minutes). A value of 0 disables the session timer.

off Resets the idle-session logout timer to its default timer.

Guidelines

You can create only a single WebGUI server.

The idle-session logout time specifies the time after which an idle WebGUI session expires.

When issued without an argument, enters Web Service Management configuration mode which also supports the configuration of a specialized HTTP server that supports WebGUI access.

Use the **no web-mgmt** command to delete the WebGUI server.

Related Commands

webgui-password-map

Examples

- Enters Web Management Service configuration mode.

```
# web-mgmt
Web Management Service configuration mode
#
```

- Creates a WebGUI server on the specified IP address-port pair. The idle-session logout timer defaults to 5 minutes.

```
# web-mgmt 10.10.13.31 9090
⋮
HTTP configuration update successful
#
```

- Creates a WebGUI server on the specified IP address-port pair. The idle-session logout timer is explicitly set to 10 minutes.

```
# web-mgmt 10.10.13.31 9090 on 600
:
HTTP configuration update successful
#
```

- Restores the idle-session logout timer to its default.

```
# web-mgmt 10.10.13.31 9090 off
:
HTTP configuration update successful
#
```

- Disables the idle-session logout timer.

```
# web-mgmt 10.10.13.31 9090 on 0
:
HTTP configuration update successful
#
```

- Disables the WebGUI server.

```
# no web-mgmt
:
Successfully deleted matching webgui-match
#
```

webapp-error-handling

Enters Web Application Error Handling Policy configuration mode.

Syntax

webapp-error-handling *name*

no webapp-error-handling *name*

Parameters

name Specifies the name of the Web Application Error Handling Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** commands to exit Web Application Error Handling Policy configuration mode and return to Global configuration mode.

Use the **no webapp-error-handling** command to delete a Web Application Error Handling Policy.

Related Commands

cancel, **exit**

webapp-gnvc

Enters Web Application Name Value Profile configuration mode.

Syntax

webapp-gnvc *name*

no webapp-gnvc *name*

Parameters

name Specifies the name of the Web Application Name Value Profile.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** commands to exit Web Application Name Value Profile configuration mode and return to Global configuration mode.

Use the **no webapp-gnvc** command to delete a Web Application Name Value Profile.

Related Commands

cancel, **exit**

---

webapp-request-profile

Enters Web Application Request Profile configuration mode.

Syntax

webapp-request-profile *name*

no webapp-request-profile *name*

Parameters

name Specifies the name of the Web Application Request Profile.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** commands to exit Web Application Request Profile configuration mode and return to Global configuration mode.

Use the **no webapp-request-profile** command to delete a Web Application Request Profile.

Related Commands

cancel, **exit**

---

webapp-response-profile

Enters Web Application Response Profile configuration mode.

Syntax

webapp-response-profile *name*

no webapp-response-profile *name*

Parameters

name Specifies the name of the Web Application Response Profile.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** commands to exit this configuration mode and return to Global configuration mode.

Use the **no webapp-response-profile** command to delete a Web Application Response Profile.

Related Commands

cancel, **exit**

---

webapp-session-management

Enters Session Management Policy configuration mode.

Syntax

webapp-session-management *name*

no webapp-session-management *name*

Parameters

name Specifies the name of the Web Application Session Management policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **cancel** or **exit** commands to exit this configuration mode and return to Global configuration mode.

Use the **no webapp-session-management** command to delete a Web Application Session Management policy.

Related Commands

cancel, **exit**

---

write memory

Copies the running configuration as the startup configuration.

Syntax

write memory

Guidelines

After copying the running configuration to config:///autoconfig.cfg, the appliance determines if the current startup configuration script file can be overridden by config:///autoconfig.cfg.

If it can be overridden (determined by the **save-config overwrite** command), autoconfig.cfg becomes the startup configuration.

Related Commands

save-config overwrite

Examples

- Saves the running configuration as the startup configuration.

```
# write memory
Overwrite existing autoconfig.cfg? y
#
```

- Cancels the operation.

```
# write memory
Overwrite existing autoconfig.cfg? n
#
```

---

wsgw

Enters Web Services Proxy configuration mode.

Syntax

wsgw *name*

no wsgw *name*

Parameters

name Specifies the optional name of the Web Services Proxy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the cancel or **exit** command to exit Web Services Proxy configuration mode and return to Global configuration mode.

Use the **no wsgw** command to delete a Web Services Proxy.

Related Commands

cancel, **exit**

---

wsm-agent

Enters Web Services Management Agent configuration mode.

Syntax

wsm-agent

Guidelines

The Web Services Management Agent provides manageability for Web Services by providing status, metrics, and transaction history to external management stations.

Related Commands

cancel, **exit**

wsm-endpointrewrite

Enters WS-Proxy Endpoint Rewrite configuration mode.

Syntax

wsm-endpointrewrite *name*

no wsm-endpointrewrite *name*

Parameters

name Specifies the name of the WS-Proxy Endpoint Rewrite policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no wsm-endpointrewrite** command to delete a WS-Proxy Endpoint Rewrite policy.

Related Commands

cancel, **exit**

wsm-rule

Enters Web Services Processing Rule configuration mode.

Syntax

wsm-rule *name*

no wsm-rule *name*

Parameters

name Specifies the name of the Web Services Processing Rule.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no wsm-rule** command to delete a Web Services Processing Rule.

Related Commands

cancel, **exit**

wsm-stylepolicy

Enters Web Services Processing Policy configuration mode.

Syntax

wsm-stylepolicy *name*

no wsm-stylepolicy *name*

Parameters

name Specifies the name of the Processing Policy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no wsm-stylepolicy** command to delete a Web Services Processing Policy.

Related Commands

cancel, **exit**

---

wsrr-server

Enters WSRR Server configuration mode.

Syntax

wsrr-server *name*

no wsrr-server *name*

Parameters

name Specifies the name of the WSSR server object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In WebSphere Services Repository and Registry (WSRR) Server configuration mode, provide the information necessary to locate and access a WSRR server.

Use the **no wsrr-server** command to delete a WSRR server object.

Related Commands

cancel, **exit**

---

wsrr-subscription

Enters WSRR Subscription configuration mode.

Syntax

wsrr-subscription *name*

no wsrr-subscription *name*

Parameters

wsrrSubscriptionName
Specifies the name of the WSSR subscription object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In WebSphere Services Repository and Registry (WSRR) Subscription configuration mode, provide the information necessary to specify content obtained from a specified WSRR server.

Use the **no wsrr-subscription** command to delete a WSRR subscription object.

Related Commands

cancel, **exit**, **wsrr-synchronize**

---

wsrr-synchronize

Performs a synchronization of WSRR content with the WSSR server.

Syntax

wsrr-synchronize *wsrrSubscriptionName*

Parameters

wsrrSubscriptionName
Specifies the name of a WSSR subscription object. Content previously retrieved using this subscription is immediately synchronized with the WSSR server specified by the subscription.

Related Commands

refresh-interval, **wsrr-subscription**

Examples

- Updates content previously retrieved with the `updateWS-Proxy-1` subscription.

```
# wsrr-synchronize updateWS-Proxy-1
:
#
```

---

xacml-pdp

Enters XACML Policy Decision Point configuration mode.

Syntax

xacml-pdp *name*

no xacml-pdp *name*

Parameters

name Specifies the name of the XACML PDP.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

While in XACML PDP configuration mode, specify the properties to configure an XACML PDP internal to the DataPower appliance.

Specific commands locate general and dependent XACML policies for use by the PDP, identify algorithms used to combine available XACML policies, and define XACML cache operations.

Use the **no xacml-pdp** command to delete an XACML PDP.

Use the **cancel** or **exit** command to exit XACML PDP configuration mode and return to Global configuration mode.

Related Commands

cancel, **exit**

xml parser limits

Enters XML Parser Limits configuration mode for an XML Manager.

Syntax

xml parser limits *XML-manager*

Parameters

XML-manager
Specifies the name of an XML Manager.

Guidelines

While in XML Parser Limits configuration mode, you can set configurable limits on various characteristics of XML documents (for example, input documents, style sheets, schemas) that are parsed by the appliance. These limits provide for increased security and stability to protect against denial-of-service (DoS) attacks or runaway data.

Parser limits are assigned to an XML Manager, Any service that is supported by this XML Manager inherits these limits.

Related Commands

exit

xml validate

Enables XML schema validation for a specified XML manager.

Syntax

xml validate *XML-manager matching-rule* [**attribute-rewrite** *policy*]

xml validate *XML-manager matching-rule* [**dynamic-schema** *URL*]

xml validate *XML-manager matching-rule* [**schema** *URL*]

no xml validate *XML-manager*

Parameters

XML-manager
Specifies the name of an XML manager that performs XML schema validation.

matching-rule
Specifies the name of a Matching Rule. XML documents that match any of the patterns contained within this Matching Rule are subject to manager-specific XML schema validation.

attribute-rewrite *policy*
Rewrites any schema reference in the body of the XML document with the specified URL Rewrite Policy. The rewritten schema reference usually specifies the location of a local "trusted" copy of the schema to use for document validation.

dynamic-schema *URL*
Specifies the use of a dynamically-generated schema regardless of any validation processing instructions in the XML document.

schema *URL*
Specifies the use of a specific schema regardless of any validation processing instructions in the XML document.

Guidelines

Identifies the validation methodology and XML schema to use for document validation. In the absence a keyword, validation is performed according to validation instructions, if any, that are contained in the XML document. Documents that contain no validation instructions are considered to be validated.

Do not use XML Manager-based schema validation when using processing policy-based validation filters. Simultaneous use of the two schema validations is redundant and might produce unpredictable results.

Use the **no xml validate** command to disable XML schema validation for a specified manager.

Related Commands

httpmatch, **matching**, **rewrite**, **urlmatch**, **urlrewrite**, **xmlmgr**

Examples

- Enables schema-based validation for the `mgr1` XML Manager. Validation instructions in XML documents that match `star` are rewritten in accordance with the `URL-RW-1` URL Rewrite Policy. The XML Manager uses the rewritten schema reference to validate documents.

```
# xml validate mgr1 star attribute-rewrite URL-RW-1
#
```

- Enables schema-based validation for the `mgr1` XML Manager. All XML documents that match `star` are validated against the `schema1.xsd` schema.

```
# xml validate mgr1 star schema store:///schema1.xsd
#
```

- Disables schema-based validation for the `mgr1` XML Manager.

```
# no xml validate mgr1
#
```

xmlfirewall

Enters XML Firewall Service configuration mode.

Syntax

xmlfirewall [*name*]

no xmlfirewall [*name*]

Parameters

name Optionally specifies the name of the XML Firewall.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In the absence of the optional *name* argument, the DataPower appliance provides a default name. The name takes the form `xmlfirewall`*n*.

Use the **cancel** or **exit** command to exit XML Firewall configuration mode and return to Global configuration mode.

Use the **no xmlfirewall** command to delete an XML Firewall.

Related Commands

cancel, **exit**

xml-manager

Enters XML Manager configuration mode.

Syntax

xml-manager *name*

no xml-manager *name*

Parameters

name Specifies the name of the XML manager.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

In XML Manager configuration mode, you can configure the target manager to perform a rule-based action.

Use the **no xml-manager** command to delete an XML Manager.

Related Commands

documentcache, **refresh stylesheet**, **xml parser limits**, **xml validate**, **xmlfirewall**, **xpath function map**

Examples

- Enters XML Manager configuration mode to create the `ScheduleHandler` XML Manager.

```
# xml-manager ScheduleHandler
New XML Manager configuration
#
```

- Enters XML Manager configuration mode to modify the `ScheduleHandler` XML Manager.

```
# xml-manager ScheduleHandler
Modify XML Manager configuration
#
```

- Deletes the `ScheduleHandler` XML manager.

```
# no xml-manager ScheduleHandler
xml-manager ScheduleHandler - Configuration deleted.
#
```

---

xml-mgmt

Enter XML Management Interface configuration mode or enables or disables the XML Management Interface.

Syntax

xml-mgmt

xml-mgmt *address port*

no xml-mgmt

Parameters

address Specifies an IP address that, in conjunction with port, identifies the XML Management Interface. A value of 0.0.0.0 indicates all local addresses.

port Identifies the port on the local appliance that monitors SOAP/XML management traffic. Use an integer in the range of 0 through 65535.

Guidelines

The **xml-mgmt** command has the following invocations:

- Without arguments to enter XML Management Interface configuration mode.
- With the address and port parameters to enable the XML Management Interface.
- With the **no** keyword and without arguments to disable the XML Management Interface.

When enabled, the XML Management Interface allows users to send requests to the enabled service protocols to manage the DataPower appliance.

The DataPower appliance has a single XML Management Interface. The XML Management Interface runs SSL and uses HTTP Basic Authentication (user name and password).

For information about the XML Management Interface, refer to the *IBM WebSphere DataPower SOA Appliances: Administrators Guide*.

Examples

- Enters XML Management Interface configuration mode.

```
# xml-mgmt
Modify XML Management Interface configuration
#
```

- Enables the XML Management Interface on port 1080 of the specified local address.

```
# xml-mgmt 10.10.13.7 1080
XML management: successfully started
#
```

- Disables the XML Management Interface.

```
# no xml-mgmt
XML management: successfully disabled
#
```

xpath-routing

Enters XPath Routing Map configuration mode.

Syntax

xpath-routing *name*

no xpath-routing *name*

Parameters

name Specifies the name of the XPath Routing Map.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

Use the **no xpath-routing** command to delete an XPath Routing Map.

Related Commands

cancel, **exit**

xsl cache size

Specifies the maximum number of style sheets cached by a specified XML manager.

Syntax

xsl cache size *XML-manager capacity*

Parameters

XML-manager
Specifies the name of an XML manager.

capacity
Specifies the maximum size of the cache in style sheets. Use an integer in the range of 4 through 1000000.

Guidelines

The initial cache size is set to 256 style sheets.

Related Commands

xsl checksummed cache

Examples

- Assigns a cache limit of 5000 style sheets for the `mgr1` XML manager.

```
# xsl cache size mgr1 5000
#
```

---

xsl checksummed cache

Enables or disables SHA-1-assisted caching.

Syntax

xsl checksummed cache *XML-manager*

no xsl checksummed cache *XML-manager*

Parameters

XML-manager
Specifies the name of the XML Manager.

Guidelines

SHA-1 is defined in FIPS 180-1 and published to the internet community as RFC 3174. SHA-1 takes an input message or string of less than 264 bits and computes a 160-bit output that is called a message digest. The message digest can be thought of as a digital fingerprint or signature.

With SHA-1-assisted caching enabled, style sheets are cached by both URL and SHA-1 message digest values. SHA-1-assisted caching can improve performance in environments where the same style sheet could be obtained from different locations.

With SHA-1-assisted caching disabled, style sheets are cached solely by URL. Use the **no xsl checksummed cache** command to disable SHA-1 caching.

Related Commands

xsl cache size

Examples

- Enables SHA-1-assisted caching for the `mgr1` XML Manager.

```
# xsl checksummed cache mgr1
#
```

- Disables SHA-1-assisted caching for the `mgr1` XML Manager.

```
# no xsl checksummed cache mgr1
#
```

xslconfig

Assigns a Compile Options Policy.

Syntax

xslconfig *XML-manager compileOptionsPolicyName*

no xslconfig *XML-manager*

Parameters

XML-manager
Specifies the name of the XML Manager.

compileOptionsPolicyName
Specifies the name of an existing Compile Options Policy.

Guidelines

The **xslconfig** command enables stylesheet profiling using a Compile Options Policy. Stylesheet profiling consists of associating a Compile Option Policy with an XML Manager. After associating a Compile Options Policy with an XML Manager, The XML Manager profiles all style sheets that match the policy definitions.

To create a Compile Option Policy, use the **compile-options** command with the **profile** or **debug** commands.

Use the **no xslconfig** command to remove the assignment of a Compile Options Policy from an XML manager.

Related Commands

compile-options, **debug**, **profile**, **show profile**

Examples

- Assigns the `TestCustomer-1` Compile Options Policy to the `mgr1` XML manager.

```
# xslconfig mgr1 TestCustomer-1
#
```

- Disables stylesheet profiling for the `mgr1` XML Manager.

```
# no xslconfig mgr1
#
```

xslcoproc

Enters XSL Coprocessor Service configuration mode.

Syntax

xslcoproc *name*

xslcoproc *name* **0** *port-local XML-manager* [*default-style-sheet*]

xslcoproc *name address-local port-local XML-manager* [*default-style-sheet*]

no xslcoproc *name*

Parameters

name Specifies the name of the XSL Coprocessor.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

0 Binds to all enabled appliance interfaces.

address-locals
Binds to the specified appliance interface.

port-local
Specifies the port number for the appliance interfaces. Use a value in the range of 0 to 65535.

XML-manager
Specifies the name of an XML manager.

default-style-sheet
Optionally specifies the URL of a default style sheet to use in the absence of server-generated processing instructions.

Guidelines

An XSL Coprocessor provides processing or transformation services for applications or web servers under the explicit control of the application.

You can use either of two forms (referred to as single-command and multi-command) of the **xslcoproc** command to create an XSL Coprocessor.

- The single-command form creates a basic XSL Coprocessor. This form requires the following arguments:
  - The IP address of the appliance interface
  - The port number for the appliance interface
  - The XML Manager that supports coprocessor operations

  The single command form supports the ability to assign a default style sheet.
- The multi-command form creates a named XSL Coprocessor and enters XSL Coprocessor Service configuration mode. This mode supports enhanced XSL Coprocessor creation with a series of brief single-purpose commands.

  The **ip-address**, **port**, and **xml-manager** commands are required to complete XSL Coprocessor configuration; other commands add optional enhanced coprocessor functionality.

Use the **exit** command to exit XSL Coprocessor Service configuration mode and return to Global configuration mode.

Use the **no** form of this command to delete an XSL Coprocessor.

Related Commands

xmlmgr

Examples

- Enters XSL Coprocessor Service configuration mode for the `CoProc-1` XSL Coprocessor.

```
# xslcoproc CoProc-1
XSL Coprocessor Service configuration mode
#
```

- Creates the `CoProc-1` XSL Coprocessor. Listens for requests on port 3300 of all enabled appliance ports.

```
# xslcoproc CoProc-1 0 3300 mgr1
#
```

- Creates the `CoProc-1` XSL Coprocessor. Listens for requests on port 3300 of all enabled appliance ports. Uses the default `CoProc.xsl` style sheet.

```
# xslcoproc CoProc-1 0 3300 mgr1 store:///CoProc.xsl
#
```

- Deletes the `CoProc-1` XSL Coprocessor.

```
# no xslcoproc CoProc-1
#
```

---

xslproxy

Creates an XSL Proxy.

Syntax

xslproxy *name*

xslproxy *name* **0** *port-local address-server port-server XML-manager* [*processingPolicy*]

xslproxy *name address-local port-local address-server port-server XML-manager* [*processingPolicy*]

no xslproxy *name*

Parameters

name Specifies the name of the XSL Proxy.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

0 Binds to all enabled appliance interfaces.

address-locals
Binds to the specified appliance interface.

port-local
Specifies the port number for the local interface. Use a value in the range of 0 to 65535.

address-server
Specifies the IP address of the server.

port-server
Specifies the port number for the server. Use a value in the range of 0 to 65535.

XML-manager
Specifies the name of the XML manager to manage proxy resources.

processingPolicy
Optionally specifies the name of a Processing Policy to perform transforms. The default is to use processing instructions, if any, that are in incoming XML documents.

Guidelines

You can use either of two forms (referred to as single-command and multi-command) of the **xslproxy** command to create an XSL proxy.

- The single-command form creates a basic XSL proxy and provides the ability to assign a Processing Policy.
- The multi-command form creates a basic XSL proxy and provides enhanced configuration options.

  When issued with only the name argument, the command enters XSL Proxy configuration mode. This mode supports the creation of an XSL Proxy with a series of brief single-purpose commands.

  Requires the **ip-address**, **port**, **remote-address**, and **xml-manager** commands (XSL Proxy) to complete the configuration. Other commands add optional functionality.

Use the **exit** command to exit XSL Proxy configuration mode and return to Global configuration mode.

Use the **no** form of this command to delete an XSL Proxy.

Related Commands

stylepolicy, **xmlmgr**

Examples

- Enters XSL Proxy Service configuration mode for the `Proxy-1` XSL proxy.

```
# xsl proxy Proxy-1
XSL Proxy Service configuration mode
#
```

- Creates the `catalogOrders` XSL Proxy. Monitors the local address 192.168.1.10:45000 in support of the server at 10.10.1.100:34000. Performs transforms with processing instructions in the XML document.

```
# xslproxy catalogOrders 192.168.1.10 45000
 10.10.1.100 34000 mgr1
#
```

- Creates the `catalogOrders` XSL Proxy. Monitors the local address 192.168.1.10:45000 in support of the server at 10.10.1.100:34000. Assigns the `WebQuery` Processing Policy. Performs transforms with the rules in that policy.

```
# xslproxy catalogOrders 192.168.1.10 45000
 10.10.1.100 34000 mgr1 WebQuery
#
```

- Deletes the `catalogOrders` XSL Proxy.

```
# no xslproxy catalogOrders
#
```

xslrefresh

Assigns a Stylesheet Refresh Policy to an XML Manager.

Syntax

xslrefresh *XML-manager policy*

no xslrefresh *XML-manager*

Parameters

XML-manager
Specifies the name of an XML Manager.

policy Specifies the name of a Stylesheet Refresh Policy.

Guidelines

You can assign only a single Stylesheet Refresh Policy to an XML manager. With a Stylesheet Refresh Policy, an XML Manager refreshes the specified style sheets at regular intervals. Update eligibility is determined by match criteria in URL maps that are assigned to the Stylesheet Refresh Policy.

When implementing a Stylesheet Refresh Policy, frequent updates of cached style sheets increase the processing load on the DataPower appliance and on external network appliances. A well-designed refresh policy provides for the more frequent updates of style sheets that are subject to change and provides less frequent updates for stable style sheets. Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of URL maps and Stylesheet Refresh Policies.

Use the **no xslrefresh** command to remove the assignment of a Stylesheet Refresh Policy from an XML Manager.

Related Commands

interval urlmap, **match**, **test urlrefresh**, **test urlmap**, **urlmap**, **urlrefresh**

Examples

- Assigns the `2aday` Stylesheet Refresh Policy to the `xml1` XML Manager.

```
# xslrefresh mgr1 2aday
#
```

- Removes the `2aday` Stylesheet Refresh Policy from the `xml1` XML Manager.

```
# no xslrefresh mgr1
#
```

---

zos-nss

Enters z/OS® NSS Client configuration mode.

Syntax

zos-nss *name*

no zos-nss *name*

Parameters

name Specifies the name of the z/OS NSS Client object.

The name can contain a maximum of 128 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

While in z/OS NSS Client configuration mode, you configure a z/OS NSS Client which provides the parameters necessary for authentication with SAF on a z/OS Communications Server.

Use the **no zos-nss** command to delete a z/OS NSS Client object.

Chapter 3. AAA Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in AAA (Authentication, Authorization, Audit) Policy configuration mode.

To enter this configuration mode, use the Global **aaapolicy** command. While in this mode, define the AAA Policy that specifies authentication and authorization procedures.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

actor-role-id

Defines an assumed actor or role for processing.

Syntax

actor-role-id *identifier*

Parameters

identifier

Specifies the assumed `S11:actor` or `S12:role`. Some well-known values are:

`http://schemas.xmlsoap.org/soap/actor/next`
Every one, including the intermediary and ultimate receiver, receives the message should be able to processing the Security header.

`http://www.w3.org/2003/05/soap-envelope/role/none`
No one should process the Security Header.

`http://www.w3.org/2003/05/soap-envelope/role/next`
Every one, including the intermediary and ultimate receiver, receives the message should be able to processing the Security header.

`http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver`
The message ultimate receiver can process the Security header.

No value (empty string)
(Default) The empty string (without quotes) indicates that no "actor/role" identifier is configured. With a configured actor/role, the ultimate Receiver is assumed when processing the message, and no actor/role attribute will be added when generating the WS-Security Security header. Note that there should not be more than one Security headers omitting the actor/role identifier.

`USE_MESSAGE_BASE_URI`
The actor/role identifier will be the base URL of the message, if the SOAP message is transported using HTTP, the base URI is the Request-URI of the HTTP request.

A string value
Any string to identify the actor or role of the Security header.

Guidelines

If a value is specified for the WS-Security `S11:actor` or `S12:role` identifier, the AAA action will act as the assumed actor or role when it consumes the Security headers. This setting takes effect only when the AAA policy attempts to process the incoming message before making an authorization decision.

The Post Processing phase will not use the assumed actor or role, but will use its own setting in generating the message for the next SOAP node.

---

authenticate

Specifies an authentication method and authority.

Syntax

authenticate *method* [*url*] [*host*] [*port*] [*valcred*]

authenticate custom *url* "" "" ""

authenticate client-ssl "" "" "" *valcred*

authenticate ldap "" *host port* ""

authenticate netegrity "" *host port* ""

Parameters

method Specifies the authentication method and takes one of the following values:
- **cleartrust**
- **client-ssl**
- **custom**
- **kerberos**
- **ldap**
- **netegrity**
- **radius**
- **saml-artifact**
- **saml-authen-query**
- **saml-signature**
- **tivoli**
- **token**
- **validate-signer**
- **ws-secureconversation**
- **ws-trust**
- **xmlfile**

url Specifies the location of the style sheet for authentication purposes. If the method is other than **custom**, use two double quotation mark ("") characters without any intervening space.

host Specifies the hostname or IP address of an LDAP or Netegrity authentication server. If the method is not **ldap** or **netegrity**, use two double quotation mark ("") characters without any intervening space.

port Specifies a destination port on the LDAP or Netegrity authentication server. If the method is not **ldap** or **netegrity**, use two double quotation mark ("") characters without any intervening space.

valcred Optional and meaningful only if the method is **client-ssl** to identify a

Validation Credentials List that references the certificate that is used to validate the remote SSL peer. If the method is not **client-ssl** or if the credentials that are submitted by the SSL peer are not authenticated, (other than checking the expiration date of the certificate and that it has not been revoked) use two double quotation mark (“”) characters without any intervening space.

Examples

- Specifies LDAP authentication via an LDAP server at 192.168.4.4:389.

```
# authenticate ldap "" 192.168.4.4 389 ""
#
```

- Specifies XSL-based authentication using the `AAA.xsl` style sheet.

```
# authenticate stylesheet store:///AAA.xsl "" "" ""
#
```

authorize

Specifies an authorization method and authority.

Syntax

authorize *method url host port*

authorize custom *url* “” “”

authorize ldap “” *host port*

authorize netegrity “” *host port*

Parameters

method Specifies the authorization method and takes one of the following values:
- **anyauthenticated**
- **cleartrust**
- **custom**
- **ldap**
- **netegrity**
- **passthrough**
- **saml-attr**
- **saml-authz**
- **tivoli**
- **use-authen-attr**
- **xmlfile**

url Specifies the location of the style sheet for authorization purposes. If the method is other than **custom**, use two double quotation mark (“”) characters without any intervening space.

host Specifies the hostname or IP address of the LDAP or Netegrity authorization server. If the method is not **ldap** or **netegrity**, use two double quotation mark (“”) characters without any intervening space.

port Specifies a destination port on the LDAP or Netegrity authorization server. If the method is not **ldap** or **netegrity**, use two double quotation mark (“”) characters without any intervening space.

Examples

- Specifies Tivoli authorization services.

```
# authorize tivoli "" "" ""
#
```

- Specifies XSL-based authorization using the identified style sheet.

```
# authorize stylesheet store:///Authorize.xsl "" ""
#
```

authorized-counter

Specifies a message count monitor for approved messages.

Syntax

authorized-counter *name*

Parameters

name Identifies the assigned message count monitor.

cache-allow

Enables or disables caching of AAA data for the current AAA Policy.

Syntax

cache-allow {**on** | **off**}

Parameters

on (Default) Enables caching of authentication and authorization data by the current AAA policy.

off Disables caching of authentication and authorization data by the current AAA policy.

Related Commands

cache-ttl

Examples

- Specifies that authentication and authorization data is not cached by the current AAA Policy.

```
# cache-allow off
#
```

- Restores the default state. Authentication and authorization data is cached by the current AAA Policy.

```
# cache-allow on
#
```

cache-ttl

Specifies the AAA-Policy-specific cache lifetime in seconds.

Syntax

cache-ttl *seconds*

Parameters

seconds

Specifies the number of seconds that authentication and authorization data is retained in the policy cache. The default is 3.

Guidelines

Meaningful only if caching is enabled.

Related Commands

cache-allow

Examples

- Specifies a cache lifetime of 10 seconds for the current AAA Policy.

```
# cache-ttl 10
#
```

---

dos-valve

Limits the number of times to perform the same XML processing per user request.

Syntax

dos-valve *repetitions*

Parameters

repetitions

Specifies the number of repetitions. Use a value in the range of 1 through 1000. The default is 3.

Guidelines

The **dos-valve** command limits the number of times to perform the same XML processing per user request. XML processing includes encryption, decryption, message signing, and signature validation. At this time, the AAA Policy supports this setting in the following cases:

- Identity extraction when the method is Subject DN from Certificate in the Message's signature (**extract-identity** command set to the **signer-dn** method).
- Authentication when the method is Validate the Signer Certificate for a Digitally Signed Message (the **authorize** command set to the **validate-signer** method).

When used with a value of 1, the AAA Policy extracts the first signature and its first reference from the security header and ignores all other signatures or signing references. If the security header contains more signatures or a single signature contains more signing references, these signatures and signing references are ignored. During signature verification, the processing fails if the needed signature is not part of extracted identity.

For example if **dos-valve** is 2 and the needed information to verify the signature was the third signing reference, the verification would fail. However if the information was the second signing reference, the verification would succeed.

Examples

- Limits repetitions to 5.

```
# dos-valve 5
```

extract-identity

Specifies and enables the methods to extract the identity of a service requester.

Syntax

extract-identity *http WS-SEC client-SSL SAML-attribute SAML-authenticate stylesheet url*

Parameters

http Specifies either **on** or **off** to indicate whether of not the identity of a requester is presented as HTTP basic authentication (name and password).

WS-SEC
Specifies either **on** or **off** to indicate whether of not the identity of a requester is presented as a WS-Security UserName token (name and password).

client-SSL
Specifies either **on** or **off** to indicate whether of not the identity of a requester is presented as an SSL client certificate.

SAML-attribute
Specifies either **on** or **off** to indicate whether of not the identity of a requester is presented as a SAML attribute assertion.

SAML-authenticate
Specifies either **on** or **off** to indicate whether of not the identity of a requester is presented as a SAML authentication assertion.

stylesheet
Specifies either **on** or **off** to indicate whether of not the identity of a requester is extracted via an XSL style sheet.

url Meaningful only if *stylesheet* is **on** to identify the style sheet to extract identity information from the candidate XML document. If *stylesheet* is **off**, use two double quotation mark (“”) characters, without any intervening space.

Examples

- Specifies identity extraction with HTTP basic authentication, WS-Security UserName token, and SSL certificates.

```
# extract-identity on on on off off off ""
#
```

extract-resource

Specifies and enables the methods used to extract the identity of a requested resource.

Syntax

extract-resource *target-URL original-URL namespace element operation XPath expression*

Parameters

target-URL
Specifies either **on** or **off** to indicate whether of not the resource identity is based on the URL sent by the current AAA Policy to the backend server.

original-URL
Specifies either **on** or **off** to indicate whether of not the resource identity is based on the URL received by the current AAA Policy.

namespace
Specifies either **on** or **off** to indicate whether of not the resource identity is based on the top-level element in the message being processed.

element
Specifies either **on** or **off** to indicate whether of not the resource identity is based on the local name of the request element.

operation
Specifies either **on** or **off** to indicate whether of not the resource identity is based on the HTTP Request method.

xPath Specifies either **on** or **off** to indicate whether of not the resource identity is determined by an XPath expression.

expression
Meaningful only if *xPath* is **on** to specify the operative XPath expression.

Examples

- Specifies that resource extraction is based on input and output URLs.

```
# extract-resource on on on off off off
#
```

ldap-suffix

Specifies the LDAP suffix used by the current AAA Policy.

Syntax

ldap-suffix *suffix*

Parameters

suffix Specifies the LDAP suffix.

Guidelines

The LDAP suffix (immediately preceded by a comma) is appended to the username to form a distinguished name (DN) for LDAP authentication. For example, if this string's value is `O=example.com` and the username is `Bob`, the LDAP DN will be `CN=Bob,O=example.com`.

ldap-version

Specifies the LDAP version to access the authorization server.

Syntax

ldap-suffix {**2**|**3**}

Parameters

2 (Default) Indicates LDAP version 2.

3 Indicates LDAP version 3.

log-allowed

Enables or disables the logging of successful AAA operations.

Syntax

log-allowed

no log-allowed

Guidelines

By default, successful log operations are logged as info. Use the **no log-allowed** command to disable logging.

Related Commands

log-allowed-level, **log-rejected**, **log-rejected-level**

log-allowed-level

Specifies the log priority for messages that report successful AAA operations.

Syntax

log-allowed-level *priority*

Parameters

priority

Specifies the log priority of messages that report successful AAA operations, and takes one of the following values:

- `emergency`
- `alert`
- `critical`
- `error`
- `warning`
- `notice`
- `info` (Default)
- `debug`

Guidelines

Meaningful only if logging of successful AAA operations is enabled.

Related Commands

log-allowed, **log-rejected**, **log-rejected-level**

log-rejected

Enables or disables the logging of unsuccessful AAA operations.

Syntax

log-rejected

no log-rejected

Guidelines

By default, successful log operations are logged as warning. Use the **no log-rejected** command to disable unsuccessful AAA operations.

Related Commands

log-allowed, **log-allowed-level**, **log-rejected-level**

---

log-rejected-level

Specifies the log priority for messages that report successful AAA operations.

Syntax

log-rejected-level *priority*

Parameters

priority

Specifies the log priority of messages that report unsuccessful AAA operations, and takes one of the following values:

- `emergency`
- `alert`
- `critical`
- `error`
- `warning` (Default)
- `notice`
- `info`
- `debug`

Guidelines

Meaningful only if logging of unsuccessful AAA operations is enabled.

Related Commands

log-allowed, **low-allowed-level**, **log-rejected**

---

map-credentials

Specifies the method used to map authentication/authorization credentials.

Syntax

map-credentials custom *custom-URL*

map-credentials xmlfile *XML-file-URL*

map-credentials XPath *expression*

Parameters

custom *custom-URL*
Specifies the location of the style sheet.

xmlfile *XML-file-URL*
Specifies the location of the XML file.

XPath *expression*
Specifies the operative XPath expression.

Examples

- Specifies that credentials mapping uses the `mapCreds.xsl` style sheet.

```
# map-credentials custom local:///mapCreds.xsl
#
```

---

map-resource

Specifies the method used to map resources.

Syntax

map-resource custom *custom-URL*

map-resource xmlfile *XML-file-URL*

map-resource XPath *expression*

Parameters

custom *custom-URL*
Specifies the location of the style sheet.

xmlfile *XML-file-URL*
Specifies the location of the XML file.

XPath *expression*
Specifies the operative XPath expression.

Examples

- Specifies that resources mapping uses the `mapResource.xsl` style sheet.

```
# map-resource custom local:///mapResource.xsl
#
```

---

namespace-mapping

Adds XML namespace data to an AAA Policy.

Syntax

namespace-mapping *prefix uri*

Parameters

prefix Specifies the namespace prefix.

uri Specifies the namespace location.

Examples

- Specifies the schema for SOAP 1.1 envelope namespace.

```
# namespace-mapping SOAP
 http://schemas.xmlsoap.org/soap/envelope/
#
```

ping-identity-compatibility

Enables or disables compatibility with a PingFederate identity server.

Syntax

ping-identity-compatibility

no ping-identity-compatibility

Guidelines

By default, compatibility is disabled. Use the **no ping-identity-compatibility** command to disable compatibility.

Examples

- Enables PingFederate compatibility.

```
# ping-identity-compatibility
#
```

post-process

Enables or disables a style sheet-based postprocessing action.

Syntax

post-process {**on** | **off**} *URL*

Parameters

on | off
Enables or disables postprocessing.

on Enables postprocessing

off Disables postprocessing

URL Specifies the URL of the style sheet that performs post processing activities.

Examples

- Enables postprocessing with the specified style sheet.

```
# post-process on store:///PP.xsl
#
```

- Disables postprocessing with the specified style sheet.

```
# post-process off store:///PP.xsl
#
```

rejected-counter

Specifies a message count monitor for rejected messages.

Syntax

rejected-counter *name*

Parameters

name Identifies the assigned message count monitor.

Examples

- Associates the `AAA-Reject` message count monitor with the current AAA Policy.

```
# rejected-counter AAA-Reject
#
```

saml-artifact-mapping

Specifies the location of the SAML artifact-mapping file

Syntax

saml-artifact-mapping *url*

Parameters

url Specifies a local or remote URL that specifies the file location.

Guidelines

The SAML artifact-mapping file provides a mapping of SAML artifact source IDs to artifact retrieval endpoints. This is required only if artifacts will be retrieved from multiple endpoints, and the source ID for these endpoints are encoded in the artifact itself, as per the SAML spec. If there is only one artifact retrieval URL, it can be specified by the SAML artifact responder URL in the authentication phase

Examples

- Locates the SAML artifact-mapping file.

```
# saml-artifact-mapping local:///artifactTO.xml
#
```

saml-attribute

Specifies the namespace URI, local-name, and expected value of a SAML attribute.

Syntax

saml-attribute *uri name value*

Parameters

uri Provides the namespace URI for the attribute. The Namespace URI must match to match on a name. If blank, the null namespace is used. An example entry would be the following namespace:

```
http://www.examples.com
```

This entry would match messages with the following attribute:

```
<Attribute AttributeName="cats" AttributeNamespace="http://www.example.com">
  <AttributeValue>Winchester</AttributeValue>
<Attribute>
```

name Provides the local name of the attribute. For example, `cats` would match messages with the following attribute:

```
<Attribute AttributeName="cats" AttributeNamespace="http://www.example.com">
  <AttributeValue>Winchester</AttributeValue>
<Attribute>
```

value Provides the value given for the attribute with the corresponding name. For example, `Winchester` would match the following attribute:

```
<Attribute AttributeName="cats" AttributeNamespace="http://www.example.com">
  <AttributeValue>Winchester</AttributeValue>
<Attribute>
```

saml-name-qualifier

Specifies the value of the SAML `NameQualifier` attribute.

Syntax

saml-name-qualifier *name*

Parameters

name Identifies the SAML `NameQualifier` attribute value.

saml-server-name

Specifies the value of the Server field for SAML assertions.

Syntax

saml-server-name *name*

Parameters

name Identifies the `Server` field value.

saml-sign-alg

Specifies the algorithm to sign SAML messages.

Syntax

saml-sign-alg *algorithm*

Parameters

algorithm

Specify one of the following keywords:

dsa `http://www.w3.org/2000/09/xmldsig#dsa-sha1`

rsa (Default) `http://www.w3.org/2000/09/xmldsig#rsa-sha1`

rsa-md5
`http://www.w3.org/2001/04/xmldsig-more#rsa-md5`

rsa-ripemd160
http://www.w3.org/2001/04/xmldsig-more/rsa-ripemd160

rsa-sha256
http://www.w3.org/2001/04/xmldsig-more#rsa-sha256

rsa-sha384
http://www.w3.org/2001/04/xmldsig-more#rsa-sha384

rsa-sha512
http://www.w3.org/2001/04/xmldsig-more#rsa-sha512

Guidelines

If the SAML message that is generated for this policy will be digitally signed, use the **saml-sign-alg** command to specify the SignatureMethod for the signing algorithm.

saml-sign-cert

Specifies the public certificate associated with the key used by the current AAA Policy to sign SAML messages.

Syntax

saml-sign-cert *name*

Parameters

name Identifies the crypto certificate object.

Related Commands

saml-sign-key

saml-sign-hash

Specifies the algorithm to calculate the message digest for signing.

Syntax

saml-sign-hash *algorithm*

Parameters

algorithm
Specify one of the following keywords:

md5 http://www.w3.org/2001/04/xmldsig-more#md5

ripemd160
http://www.w3.org/2001/04/xmlenc#ripemd160

sha1 (Default) http://www.w3.org/2000/09/xmldsig#sha1

sha224
http://www.w3.org/2001/04/xmldsig-more#sha224

sha256
http://www.w3.org/2001/04/xmlenc#sha256

sha384
http://www.w3.org/2001/04/xmldsig-more#sha384

sha512
`http://www.w3.org/2001/04/xmlenc#sha512`

Guidelines

If the SAML message that is generated for this policy will be digitally signed, use the **saml-sign-hash** command to specify the algorithm to calculate the message digest for signing.

saml-sign-key

Specifies the key used by the current AAA Policy to sign SAML messages.

Syntax

saml-sign-key *name*

Parameters

name Identifies the crypto key object for SAML signatures.

Related Commands

saml-sign-cert

saml-valcred

Specifies the Validation Credentials Set used by the current AAA Policy for SAML signature verification.

Syntax

saml-valcred *name*

Parameters

name Identifies the Validation Credentials Set to use for signature verification.

saml2-metadata

Specifies the location of the SAML 2.0 metadata file.

Syntax

saml2-metadata *URL*

Parameters

URL Specifies a local or remote URL that specifies the file location.

Guidelines

The SAML metadata file contains metadata used in SAML 2.0 protocol message exchanges. This metadata is used to identify Identity Provider endpoints, and certificates needed to secure SAML 2.0 message exchanges.

The file must have a `<md:EntitiesDescriptor>` top-level element, with one or more `<EntityDescriptor>` child elements (one per Identity Provider).

Examples

- Locates the metadata file.

```
# saml2-metadata local:///policy-1.metadata
#
```

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies the name of the SSL Proxy Profile.

transaction-priority

Assigns a transactional priority to the user.

Syntax

transaction-priority *name priority authorize*

Parameters

name Specifies the name of the output credential.

priority
Indicates the priority to assign for scheduling or for resource allocation. Use one of the following values:

low Receives below normal priority.

normal
(Default) Receives normal priority.

high Receives above normal priority.

authorize
Indicates whether to require authorization. Use one of the following values:

on Requires authorization.

off (Default) Does not require authorization.

wstrust-encrypt-key

Specifies the certificate whose public key will be used to encrypt the WS-Trust secret.

Syntax

wstrust-encrypt-key *name*

no wstrust-encrypt-key *name*

Parameters

name Identifies the certificate object.

Guidelines

Use the **no wstrust-encrypt-key** command to remove the certificate assignment from the current AAA Policy.

Chapter 4. Access Control List configuration mode

This chapter provides an alphabetic listing of commands that are available in Access Control List (ACL) configuration mode.

To enter this configuration mode, use the Global **acl** command. While in this mode, create an ACL. An ACL consists of a sequence of `allow` and `deny` clauses. Each clause identifies an IP address or range of addresses that allow or that deny access to a service.

An ACL is associated with a specific DataPower service. An ACL grants access to the service to only addresses that are defined by the **allow** command. All other addresses are denied access.

Candidate addresses are evaluated sequentially against each `allow` and `deny` clause in the ACL. A candidate address is denied or granted access in accordance with the first clause that matches. Consequently, the order of `allow` and `deny` clauses in the ACL is vital.

For example, the following ACL fails its intended purpose. The address range that is specified by the `deny` clause (192.168.14.224 through 192.168.14.255) is granted access before the `allow` clause.

```
allow 192.168.14.0/24
deny 192.168.14.0/27
```

However, as shown in the following example, reversing the sequence of the clauses achieves the desired effect.

```
deny 192.168.14.0/27
allow 192.168.14.0/24
```

An ACL that contains only `deny` clauses effectively disables a service by denying access to all addresses. To complete an ACL, include the `allow any` clause. This clause ensures that addresses that are not explicitly denied access are granted access.

The following example denies access to two ranges of addresses and allows access to all other addresses.

```
deny 10.10.10.0/24
deny 172.16.0.0/16
allow any
```

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in ACL configuration mode.

allow

Identifies IP addresses to grant access.

Syntax

allow *address/netmask*

allow any

Parameters

address/netmask
Defines a range of IP addresses. Specify the IP address in dotted decimal format. Specify the net mask in CIDR (slash) format or dotted decimal format. CIDR format is an integer that specifies the length of the network portion of the address.

any Specifies all IP addresses.

Guidelines

The **allow** command defines an `allow` clause for the ACL. This clause identifies which IP addresses to grant access. If the ACL contains only `deny` clauses, the last clause in the ACL must be the `allow any` clause.

Related Commands

deny

Examples

- Enters ACL configuration mode for the `Restricted` ACL. Limits access to IP addresses 10.10.10.224 through 10.10.10.255, 192.168.14.1, and 10.10.100.1. All other IP addresses are denied access.

```
# acl Restricted
ACL configuration mode
# allow 10.10.10.0/27
# allow 192.168.14.1/32
# allow 10.10.100.1/32
# exit
#
```

---

deny

Identifies IP addresses to deny access.

Syntax

deny *address/netmask*

deny any

Parameters

address/netmask
Defines a range of IP addresses. Specify the IP address in dotted decimal format. Specify the net mask in CIDR (slash) format or dotted decimal format. CIDR format is an integer that specifies the length of the network portion of the address.

any Specifies all IP addresses.

Guidelines

The **deny** command defines an `deny` clause for the ACL. This clause identifies which IP addresses to deny access. If the ACL contains only `deny` clauses, the last clause in the ACL must be the `allow any` clause.

Related Commands

allow

Examples

- Enters ACL configuration mode for the `Public` ACL. Denies access to IP addresses 10.0.0.0 through 10.255.255.255 and to addresses 192.168.0.0 through 192.168.255.255. All other IP addresses are granted access.

```
# acl Public
ACL configuration mode
# deny 10.0.0.0/8
# deny 192.168.0.0/16
# allow any
# exit
#
```

Chapter 5. Application Domain configuration mode

This chapter provides an alphabetic listing of commands that are available in Application Domain configuration mode.

To enter this configuration mode, use the Global **domain** command. The Global command creates the specified application domain if it does not exist. While in this mode, define properties for the application domain.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

config-mode

Specifies the Application Domain configuration mode.

Syntax

config-mode [**local**] [**import**]

Parameters

local (Default) Specifies local, file-based configuration. Domain configuration is defined in a local configuration file.

import
Specifies remote configuration. Domain configuration is defined in a remote resource.

Guidelines

If **config-mode** is set to **import**, you must specify both the location and type of the remote configuration resource with the **import-url** and **import-format** commands. Also if **config-mode** is set to **import**, you can specify a deployment policy to preprocess the package with the **deployment-policy** command.

Related Commands

deployment-policy, **import-format**, **import-url**

Examples

- Creates the `Randall` application domain. Identifies a remote configuration resource at the specified URL.

```
# domain Randall
New Application Domain configuration
# config-mode import
# import-url
 http://www.datapower.com/configs/AppDomainTest.cfg
# import-format xml
#
```

deployment-policy

Specifies the deployment policy that preprocesses the configuration package.

Syntax

deployment-policy *name*

Parameters

name Specifies the name of an existing Deployment Policy object.

Guidelines

The **deployment-policy** command specifies the name of the Deployment Policy object that preprocesses the configuration package. To create a Deployment Policy object, use the Global **deployment-policy** command.

Related Commands

deployment-policy

Examples

- Creates the `Rothington` application domain. Identifies a remote configuration resource at the specified URL, and applies the existing `GeneralDeploy` Deployment Policy to the import package.

```
# domain Rothington
New Application Domain configuration
# config-mode import
# import-url
 http://www.datapower.com/configs/AppDomainTest.cfg
# import-format xml
# deployment-policy GeneralDeploy
#
```

domain-user (deprecated)

This command is deprecated. To provide the same behavior, use the **domain** command from the User configuration mode. In the absence of an RBM policy, defines the accessible domains for all user interfaces.

Syntax

domain-user *user*

Parameters

user Specifies the name of a valid locally configured user.

Guidelines

The **domain-user** command applies to command line access only. This command does not affect WebGUI access.

Related Commands

file-permissions

Examples

- Modifies the `test` application domain to add `gharrison` to the list of users who can access this domain.

```
# domain test
Modify Application Domain configuration
# domain-user gharrison
# exit
#
```

file-monitoring

Establishes the level of monitoring applied to files stored in the local: domain directory.

Syntax

file-monitoring *type*[+*type*]

Parameters

type Can be `audit` or `log`. The type audit causes the system to place entries in the audit log whenever a file is added, deleted or altered. The type log causes the system to propagate a log message whenever a file is added, deleted or altered.

Guidelines

File monitoring may be useful for administrative tracking and control of files used in the application domain.

Related Commands

file-permissions

Examples

- Modifies the `test` application domain and enables both auditing and logging of changes to files.

```
# domain test
Modify Application Domain configuration
# file-monitoring audit+log
# exit
#
```

file-permissions

Establishes user access permissions for files stored in the local: domain directory.

Syntax

file-permissions *type*[+*type*...]

Parameters

type Can be **CopyFrom**, **CopyTo**, **Delete**, **Display**, or **Exec**.

Guidelines

File permissions can be useful for administrative tracking and control of files used in the application domain.

If access controls are set both with file permissions and in the RBM system, the user will get the least privilege allowed. For example, if the file permission allow

only Display but RBM allows a user to Display and Delete, the user will only be able to Display the contents of files. On the other hand, if the permissions allow both Display and Delete but RBM allows only Display, the user will only be able to Display the contents of files.

Examples

- Modifies the `test` application domain to grant permissions to copy from files, to display contents of files, and to execute files as scripts.

```
# domain test
Modify Application Domain configuration
# file-permissions CopyFrom+Display+Exec
# exit
#
```

import-format

Specifies the format of the remote configuration file.

Syntax

import-format {**xml** | **zip**}

Parameters

xml Specifies an XML file.

zip (Default) Specifies a ZIP file.

Guidelines

If **config-mode** is set to **import**, you must specify both the location and type of the remote configuration resource with the **import-url** and **import-format** commands.

Related Commands

config-mode, **import-url**

Examples

- Creates the `test` application domain. Identifies a remote configuration resource at the specified URL.

```
# domain test
New Application Domain configuration
# config-mode import
# import-url
 http://www.datapower.com/configs/AppDomainTest.cfg
# import-format xml
# exit
#
```

import-url

Specifies the location of a remote domain-specific configuration resource.

Syntax

import-url *URL*

Parameters

URL Specifies the location of the remote configuration file.

Guidelines

If **config-mode** is set to **import**, you must specify both the location and type of the remote configuration resource with the **import-url** and **import-format** commands.

Related Commands

config-mode, **import-format**

Examples

- Creates the `test` application domain. Identifies a remote configuration resource at the specified URL.

```
# domain test
New Application Domain configuration
# config-mode import
# import-url
 http://www.datapower.com/configs/AppDomainTest.cfg
# import-format xml
# exit
#
```

local-ip-rewrite

Indicates whether to rewrite the local IP address during an import.

Syntax

local-ip-rewrite {**on** | **off**}

Parameters

on (Default) Rewrites the IP address in the package to match the equivalent interfaces on the appliance during an import. In other words, a service bound to `eth1` in the package is rewritten to bind to `eth1` during the import.

off Retains the IP addresses from the package.

Guidelines

If **config-mode** is set to **import**, use the **local-ip-rewrite** command to indicate whether to rewrite the local IP address during an import.

Related Commands

config-mode

maxchkpoints

Sets the configuration checkpoint limit.

Syntax

maxchkpoints *count*

Parameters

count Specifies the maximum number of configuration checkpoints to allow. Use an integer in the range of 1 through 5. The default is 3.

Related Commands

config-mode, **import-format**, **import-url**

---

reset domain

Deletes the currently running configuration of the domain and returns the domain to its initial state.

Syntax

reset domain [*domain*]

Parameters

domain Identifies the domain to be returned to its initial state.

Guidelines

The **reset domain** command deletes the currently running configuration of the domain and returns the domain to its initial state.

- When in an application domain, this command deletes the currently running configuration of the domain and returns the domain to its initial state.
- When in the `default` domain, this command deletes the currently running configuration of the `default` domain and returns the `default` domain to its initial state.
-
-

The primary difference between the **no domain** command and the **reset domain** command removes only the currently running configuration for a domain, and does not delete the physical domain. A possible use case would be to import new configuration settings into the domain without deleting and recreating the domain. The **reset domain** does not delete the local directories of the domain. The **no domain** command deletes the physical domain.

Related Commands

boot config, **save-config overwrite**, **write memory**

Examples

- Resets the `Test` domain while in the `default` domain.

```
# reset domain Test
Resetting 'Test' will delete all services configured within the domain!
Do you want to continue? [y/n]:y
Domain reset successfully.
#
```

- Resets the `Test` domain while in the `Test` domain.

```
[Test]# reset domain
reset domain
Resetting 'Test' will delete all services configured within the domain!
Do you want to continue? [y/n]:y
Domain reset successfully.
[Test]#
```

visible-domain

Specifies other application domains that are visible to this domain.

Syntax

visible-domain *domain*

Parameters

domain Specifies the name of a valid application domain on the current system.

Guidelines

The **visible-domain** command specifies other application domains that are visible to this domain. All files stored in visible domains are read-only to this domain. No other objects are available.

Note: References to visible domains are explicit, not bidirectional. If `domainA` is made visible to `domainB`, `domainB` cannot see `domainA`. In this case, you cannot make `domainA` visible to `domainB`. References to visible domains cannot be circular.

Examples

- Modifies the `test` application domain to make files in the `StandardApps` application domain visible to this domain.

```
# domain test
Modify Application Domain configuration
# visible-domain StandardApps
# exit
#
```

Chapter 6. Application Security Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Application Security Policy configuration mode.

To enter this configuration mode, use the Global **application-security-policy** command. If the policy does not exist, the global command creates the Application Security Policy. While in this Mode, define the parameters for the Application Security Policy object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Application Security Policy configuration mode.

error-match

Establishes one or more Error Policy Maps for this Security Policy.

Syntax

error-match *match-rule processing-rule*

no error-match

Parameters

match-rule
Specifies the name of an existing Match Rule. Use the Global **match** command to create a new Match Rule.

processing-rule
Specifies the name of an existing Processing Rule. Use the Global **rule** command to create a new Processing Rule.

Guidelines

Any error condition that matches the configured Match Rule will be handled by the corresponding Processing Rule. The first Match Rule in the Map that matches will handle the error. A Security Policy may have no Error Maps, in which case the Error Map set at the Firewall level (if any) will apply.

Note that either the Web Request Profile or the Web Response Profile invoked to handle transactions may override this Error Map.

Use the **no error-match** command to remove the entire Error Map.

Related Commands

error-policy (Web Application Firewall), **error-policy-override** (Web Request Profile), **error-policy-override** (Web Response Profile), **match** (Global), **rule** (Global)

Examples

- Creates three entries in the Error Map, in the order in which they were created.

```
# error-match SvrRedir portal-redir-errors
# error-match SvrErr portal-svr-errors
# error-match AllErr portal-default-errors
```

- Empties the Error Map, effectively eliminating all custom error handling from the security policy.

```
# no error-match
```

request-match

Establishes one or more Web Request Maps for this Security Policy.

Syntax

request-match *rule profile*

no request-match

Parameters

rule Specifies the name of an existing Match Rule. Use the Global **match** command to create a new Match Rule.

profile Specifies the name of an existing Web Request Profile. Use the Global **webapp-request-profile** command to create a new Web Request Profile.

Guidelines

Any client request that matches a configured Match Rule will be handled by the corresponding Web Request Profile. The first Match Rule in the Map that matches will handle the request. A Security Policy must have at least one entry in the Web Request Map.

Use the **no request-match** command to remove the entire Web Request Map.

Related Commands

webapp-request-profile (Global), **match** (Global)

Examples

- Creates three entries in the Web Request Map, in the order in which they were created.

```
# request-match PortalA portal-a-req
# request-match PortalB portal-b-req
# request-match All portal-default-req
```

- Empties the Request Map. At least one entry is required to create a Security Policy.

```
# no request-match
```

response-match

Establishes one or more Web Response Maps for this Security Policy.

Syntax

request-match *rule profile*

Parameters

rule Specifies the name of an existing Match Rule. Use the Global **match** command to create a new Match Rule.

profile Specifies the name of an existing Web Response Profile. Use the Global **webapp-response-profile** command to create a new Web Response Profile.

Guidelines

Any server response that matches a configured Match Rule will be handled by the corresponding Web Response Profile. The first Match Rule in the Map that matches will handle the response. A Security Policy must have at least one entry in the Web Response Map.

Use the **no request-match** command to remove the entire Web Response Map.

Related Commands

webapp-request-profile (Global), **match** (Global)

Examples

- Creates three entries in the Web Response Map, in the order in which they were created.

```
# response-match PortalA portal-a-resp
# request-match PortalB portal-b-resp
# request-match All portal-default-resp
```

- Empties the Response Map. At least one entry is required to create a Security Policy.

```
# no request-match
```

Chapter 7. Compact Flash configuration mode (Type 9235)

This chapter provides an alphabetic listing of commands that are available in Compact Flash configuration mode. To enter this configuration mode, use the Global **compact-flash** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Compact Flash configuration mode.

directory

Specifies the name of the directory.

Syntax

directory *name*

Parameters

name Specifies the name of the subdirectory.

Guidelines

The **directory** command specifies the directory under which to make the files on the compact flash available in the local: and logstore: directories in each application domain.

Examples

- Makes the files on the compact flash storage card accessible in the `local:///flash` and `logstore:///flash` directories.

```
# compact-flash cf0
Compact Flash configuration mode
# directory flash
#
```

read-only

Sets the files on the compact flash to read-only access.

Syntax

read-only

no read-only

Guidelines

The **read-only** command sets the files on the compact flash to read-only access. The default is read-write.

Examples

- Makes the file system read-only.

```
# compact-flash cf0
Compact Flash configuration mode
# read-only
#
```

- Makes the file system read-write, the default state.

```
# compact-flash cf0
Compact Flash configuration mode
# no read-only
#
```

Chapter 8. Compile Options Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Compile Options Policy configuration mode.

To enter this configuration mode, use the Global **compile-options** command. While in this configuration mode, define the Compile Options Policy. This policy is intended for use in test or debug environments only. In these environments, this policy measures the time that is spent processing a specified set of style sheets.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Compile Options Policy configuration mode.

allow-soap-enc-array

Designates the set of schemas that accept most uses of elements with `xsi:type="SOAP-ENC:Array"`.

Syntax

allow-soap-enc-array *map*

Parameters

map Identifies the URL map that defines the set of schemas that specifically allow the `xsi:type="SOAP-ENC:Array"` rule.

Guidelines

The **allow-soap-enc-array** command designates a set of schemas that will accept most uses of elements with `xsi:type="SOAP-ENC:Array"` consistent with SOAP 1.1 Section 5, even when these attributes violate the XML Schema specification.

Normally the `xsi:type` attribute must name a type equal to or derived from the actual type of the element. For schemas compiled with this option, `xsi:type` is accepted specifically for the SOAP 1.1 Encoding Array complex type if the element's type is derived from `SOAP-ENC:Array` — the opposite of the normal allowable case.

debug

Identifies set of style sheets to profile in debug mode.

Syntax

debug *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

A Compile Options Policy can contain multiple**profile** and **debug** commands.

A candidate URL is subject to debug profiling if it matches any of the match criteria specified in the URL Map.

Refer toAppendix D, “Compile Options Policy configuration,” on page 1115for procedural details regarding the creation and implementation of profiling policies.

Related Commands

profile, **show profile**

Examples

- Assigns the `URLMap-Compile-1` URL map to the current Compile Options Policy. All candidate style sheets that match the map criteria are profiled in debug mode.

```
# debug URLMap-Compile-1
#
```

disallow-xg4

Specifies a set of style sheets that will never use XG4 hardware acceleration.

Syntax

disallow-xg4 *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

XG4 will be used only when the first action in a processing policy is validation. Even when hardware resources are available, XG4 hardware acceleration will never be used.

Related Commands

prefer-xg4

Examples

- Assigns the `neverXG4` URL map to the current policy.

```
# disallow-xg4 neverXG4
#
```

minesc

Disables output escaping for a specified set of style sheets.

Syntax

minesc *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

By default, output of style sheets is escaped during the transformation process; for example, when handling non-English character sets that requires minimal escaping.

Examples

- Assigns the `noEscape` URL map to the current policy.

```
# minesc noEscape
#
```

---

prefer-xg4

Specifies a set of style sheets that must use XG4 hardware acceleration when possible.

Syntax

prefer-xg4 *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

XG4 will be used only when the first action in a processing policy is validation. When hardware resources are available, XG4 hardware acceleration will be used.

Related Commands

disallow-xg4

Examples

- Assigns the `mustXG4` URL map to the current policy.

```
# prefer-xg4 mustXG4
#
```

---

profile

Identifies a set of style sheets to be profiled.

Syntax

profile *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

A Compile Options Policy may contain multiple**profile** and **debug** commands.

A candidate URL is subject to standard profiling if it matches any of the match criteria specified in the URL Map.

Refer to Appendix D, "Compile Options Policy configuration," on page 1115 for procedural details regarding the creation and implementation of profiling policies.

Related Commands

debug, show profile

Examples

- Assigns the `URLMap-Compile-2` URL map to the Compile Options Policy.

```
# profile URLMap-Compile-2
#
```

stack-size

Specifies the maximum memory in bytes that are available to process a style sheet.

Syntax

stack-size *bytes*

Parameters

bytes Specifies the maximum memory usage. Use an integer in the range of 10240 through 104857600.

Examples

- Sets a maximum of 10 megabytes to allow for the compilation for style sheets.

```
# stack-size 10485760
#
```

stream

Specifies a set of style sheets whose output is streamed.

Syntax

stream *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

With streaming enabled, transformation of a document begins before the input is fully parsed. Not all style sheets can be streamed. If a style sheet cannot be streamed, an error is issued and document processing is abandoned.

Related Commands

try-stream

Examples

- Assigns the `fastPath` URL map to the current policy.

```
# stream fastPath
#
```

strict

Controls strict XSLT error-checking.

Syntax

strict

Guidelines

Use this command to toggle between enabling and disabling strict XSLT error-checking.

By default, the Compile Options Policy disables strict XSLT error-checking.

Non-strict operation attempts to recover from certain common XSLT errors such as use of undeclared variables or templates.

Examples

- Enables and subsequently disables strict XSLT error-checking.

```
# strict
⋮
# strict
#
```

try-stream

Specifies a set of style sheets whose output is conditionally streamed.

Syntax

try-stream *map*

Parameters

map Specifies the name of a URL map that defines the set of style sheets.

Guidelines

With conditional streaming enabled, transformation of a document begins before the input is fully parsed. Not all style sheets can be streamed. If a style sheet cannot be conditionally streamed, a warning is issued, and the document will be reprocessed in standard (non-streaming) mode.

Related Commands

stream

Examples

- Assigns the `conditionalPath` URL map to the current policy.

```
# try-stream conditionalPath
#
```

validate-soap-enc-array

Designates the set of schemas to perform extra validation on elements of type `SOAP-ENC:Array`.

Syntax

validate-soap-enc-array *map*

Parameters

map Identifies the URL map that defines the set of schemas that perform extra validation on elements of type `SOAP-ENC:Array` rule.

Guidelines

The **allow-soap-enc-array** command designates a set of schemas that will perform extra validation on elements of type `SOAP-ENC:Array`, following the encoding rules in SOAP 1.1 Section 5.

Using only rules for the XML Schema, any content is allowed in a SOAP array. The rules for SOAP 1.1 further require that child elements of array elements have the type specified in the array's `SOAP-ENC:arrayType` attribute, and allows them to have a `SOAP-ENC:position` attribute even if the array element type does not explicitly allow it.

wildcard-ignore-xsi-type

Designates a set of schemas where wildcards (`xs:any` elements) only validate children by element name.

Syntax

wildcard-ignore-xsi-type *map*

Parameters

map Identifies the URL map that defines the set of schemas set of schemas where wildcards (`xs:any` elements) only validate children by element name.

Guidelines

The **allow-soap-enc-array** command designates a set of schemas set of schemas set of schemas where wildcards (`xs:any` elements) only validate children by element name.

The XML Schema specification requires that, if a wildcard matches an element but that element does not have an element declaration, the element is instead validated according to an `xsi:type` attribute on it. This command ignores those `xsi:type` attributes and should be used for cases such as SOAP envelope validation where a further validation step will validate the contents matching the wildcard, possibly using the SOAP 1.1 encoding rules.

wsdl-strict-soap-version

Determine whether to strictly follow the SOAP binding in the WSDL.

Syntax

wsdl-strict-soap-version {**on** | **off**}

Parameters

on Follows the version of the SOAP binding in the WSDL. Allows only messages that are bound to SOAP 1.2 to appear in SOAP 1.2 envelopes, and allows only messages that are bound to SOAP 1.1 to appear in SOAP 1.1 envelopes.

off (Default) Does not follow the version of the SOAP binding in the WSDL.

Examples

- Enables and subsequently disables strict WSDL version checking.

```
# wsdl-strict-soap-version on
:
# wsdl-strict-soap-version off
#
```

wsdl-validate-body

Specifies validation behavior for the `soap:Body` of a message.

Syntax

wsdl-validate-body {**strict** | **lax** | **strict**}

Parameters

skip Disables validation of the body.

lax Forces validation of the bodies that match the WSDL definition.

strict Validates all bodies, which allows only messages that match the WSDL description.

Guidelines

By default, strict validation is applied to `soap:Body`. Use this command to relax these restrictions, thus allowing more messages to pass validation.

Related Commands

wsdl-validate-faults, **wsdl-validate-headers**

Examples

- Sets the `soap:Body` validation strict. Only messages that match the WSDL description in the `soap:Body` are allowed.

```
# wsdl-validate-body strict
#
```

wsdl-validate-faults

Specifies validation behavior for the fault detail.

Syntax

wsdl-validate-faults {**skip** | **lax** | **strict**}

Parameters

skip Disables validation of the fault detail.

lax Forces validation of the fault details that match the WSDL definition.

strict (Default) Validates all fault details, which allows only messages that match the WSDL description.

Guidelines

By default, strict validation is applied to SOAP Fault messages. Use this command to relax these restrictions, thus allowing more messages to pass validation.

Related Commands

wsdl-validate-body, **wsdl-validate-headers**

Examples

- Sets the fault detail validation strict. Only messages that match the WSDL description in the fault detail are allowed.

```
# wsdl-validate-faults strict
#
```

---

wsdl-validate-headers

Specifies validation behavior of the `soap:Header` of the message.

Syntax

wsdl-validate-headers {**skip** | **lax** | **strict**}

Parameters

skip Disables validation of the headers.

lax Forces validation of the headers that match the WSDL definition.

strict Validates all headers, which allows only messages that match the WSDL description.

Guidelines

By default, lax validation is applied to the `soap:Header` of the message. Use this command to relax these restrictions, thus allowing more messages to pass validation.

Related Commands

wsdl-validate-body, **wsdl-validate-faults**

Examples

- Sets the `soap:Header` validation strict. Only messages that match the WSDL description in the `soap:Header` are allowed.

```
# wsdl-validate-headers strict
#
```

wsdl-wrapped-faults

Controls compatibility with RPC-style wrappers.

Syntax

wsdl-wrapped-faults

Guidelines

By default, the Compile Options Policy disables required compatibility with RPC-style wrappers.

Use this command to toggle between enabling and disabling required compatibility with RPC-style wrappers.

Related Commands

wsdl-validate-faults

Examples

- Enables and subsequently disables required compatibility with RPC-style wrappers.

```
# wsdl-wrapped-faults
:
# wsdl-wrapped-faults
#
```

wsi-validate

Validates WSDL files against section 5 of WS-I Basic Profile.

Syntax

wsi-validate {**ignore** | **warn** | **fail**}

Parameters

ignore Disables conformance checking of the WS-I Basic Profile.

warn Logs warnings for WS-I Basic Profile violations.

fail Forces conformance of the WS-I Basic Profile.

Guidelines

By default, the system issues a warning when validation fails. Use this command to change this behavior.

Examples

- Forces conformance against section 5 of WS-I Basic Profile.

```
# wsi-validate fail
#
```

xacml-debug

Indicates whether to compile XACML policy with debug information.

Syntax

xacml-debug {**on** | **off**}

Parameters

on Makes the compiler add more debugging information when evaluating a XACML policy.

off (Default) Does not compile the XACML policy with debugging information.

Guidelines

The **xacml-debug** command indicates whether to compile the XACML policy with debug information.

Note: XACML debugging messages are also controlled by log events in the XACML category. Use the `debug` log level to view the full XACML debugging messages.

---

xslt-version

Specifies the XSLT version to use during compilation.

Syntax

xslt-version { **xslt10** | **xslt20** | **stylesheet**}

Parameters

xslt10 Uses XSLT 1.0.

xslt20 Uses XSLT 2.0.

stylesheet
Supports for both versions. Selection is based on specifications internal to each style sheet.

Examples

- Uses XSLT 2.0 when compiling style sheets.

```
# xslt-version xslt20
#
```

Chapter 9. Conformance Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Conformance Policy configuration mode. To enter this configuration mode, use the Global **conformancepolicy** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Conformance Policy configuration mode.

assert-bp10-conformance

Controls the attachment of an assertion to WS-I Basic Profile 1.0 messages.

Syntax

assert-bp10-conformance {**on** | **off**}

Parameters

on (Default) Attaches an assertion.

off Does not attach an assertion. Use this setting with `filter` action.

Guidelines

The **assert-bp10-conformance** command is meaningful only when validating messages against WS-I Basic Profile 1.0. The profiles for message validation are defined using the **profiles** command. Use the **assert-bp10-conformance** command only when the value for the **profiles** command includes `BP10` as part of its definition.

Related Commands

profiles

Examples

- Disables the attachment of assertions when validating compliance against WS-I Basic Profile 1.0.

```
# profiles BP10
# assert-bp10-conformance off
```

- Enables the attachment of assertions when validating compliance against WS-I Basic Profile 1.0, which restores the default state.

```
# assert-bp10-conformance on
```

fixup-stylesheet

Identifies which style sheets to invoke after conformance analysis.

Syntax

fixup-stylesheet *file*

no fixup-stylesheet *file*

Parameters

file Specifies the name and location of the style sheet.

Guidelines

The **fixup-stylesheet** command defines which style sheets to invoke after conformance analysis. These style sheets can transform the analysis results to repair instances of nonconformance. Corrective style sheets cannot be applied to `filter` actions.

For each style sheet to include, use the **fixup-stylesheet** command. To remove a style sheet, use the **no fixup-stylesheet** command.

Examples

- Defines the `local:///conformError1.xsl` and `local:///conformError2.xsl` style sheets as corrective style sheets.

```
# fixup-stylesheet local:///conformError1.xsl
# fixup-stylesheet local:///conformError2.xsl
#
```

- Removes the `local:///conformError2.xsl` style sheet as a corrective style sheet.

```
# no fixup-stylesheet local:///conformError2.xsl
#
```

---

ignored-requirements

Identifies which profile requirements to exclude from validation.

Syntax

ignored-requirements *profile:requirement*

no ignored-requirements *profile:requirement*

Parameters

profile Specifies the literal representation for the name of the profile. Use one of the following literals:

`AP1.0` Indicates Web Services Interoperability (WS-I) Attachment Profile, version 1.0.

`BP1.0` Indicates WS-I Basic Profile, version 1.0.

`BP1.1` Indicates WS-I Basic Profile, version 1.1.

`BSP1.0` Indicates WS-I Basic Security Profile, version 1.0.

requirement
Specifies the identifier of the requirement in the profile. The identifier follows the naming convention in the profile documentation.

Guidelines

The **ignored-requirements** command defines which profile requirements to exclude from validation.

For each requirement to exclude, use the **ignored-requirements** command. To remove an excluded requirement, use the **no ignored-requirements** command.

For information about the requirements defined in the supported profiles, refer to the following Web sites:

WS-I Attachments Profile, version 1.0
http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html

WS-I Basic Profile, version 1.0
http://www.ws-i.org/Profiles/BasicProfile-1.0.html

WS-I Basic Profile, version 1.1
http://www.ws-i.org/Profiles/BasicProfile-1.1.html

WS-I Basic Security Profile, version 1.0
http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html

Examples

- Excludes requirements `R4221` and `R4222` in the WS-I Basic Security Profile, version 1.0, from validation.

```
# ignored-requirements BSP1.0:R4221
# ignored-requirements BSP1.0:R4221
#
```

- Removes the excluded requirement `BSP1.0:R4221` so that it is part of conformance validation.

```
# no ignored-requirements BSP1.0:R4221
#
```

---

profiles

Defines the profiles against which to validate conformance.

Syntax

profiles *profile*[+*profile*...]

Parameters

profile Specifies a literal for a Web Services Interoperability (WS-I) profile. To specify multiple profiles, use a plus sign (+) between each profile. Use a combination of the following literal representation for profiles:

`AP10` Includes WS-I Attachment Profile, version 1.0.

`BP10` Includes WS-I Basic Profile, version 1.0.

`BP11` Includes WS-I Basic Profile, version 1.1.

`BSP10` Includes WS-I Basic Security Profile, version 1.0.

The default is `AP10+BP11+BSP10`.

Examples

- Specifies that messages validation is against WS-I Basic Profile, version 1.1 and WS-I Basic Security Profile, version 1.0.

```
# profiles BP11+BSP10
#
```

- Specifies that messages validation is against WS-I Attachments Profile, WS-I Basic Profile, version 1.1, and WS-I Basic Security Profile, version 1.0, which restores the default state.

```
# profiles AP+10+BP11+BSP10
#
```

reject-include-summary

Controls the inclusion of the summary in the rejection request message.

Syntax

reject-include-summary {**on** | **off**}

Parameters

on Includes the summary.

off (Default) Does not include the summary.

Guidelines

The **reject-include-summary** command determines whether to include a summary of the conformance analysis in the request rejection message. This command is meaningful only when request messages are rejected. Request messages are rejected when the value for the **reject-level** command is set to **failure** or **warning**.

Related Commands

reject-level

Examples

- Includes a summary in rejection request messages that indicate conformance failures.

```
# reject-level failure
# reject-include-summary on
#
```

- Does not include a summary in rejection request messages, which restores the default state.

```
# reject-include-summary off
#
```

reject-level

Identifies when to reject messages for requests based on conformance.

Syntax

reject-level {**failure** | **never** | **warning**}

Parameters

failure
Rejects messages that are identified as conformance failures.

never (Default) Never rejects messages.

warning
Rejects messages that are identified as either conformance failures or conformance warnings.

Guidelines

The **reject-level** command identifies the degree of nonconformance that causes a request message to be rejected. When a request message is rejected, you can use the **reject-include-summary** command to include a summary of the conformance analysis in the rejection message.

Examples

- Includes a summary in rejection messages that indicate conformance failures.

```
# reject-level failure
# reject-include-summary on
#
```

---

report-level

Identifies when to record a conformance report for requests.

Syntax

report-level {**always** | **failure** | **never** | **warning**}

Parameters

always
Always records a conformance report.

failure
Records a conformance report for conformance failures.

never (Default) Never records a conformance report.

warning
Records a conformance report for both conformance failures and conformance warnings.

Guidelines

The **report-level** command determines when to send a conformance report. To send a conformance report for requests, you must use the **report-target** command to define where to send the report.

Related Commands

report-target

Examples

- Sends conformance reports for conformance failures to `datapower.com/conform` with the HTTP protocol.

```
# report-level failures
# report-target http://datapower.com/conform
```

report-target

Specifies where to send conformance reports for requests.

Syntax

report-target *URL*

Parameters

URL Specifies the location to send conformance reports. Use the following URL format:

```
protocol://host/URI
```

Guidelines

The **report-target** command identifies where to send conformance reports for requests. This command is meaningful only when the value for the **report-level** command is **always**, **failure**, or **warning**.

Examples

- Sends conformance reports for conformance failures for requests to `datapower.com/conform` with the HTTP protocol.

```
# report-level failures
# report-target http://datapower.com/conform
```

response-properties-enabled

Controls the enablement of conformance for response messages.

Syntax

response-properties-enabled {**on** | **off**}

Parameters

on Allows the definition of conformance for response messages.

off (Default) Does not allow the definition of conformance for response messages.

Guidelines

The **response-properties-enabled** command determines whether to enable conformance on response messages.

Examples

- Enables conformance for response messages.

```
# response-properties-enabled on
# response-reject-level failure
#
```

- Does not enable conformance for response messages.

```
# response-properties-enabled off
#
```

response-reject-include-summary

Controls the inclusion of the summary in the rejection message for responses.

Syntax

response-reject-include-summary {**on** | **off**}

Parameters

on Includes the summary.

off (Default) Does not include the summary.

Guidelines

The **response-reject-include-summary** command determines whether to include a summary of the conformance analysis in the rejection message for responses. This command is meaningful only when response messages are rejected. Response messages are rejected when the value for the **response-reject-level** command is set to **failure** or **warning**.

Related Commands

response-reject-level

Examples

- Includes a summary in rejection messages that indicate conformance failures for responses.

```
# response-reject-level failure
# response-reject-include-summary on
#
```

- Does not include a summary in rejection messages for responses, which restores the default state.

```
# response-reject-include-summary off
#
```

response-reject-level

Identifies when to reject messages for responses based on conformance.

Syntax

response-reject-level {**failure** | **never** | **warning**}

Parameters

failure
Rejects messages that are identified as conformance failures.

never (Default) Never rejects messages.

warning
Rejects messages that are identified as either conformance failures or conformance warnings.

Guidelines

The **response-reject-level** command identifies the degree of nonconformance that causes a response message to be rejected. When a response message is rejected, you can use the **response-reject-include-summary** command to include a summary of the conformance analysis in the rejection message.

Examples

- Includes a summary in rejection messages that indicate conformance failures for responses.

```
# response-reject-level failure
# response-reject-include-summary on
#
```

---

response-report-level

Identifies when to record a conformance report for responses.

Syntax

response-report-level {**always** | **failure** | **never** | **warning**}

Parameters

always
Always records a conformance report.

failure
Records a conformance report for conformance failures.

never (Default) Never records a conformance report.

warning
Records a conformance report for both conformance failures and conformance warnings.

Guidelines

The **response-report-level** command determines when to send a conformance report for responses. To send a conformance report for responses, you must use the **response-report-target** command to define where to send the report.

Related Commands

response-report-target

Examples

- Sends conformance reports for conformance failures for responses to `datapower.com/conform/responses` with the HTTP protocol.

```
# response-report-level failures
# response-report-target http://datapower.com/conform/responses
```

---

response-report-target

Specifies where to send conformance reports for responses.

Syntax

response-report-target *URL*

Parameters

URL Specifies the location to send conformance reports. Use the following URL format:

```
protocol://host/URI
```

Guidelines

The **response-report-target** command identifies where to send conformance reports for responses. This command is meaningful only when the value for the **response-report-level** command is **always**, **failure**, or **warning**.

Examples

- Sends conformance reports for conformance failures for responses to `datapower.com/conform` with the HTTP protocol.

```
# response-report-level failures
# response-report-target http://datapower.com/conform/responses
```

---

result-is-conformance-report

Determines whether to use conformance analysis as output.

Syntax

result-is-conformance-report {**on** | **off**}

Parameters

on Delivers a conformance analysis as a `results` action.

off (Default) Delivers the original message, possibly modified by one or more style sheets, to the next document processing action.

Guidelines

The **result-is-conformance-report** command indicates whether the conformance analysis is delivered as a `results` action or is delivered to the next action in the processing rule. When delivered as a `results` action, the intended purpose is with a loopback XML firewall, which returns the results to the client.

Examples

- Specifies that analysis results are used as output.

```
# result-is-conformance-report on
```

Chapter 10. CRL configuration mode

This chapter provides an alphabetic listing of commands that are available in CRL configuration mode. CRL is the abbreviation for Certificate Revocation List.

To enter this configuration mode, use the crypto **crl** command. While in CRL configuration mode, define the CRL update policy to enable the scheduled retrieval of CRLs with either LDAP or HTTP as a retrieval mechanism.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in CRL configuration mode.

bind-dn

Specifies the login DN (distinguished name) used to access an LDAP server.

Syntax

bind-dn *dn*

Parameters

dn Specifies the login name to access the target LDAP server.

Guidelines

You must specify a login DN when defining LDAP-enabled CRL Update Policy.

Related Commands

bind-pass, **read-dn**, **refresh**, **remote-address**

Examples

- Enters CRL Mode to create the `LDAP1440` LDAP-enabled CRL Update Policy. The LDAP server is accessed with the account name of `X` with a password of `1PAss$WorD`.

```
# crl LDAP1440 ldap
Entering CRL mode for 'LDAP1440'
# bind-dn X
# bind-pass 1PAss$WorD
#
```

bind-pass

Specifies the password to access an LDAP server.

Syntax

bind-pass *password*

Parameters

password
Specifies the password for the login DN.

Guidelines

You must specify a password when defining an LDAP-enabled CRL Update Policy.

Related Commands

bind-dn, **read-dn**, **refresh**, **remote-address**

Examples

- Enters CRL Mode to create the `LDAP1440` LDAP-enabled CRL Update Policy. The LDAP server is accessed with the account name of `X` with a password of `1PAss$WorD`.

```
# crl LDAP1440 ldap
Entering CRL mode for 'LDAP1440'
# bind-dn X
# bind-pass 1PAss$WorD
#
```

---

fetch-url

Specifies the URL of the target CRL.

Syntax

fetch-url *URL*

Parameters

URL Specifies the URL of the target CRL.

Guidelines

You must specify the URL of the target CRL when defining an HTTP-enabled CRL Update Policy. The CRL is stored in memory. Consequently, the CRL is lost after a system reboot.

Related Commands

refresh

Examples

- Enters CRL mode to create the `HTTP30` HTTP-enabled CRL Update Policy. The target CRL is retrieved from the specified URL.

```
# crl HTTP30 http
Entering CRL mode for 'HTTP30'
# fetch-url http://crl.verisign.com/ATTClass1Individual.crl
#
```

---

issuer

Specifies a Validation Credentials Set to validate the credentials of the CRL issuer.

Syntax

issuer *name*

Parameters

name Specifies the name of an existing Validation Credentials Set.

Guidelines

This property is required to implement a CRL Update Policy.

Examples

- Enters CRL mode to create the HTTP30 HTTP-enabled CRL Update Policy. Specifies crlValidate as the Validation Credentials to validate the CRL issuer.

```
# crl HTTP30 http
Entering CRL mode for 'HTTP30'
# issuer crlValidate
#
```

read-dn

Specifies the Distinguished Name of the CA that issued the target CRL.

Syntax

read-dn *dn*

Parameters

dn Specifies the Distinguished Name of the CA that issued the CRL. Enclose the value in double quotation marks.

Guidelines

You must specify a CA when defining an LDAP-enabled CRL Update Policy.

The specified CRL is stored in memory. Consequently, the CRL is lost after a system reboot.

Related Commands

bind-dn, **bind-pass**, **refresh**, **remote-address**

Examples

- Enters CRL Mode to create the LDAP1440 LDAP-enabled CRL Update Policy. The LDAP server is accessed with the account name of X and a password of 1PAss$WorD. The target certificate is issued by VeriSign Australia.

```
# crl LDAP1440 ldap
Entering CRL mode for 'LDAP1440'
# bind-dn X
# bind-pass 1PAss$WorD
# read-dn "C=AU,
 ST=Victoria, L=South Melbourne, O=VeriSign Australia
 Limited, OU=IT Department, CN=www.verisign.com.au"
#
```

refresh

Specifies the interval between CRL updates.

Syntax

refresh *minutes*

Parameters

minutes
Specifies the interval in minutes between CRL updates.

Guidelines

You must specify a refresh interval when defining either an HTTP-enabled or LDAP-enabled CRL Update Policy.

Related Commands

bind-dn, **bind-pass**, **fetch-URL**, **read-dn**, **remote-address**

Examples

- Enters CRL Mode to create the `LDAP1440L` DAP-enabled CRL Update Policy. The `ragnarok` LDAP server (with default port 389) is accessed with the account name of `X` and a password of `1PAss$WorD`. The target certificate is issued by VeriSign Australia and is refreshed on daily.

```
# crl LDAP1440 ldap
Entering CRL mode for 'LDAP1440'
# bind-dn X
# bind-pass 1PAss$WorD
# read-dn "C=AU,
 ST=Victoria, L=South Melbourne, O=VeriSign Australia
 Limited, OU=IT Department, CN=www.verisign.com.au"
# remote-address ragnarok
# refresh 1440
#
```

- Enters CRL Mode to create the `HTTP30` HTTP-enabled CRL Update Policy. The target CRL is retrieved from the specified URL and refreshed every half hour.

```
# crl HTTP30 http
Entering CRL mode for 'HTTP30'
# fetch-URL http://crl.verisign.com/ATTClass1Individual.crl
# refresh 30
#
```

---

remote-address

Specifies the address of the LDAP server.

Syntax

remote-address *server* [*port*]

Parameters

server Specifies the host name or IP address of the LDAP server.

port Identifies the server port that monitors LDAP traffic. Use an optional integer in the range of 0 through 65535. The default is 389.

Guidelines

You must specify a server IP address to define an LDAP-enabled CRL Update Policy.

Related Commands

bind-dn, **bind-pass**, **read-dn**, **refresh**

Examples

- Enters CRL Mode to create the LDAP1440 LDAP-enabled CRL Update Policy. The ragnarok LDAP server (with default port 389) is accessed with the account name of X and a password of 1PAss$WorD. The target certificate is issued by VeriSign Australia.

```
# crl LDAP1440 ldap
Entering CRL mode for 'LDAP1440'
# bind-dn X
# bind-pass 1PAss$WorD
# read-dn "C=AU,
 ST=Victoria, L=South Melbourne, O=VeriSign Australia
 Limited, OU=IT Department, CN=www.verisign.com.au"
# remote-address ragnarok
#
```

---

ssl-profile

Assigns a client (or forward) SSL Proxy Profile.

Syntax

ssl-profile *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The client SSL Proxy profile must be previously created with the **sslproxy** command.

The assigned SSL Proxy Profile enables CRL retrieval over a secure connection. The SSL proxy profile specifies the SSL operational mode (client) and identifies the cryptographic resources (key, certificates, and cipher list) available to support the SSL connection.

Assignment of a client SSL Proxy profile is optional. In the absence of an assigned SSL Proxy Profile, the CRL Update Policy attempts to establish a nonsecure connection (either HTTP or LDAP) with the CRL server.

Related Commands

sslproxy

Examples

- Assigns the SSL-CRLAccess SSL Policy Profile to the current CRL Update Policy.

```
# ssl-profile SSL-CRLAccess
#
```

Chapter 11. Crypto configuration mode

This chapter provides an alphabetic listing of commands that are available in Crypto configuration mode. To enter this configuration mode, use the Global **crypto** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

certificate

Creates an alias for an X.509 certificate stored on the local flash.

Syntax

certificate *certificate-alias URL* **password** *password* [**ignore-expiration**]

certificate *certificate-alias URL* **password-alias** *password-alias* [**ignore-expiration**]

no certificate

Parameters

certificate-alias
Specifies an alias for a stored certificate.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

URL Specifies a local URL that identifies the file that contains the certificate.

- If stored in the public cryptographic area, takes the `pubcert:///`*`filename`* form.
- If stored in the private cryptographic area, takes the *`filename`* form.

password *password*
Specifies the plaintext password required to access the certificate file.

password-alias *password-alias*
Specifies the alias for the encrypted password required to access the certificate file.

ignore-expiration
Specifies an optional keyword to allow the creation of a certificate prior to its activation date (the `NotBefore` value in the certificate) or after its expiration date (the `NotAfter` value in the certificate). Although the certificate is in the **up** state, objects that reference the certificate use the internal expiration values.

In other words, the certificate itself is in the **up** state, but Validation Credentials, Firewall Credentials, or Identification Credentials that references the certificate adhere to the internal expiration values. If the certificate is used for a certificate chain validation from a Validation Credentials and the certificate is not valid, validation fails. Similarly, if the certificate is used from an Identification Credentials, the DataPower

appliance sends the certificate to the SSL peer for an SSL connection, but the peer can reject the certificate as not valid.

Guidelines

The **password** or **password-alias** keyword is required only when a certificate file is password-protected.

Prior to using the **password-alias** keyword, you must use the **password-map** command to 3DES-encrypt the certificate password and associate an alias with the encrypted password. An attempt to reference an encrypted password not found in the Password map results in command failure.

- In environments that use plaintext (unencrypted) passwords, the *password* argument is used to open and read the certificate file.
- In environments that use encrypted passwords, the *password-alias* argument is searched for in the password map file and its associated encrypted password is identified. The encrypted password, in turn, is 3DES-decrypted (using the locally generated host key) to yield the plaintext password used to open and read the certificate file.

Use the **certificate** command in conjunction with the **key** and **idcred** commands to create an Identification Credentials. An Identification Credentials consists of a certificate, which contains a public key, and the corresponding private key.

Use the **certificate** command in conjunction with the **valcred** command to create a Validation Credentials. A Validation Credentials can be used, but is not required, during the SSL handshake procedure to authenticate the certificate that is received from the remote SSL peer.

The **no certificate** command deletes only the alias for the stored certificate. The file that contains the actual certificate remains on the appliance.

Related Commands

certificate (Crypto Validation), **copy**, **key**, **password-map**, **profile**, **valcred**

Examples

- Creates the `bob` alias for the `bob.pem` X.509 certificate. Stores the target certificate in the public cryptographic area.

```
# certificate
bob pubcert:bob.pem
Creating certificate 'bob'
#
```

- Creates an the `bob` alias for the `bob.pem` certificate. Stores the target certificate in the public cryptographic area. Allows the certificate to be accessed with the `pikesville` plaintext password.

```
# certificate bob pubcert:bob.pem
password pikesville
Creating certificate 'bob'
#
```

- Creates an the `bob` alias for the `bob.pem` certificate. Stores the target certificate in the public cryptographic area. Allows the certificate to be accessed with the `dundaulk` encrypted password alias.

```
# certificate bob pubcert:bob.pem
password-alias dundaulk
Creating certificate 'bob'
#
```

- Deletes the bob certificate alias.

```
# no certificate bob
Certificate 'bob' deleted
#
```

cert-monitor

Enters Crypto Certificate Monitor configuration mode.

Syntax

cert-monitor

Guidelines

The Certificate Monitor is a configurable periodic task that checks the expiration date of all certificate objects.

While in Crypto Certificate Monitor configuration mode, you can set values that establish both a polling frequency and a notification window, during which the monitor generates log messages recording that a specified certificate is nearing its expiration date.

Examples

- Enters Crypto Certificate Monitor configuration mode.

```
# cert-monitor
Crypto Certificate Monitor configuration mode
#
```

crl

Creates a named CRL (Certificate Revocation List) Update Policy and enters CRL Mode.

Syntax

crl *name* {**http** | **ldap**}

no crl

Parameters

name Specifies the name of the CRL update policy.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

http Indicates an HTTP-enabled CRL update policy.

ldap Indicates an LDAP-enabled CRL update policy

Guidelines

While in CRL Mode, use the **fetch-url** and **refresh** commands to define an HTTP-enabled CRL update policy; use the **bind-dn**, **bind-pass**, **read-dn**, **refresh**, and **remote-address** commands to define an LDAP-enabled CRL update policy.

Use the **no crl** command to delete a CRL update policy.

Examples

- Enters CRL Mode to create the `HTTP30` HTTP-enabled CRL update policy.

```
# crl HTTP30 http
Entering CRL mode for 'HTTP30'
#
```

- Enters CRL Mode to create the `LDAP1440` LDAP-enabled CRL update policy.

```
# crl LDAP1440 ldap
Entering CRL mode for 'LDAP1440'
#
```

- Deletes the `LDAP1440` LDAP-enabled CRL update policy.

```
# no crl LDAP1440
#
```

crypto-export

Creates an export package that contains certificate or key objects.

Syntax

Exporting certificates

crypto-export cert *name* [...] **output** *file*

Exporting keys (HSM models)

crypto-export key *name* [...] **output** *file* **mechanism hsmkwk**

Parameters

key *name* [...]
Identifies the names of the keys to include in the export package. To specify more than one key, use a space separated list.

cert *name* [...]
Identifies the names of the certificates to include in the export package. To specify more than one certificate, use a space separated list.

output *file*
Identifies the name and location to store the export.

mechanism hsmkwk
(HSM appliances only) Indicates that the export package is exported via an HSM-generated key wrapping key.

Examples

- Creates the `exportBob` export package. The package contains the `bob` Certificate object.

```
# crypto-export
cert bob output exportBob
#
```

crypto-import

Imports an export package that contains certificate or key objects.

Syntax

Importing certificates

crypto-import cert *name* [...] **input** *file*

Importing keys (HSM models)

crypto-import key *name* [...] **input** *file* [**password-alias** *alias*] [**mechanism hsmkwk**]

crypto-import key *name* [...] **input** *file* [**password** *password*] [**mechanism hsmkwk**]

Parameters

key *name* [...]
Identifies the names of the keys to import. To specify more than one key, use a space separated list.

cert *name* [...]
Identifies the names of the certificates to import. To specify more than one certificate, use a space separated list.

input *file*
Identifies the name and location of the stored export.

password *password*
(HSM appliances only) Optionally specifies the password that was used to encrypt the input file. This parameter is mutually exclusive to the **password-alias** parameter.

password-alias *alias*
(HSM appliances only) Optionally specifies the password that was used to encrypt the input file. This parameter is mutually exclusive to the **password** parameter.

mechanism hsmkwk
(HSM appliances only) Optionally indicates that the imported material can be exported at a later time with an HSM-generated key wrapping key.

Examples

- Creates the `exportBob` export package. The package contains the `bob` Certificate object.

```
# crypto-export
cert bob output exportBob
#
```

decrypt

Decrypts a file stored on the appliance.

Syntax

decrypt *URL* **idcred** *name* **alg** *algorithm*

Parameters

URL Identifies the local file to be decrypted, and takes the *directory*:/// *filename* format.

directory
Must be one of the following directory-specific keywords:

`audit:` Contains the audit log

`cert:` Contains domain-specific private keys and certificates

`config:`
Contains configuration scripts

`export:`
Contains export packages

`image:` Contains primary and secondary firmware images

`local:` Contains user processing resources such as style sheets, schemas, document encryption maps, or XML mapping files

`logstore:`
Contains logging files

`logtemp:`
Contains active and rotated log files

`pubcert:`
Contains well-known (for example, VeriSign) public certificate files

`sharedcert:`
Contains private keys and certificates which are shared across domains

`store:` Contains DataPower-supplied processing resources such as style sheets, schemas and authentication/authorization files

`tasktemplates:`
Contains Task Template files

`temporary:`
Contains temporary files

filename
Specifies the name of the file to decrypt.

idcred *name*
Specifies an existing alias for an Identification Credentials (a matched public/private key pair) to identify the DataPower appliance. The specified argument references the "local" private key to decrypt a file.

alg *algorithm*
Identifies the decryption method.

Related Commands

certificate, **encrypt** (Crypto), **idcred**, **key**

Examples

- Uses S/MIME to decrypt the `HSec.xsd` schema file with the Identification Credentials referenced by the `bob` alias.

```
# decrypt
store:///HSec.xsd idcred bob alg smime
File 'HSEC.xsd' successfully decrypted
#
```

encrypt

Encrypts a file stored on the appliance.

Syntax

encrypt *URL* **cert** *alias* **alg** *algorithm*

Parameters

URL Identifies the local file to be encrypted, and takes the *directory*:/// *filename* format.

directory
Must be one of the following directory-specific keywords that reference specific directories.

`audit:` Contains the audit log

`cert:` Contains domain-specific private keys and certificates

`config:`
Contains configuration scripts

`export:`
Contains export packages

`image:` Contains primary and secondary firmware images

`local:` Contains user processing resources such as style sheets, schemas, document encryption maps, or XML mapping files

`logstore:`
Contains logging files

`logtemp:`
Contains active and rotated log files

`pubcert:`
Contains well-known (for example, VeriSign) public certificate files

`sharedcert:`
Contains private keys and certificates which are shared across domains

`store:` Contains DataPower-supplied processing resources such as style sheets, schemas and authentication/authorization files

`tasktemplates:`
Contains Task Template files

`temporary:`
Contains temporary files

filename
Specifies the name of the file to encrypt.

cert *alias*
Specifics an existing alias for the personal (public) certificate of the recipient. The argument references the local copy of the public key to encrypt a file.

alg *algorithm*
Identifies the encryption method.

Related Commands

certificate, **idcred**, **send file**, **sign** (Crypto)

Examples

- Encrypts the `FWSec-1` log file with the recipient certificate that is referenced by the `bob` alias.

```
# encrypt
logtemp:///FWSec-1 cert bob alg smime
File 'FWSec-1' successfully encoded
#
```

---

fwcred

Enters Firewall Credentials configuration mode.

Syntax

fwcred *name*

no fwcred *name*

Parameters

name Specifies the name of the Firewall Credentials.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

While in Firewall Credentials configuration mode, use the **key** and **sskey** commands to add specified keys to the Firewall Credentials, and use the **certificate** command to add specified certificates to the list.

A Firewall Credentials can be assigned to DataPower services. The Firewall Credentials provides a means to specify which keys and certificates are permitted with various cryptographic extension functions in support of service-specific security activities.

In the absence of a Firewall Credentials, all keys and certificates that are available on the DataPower appliance can support firewall activities.

Use the **no fwcred** command to delete a Firewall Credentials.

Related Commands

certificate, **key**, **sskey**

Examples

- Enters Firewall Credentials configuration mode to create the `FWCred-1` Firewall Credentials.

```
# fwcred FWCred-1
Entering Firewall Credentials mode for 'FWCred-1'
#
```

- Deletes the FWCred-1 Firewall Credentials.

```
# no fwcred FWCred-1
Firewall Credentials 'FWCred-1' deleted
#
```

hsm-clone-kwk (HSM models)

Clones a key wrapping key between HSM-equipped appliances.

Syntax

hsm-clone-kwk [**input** *filename*] [**output** *filename*]

Parameters

input *filename*
Indicates the name of the local file to use as input to the cloning action. During the first part of this four-part task, do not specify this parameter. During the other parts of this task, this parameter is required.

output *filename*
Indicates the name of the local file that the cloning action creates. During the last part of this four-part task, do not specify this parameter. During the other parts of this task, this parameter is required.

Guidelines

This command is available only on systems with an internal HSM.

Use the **hsm-clone-kwk** command if two HSM-equipped systems share the same red (hardware) key and if both systems are at the same FIPS security level. This command copies the key-wrapping key from the source HSM system to the destination HSM system. You must run this command four times.

1. On the source HSM system, create an output file (for example, temporary:///source-one that contains the key material. After validating that the command created the file, copy it to the destination HSM system.
2. On the destination HSM system, create an output file (for example, temporary:///destination-two that uses the copied file (for example, temporary:///source-one) as the input file. After validating that the command created the file, copy it to the source HSM system.
3. On the source HSM system, create an output file (for example, temporary:///source-three that uses the copied file (for example, temporary:///destination-two) as the input file. After validating that the command created the file, copy it to the destination HSM system.
4. On the destination HSM system, use the copied file (for example, temporary:///source-three) as the input file.

At this point, the source and destination HSM systems share the same key-wrapping key. After cloning the key-wrapping key on each HSM domain member, the domain member can share keys in the following way:

1. Creating an export crypto object
2. Transferring the export crypto object to a target system in the HSM domain
3. Importing the export crypto object on the target system

Refer to the HSM documentation for information about performing the key cloning task from the WebGUI.

Related Commands

hsm-delete-key, **hsm-reinit**

hsm-delete-key (HSM models)

Deletes a key from the HSM (Hardware Security Module).

Syntax

hsm-delete-key *key*

Parameters

key Identifies the key stored on the HSM.

Guidelines

This command is available only on systems with an internal HSM.

Related Commands

hsm-clone-kwk, **hsm-reinit**

Examples

- Deletes the `bob` key from the HSM.

```
# hsm-delete-key bob
#
```

hsm-reinit (HSM models)

Restores the HSM to its factory state

Syntax

hsm-reinit

Guidelines

This command is available only on systems with an internal HSM.

CAUTION:
This command destroys all data stored on the HSM and restores the HSM to its factory state.

Related Commands

hsm-clone-kwk, **hsm-delete-key**

Examples

- Restores the HSM to its factory state.

```
# hsm-reinit
#
```

idcred

Creates an SSL proxy Identification Credentials.

Syntax

idcred *name key-alias certificate-alias* [**ca** *certificate-alias-n* ...]

Parameters

name Specifies the name of the Identification Credentials that authenticates the appliance.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

key-alias
Specifies an existing alias for the private key that is referenced by the Identification Credentials.

certificate-alias
Specifies and existing alias for the certificate that is referenced by the Identification Credentials.

ca *certificate-alias-n*
Optionally identifies an intermediate certificate required to establish a *chain-of-trust* starting with the certificate that is referenced by the *certificate-alias* argument and a CA trusted by the remote SSL peer.

The list can contain up to 10 intermediate certificates.

Guidelines

An SSL proxy uses an Identification Credentials to authenticate itself to a remote peer.

The SSL standard requires an SSL server to authenticate itself to a remote SSL client. Consequently, an SSL proxy operating as an SSL server (in either `reverse` or `two-way` proxy mode) must be assigned an Identification Credentials with which to authenticate itself to a remote SSL client.

The SSL standard allows an SSL server to authenticate the remote client peer. Consequently, an SSL proxy operating as a SSL client (in either `forward` or `two-way` proxy mode) can be assigned a set of identification credentials if the remote SSL server requires authentication. While SSL servers typically do not require client identification, certain highly secure web sites may impose such a requirement.

Prior to creating an Identification Credentials, you must:

- Use the **key** command to create an alias for the private key.
- Use the **certificate** command to create an alias for the certificate.

The **no idcred** command deletes only the alias for the Identification Credentials. The aliases used to create the set (that is the certificate alias and private key alias) remain available for use, as do as the files that contain the actual certificate and private key that comprise the Identification Credentials.

Related Commands

certificate, **decrypt**, **key**

Examples

- Creates the `bob` Identification Credentials that consists of the private key aliased by `bob` and the X.509 certificate aliased by `bob`.

```
# idcred bob bob bob
Creating identification credentials 'bob'
#
```

- Creates the `bob` Identification Credentials that consists of the private key aliased by `bob` and the X.509 certificates aliased by `bob` and `bob-intermediate`.

```
# idcred bob bob bob ca bob-intermediate
Creating identification credentials 'bob'
#
```

- Deletes the Identification Credentials alias `bob`.

```
# no idcred bob
Identification Credentials 'bob' deleted
#
```

kerberos-kdc

Enters Kerberos KDC Server Configuration.

Syntax

kerberos-kdc *name*

no kerberos-kdc *name*

Parameters

name Specifies the name of the Kerberos KDC.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A Kerberos KDC Server issues Kerberos tickets; essentially a KDC is a database of all users within the Kerberos realm, or administrative domain. Each user entry in the database is called a principal, and includes an associated encryption key derived from the user password.

Use the **no kerberos-kdc** command to delete a KDC Server object.

Examples

- Creates the `Hades` Kerberos KDC Server object, and enters Kerberos KDC Server configuration mode.

```
# kerberos-kdc Hades
Kerberos KDC Server configuration mode
#
```

- Deletes the `Hades` Kerberos KDC Server object.

```
# no kerberos-kdc Hades
kerberos-kdc Hades: deleted
#
```

kerberos-keytab

Enters Kerberos Keytab Configuration.

Syntax

kerberos-keytab *name*

no kerberos-keytab *name*

Parameters

name Specifies the name of the Kerberos keytab.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A keytab (or key table) is an unencrypted file that contains a list of Kerberos principals and their passwords.

Use the **no kerberos-keytab** command to delete a Kerberos keytab object.

Examples

- Creates the Kerberos keytab object, `Inferno`, and enters Kerberos Keytab configuration mode.

```
# kerberos-keytab Inferno
Kerberos Keytab configuration mode
#
```

- Deletes the Kerberos keytab object, `Inferno`.

```
# no kerberos-keytab Inferno
kerberos-keytab Inferno: deleted
#
```

---

key

Creates an alias for a private key stored on the appliance.

Syntax

key *key-alias URL* [**password** *password*]

key *key-alias URL* [**password-alias** *password-alias*]

no key *key-alias*

Parameters

key-alias

Specifies an alias for the stored private key.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

URL Specifies a local URL that identifies the file that contains the private key.

- To store the private key in the private cryptographic area, the URL takes the *filename* form.
- To store the private key in the public cryptographic area, the URL takes the pubcert:///*filename* form.

CAUTION:
Do not store private key files in the public cryptographic area. This area is intended for the storage of public certificate files.

password *password*
Optionally identifies the plaintext password required to access the private key file.

password-alias *password-alias*
Optionally identifies the alias for the encrypted password required to access the private key file.

Guidelines

The **password** or **password-alias** keyword is required only when a key file is password-protected.

Before using the **password-alias** keyword, you must use the **password-map** command to 3DES-encrypt the key password and associate an alias with the encrypted password. An attempt to reference an encrypted password not found in the Password map results in command failure.

- In environments that use plaintext (unencrypted) passwords, the *password* argument opens and reads the key file.
- In environments that use encrypted passwords, the *password-alias* argument is used to search for the password map file and to identify its associated encrypted password. The encrypted password in turn is then 3DES-decrypted (with the locally generated host key) to yield the plaintext password that opens and reads the key file.

Use the **key** command in conjunction with the **certificate** and **idcred** commands to create an Identification Credentials that consists of a certificate, which contains a public key, and the corresponding private key.

Use the **no key** command to delete only the alias for the stored private key. The file that contains the actual key remains on the appliance.

Related Commands

certificate, **idcred**, **password-map**

Examples

- Creates the `bob` alias for the `K2.pem` private key. The target key is in the private cryptographic storage area.

```
# key bob K2.pem
Creating key 'bob'
#
```

- Creates the `bob` alias for the `K2.der` private key. The target key is in the private cryptographic area, and is accessed with the `annapolis` plaintext password.

```
# key bob K2.der password annapolis
Creating key 'bob'
#
```

- Creates the `bob` alias for the `K2.der` private key. The target key is in the private cryptographic area, and is accessed with the `towson` encrypted password aliased.

```
# key bob K2.der password-alias towson
Creating key 'bob'
#
```

- Deletes the `bob` private key alias.

```
# no key bob
Key 'bob' deleted
#
```

keygen

Generates a public-private key pair and a CSR (certificate signing request) for a server.

Syntax

Generates a key pair on a non-HSM appliance

keygen [{**C** | **countryName**} *iso-code*] [{**L** | **localityName**} *locality*] [{**ST** | **stateOrProvinceName**} *state*] [{**O** | **organizationName**} *org*] [{**OU** | **organizationalUnitName**} *unit-name*] {**CN** | **commonName**} *server-name* **rsa** {**1024** | **2048** | **4096**} [**gen-object**] [**object-name** *name*] [**gen-sscert**] [**days** *number-days*] [**file-name** *name*] [**export-key**] [**export-sscert**] [**password** *plaintext*] [**password-alias** *alias*] [**using-key** *name*]

Generates a key pair on an HSM appliance

keygen [{**C** | **countryName**} *ISO-code*] [{**L** | **localityName**} *locality*] [{**ST** | **stateOrProvinceName**} *state*] [{**O** | **organizationName**} *organization*] [{**OU** | **organizationalUnitName**} *unit-name*] {**CN** | **commonName**} *server-name* **rsa** {**1024** | **2048** | **4096**} [**gen-object**] [**object-name** *name*] [**gen-sscert**] [**days** *number-days*] [**file-name** *name*] [**export-key**] [**export-sscert**] [**password** *plaintext*] [**password-alias** *alias*] [**using-key** *name*] **hsm** [**hsm-name** *name*] [**exportable** *mechanism*]

Parameters

{**C** | **countryName**} *ISO-code*
Optionally specifies the ISO two-character country identifier for the CSR.

{**L** | **localityName**} *locality*
Optionally specifies the city or town name for the CSR. Use a text string up to 64 characters in length. If the string contains spaces, enclose in double quotation marks.

{**ST** | **stateOrProvinceName**} *state*
Optionally specifies the unabbreviated state or province name for the CSR. Use a text string up to 64 characters in length. If the string contains spaces, enclose in double quotation marks.

{**O** | **organizationName**} *organization*
Optionally specifies the organization name for the CSR. Use a text string up to 64 characters in length. If the string contains spaces, enclose in double quotation marks.

{**OU** | **organizationalUnitName**} *unit-name*
Optionally specifies the organizational unit name for the CSR. Use a text string up to 64 characters in length. If the string contains spaces, enclose in double quotation marks.

{**CN** | **commonName**} *server-name*
Required. Specifies the fully qualified domain name of the server for the CSR. Use a text string up to 64 characters in length.

rsa {**1024** | **2048** | **4096**}
Indicates the length of the generated RSA key. The default is 1024.

The generation of a 4096-bit key can take up to 30 seconds.

gen-object
Creates a crypto key management object. To create a crypto certificate management object use the **gen-sscert** property.

object-name *name*
Optionally specifies the names for the objects that are created by the **gen-object** property. If not specified, the value for the **commonName** property is used.

gen-sscert
Optionally creates a self-signed certificate in addition to the private key and CSR.

days *number-days*
Optionally specifies the validity period in days for the self-signed certificate. The default is 365 days.

file-name *name*
Optionally specifies a common prefix for the generated private key, CSR, and self-signed certificate. If not specified, the value for the **object-name** property is used.

export-key
Optionally creates a copy of the private key in the temporary: directory in addition to the one in the cert: directory.

export-sscert
Optionally creates a copy of the self-signed certificate in the temporary: directory in addition to the one in the cert: directory.

password *plaintext*
Optionally specifies the password to 3DES-encrypt the private key when it is saved to a file.

password-alias *alias*
Optionally specifies a password alias in an existing password map file. This alias is used to 3DES-decrypt the password.

using-key *name*
Optionally specifies an existing key object to sign the CSR and any self-signed certificate that is generated. The point of this parameter is to reissue a new CSR or self-signed certificate with the existing key material to do the signature.

The following parameters are available on HSM-equipped appliances:

hsm Optionally creates the private key on the HSM instead of in memory.

hsm-name *name*
Optionally specifies a label for the key created on the HSM. If not specified, the value of the **object-name** parameter is used.

exportable *mechanism*
Optionally indicates the mechanism that can be used to export the imported object at a later time. Only keys will be exportable with the defined mechanism. The only supported mechanism is **hsmkwk**.

Guidelines

CA policies can vary with regard to the amount of information that is required in the CSR. Check with the CA before generating the CSR to ensure that you provide sufficient information.

Use the **password** and **password-alias** properties in environments that require password-protected files. Before using the **password-alias** property, use the **password-map** command to 3DES-encrypt the private key password (*plaintext*) and associate an alias with the encrypted password. An attempt to reference an encrypted password that is not in the password map results in command failure.

- In environments that use unencrypted passwords, the value of the **password** property is used to open and read the key file.
- In environments that use encrypted passwords, the password map file is queried for the value of the **password-alias** property, and its associated encrypted password is identified. The encrypted password, in turn, is 3DES-decrypted using the locally generated host key to yield the plaintext password that is used to open and read the key file.

Related Commands

password-map

Examples

- Generates a private key and CSR for the specified server. Default conditions apply as follows:
  - The private key (1024-bits in length) is saved as `cert:sample-privkey.pem`.
  - The CSR is saved as `temporary:sample.csr`.
  - The private key file is not password protected

```
# keygen C au L "South Melbourne" ST Victoria
 O "DataPower Australia, Ltd." OU "Customer
 Support" CN www.bob.datapower.com.au
#
```

- Generates a private key and CSR for the specified server with the following options.
  - The private key (2048-bits in length) is saved as `cert:bob-privkey.pem`.
  - The CSR is saved as `temporary:bob.csr`.
  - The private key file is password protected with the plaintext password `didgeridoo`.

```
# keygen C au L "South
 Melbourne" ST Victoria
 O "DataPower Australia, Ltd." OU "Customer
 Support" CN www.bob.datapower.com.au rsa 2048 out bob password
 didgeridoo
#
```

- Creates a new password map and generates a host key to 3DES-encrypt the plaintext password `didgeridoo`, and associates the alias `WaltzingMatilda` with the encrypted password.

  Generates a private key and CSR for the specified server with the following options.
  - The private key (2048 bits in length) is saved as `cert:bob-privkey.pem`.
  - The CSR is saved as `temporary:bob.csr`.
  - The private key file is password protected with the encrypted password `didgeridoo`.

```
# password-map
Please enter alias-name and plaintext password pairs
- Leading and trailing white space is removed
- Enter a blank alias name to finish
Alias-name: WaltzingMatilda
Plaintext password: didgeridoo
```

```
Alias-name:
SSL: password-map saved
# keygen C au
 L "South Melbourne" ST Victoria
 O "DataPower Australia, Ltd." OU "Customer Support"
 CN www.bob.datapower.com.au rsa 2048 out bob
 password-alias WaltzingMatilda
#
```

password-map

Creates a Password map, a which associates an alias with an encrypted password.

Syntax

password-map

no password-map

Guidelines

password-map interactively prompts for *alias*:*password* pairs.

alias Specifies the name of the alias. This name must consist of alphanumeric characters, and cannot contain white space; its length is limited to 127 characters.

password
Specifies the plaintext password. This password must also consist of alphanumeric characters, and may contain white space (spaces or tabs), although leading and trailing white space is ignored; its length is limited to 127 characters.

Each plaintext password is 3DES encrypted using a locally generated host key, with the final encrypted password mapped to alias-name in a password map file. The password map and the host key are saved to separate files on the appliance. The plaintext passwords are *not* stored in memory or committed to the flash.

You must ensure that synchronization is maintained between the startup configuration and the password map file. You must use the **password-map** command to generate aliases for, and encrypt, certificate or key passwords before using the **certificate** or **key** commands to access files protected by an encrypted password. An attempt to reference an encrypted password not found in the Password map results in command failure.

Deletion of the Password map and host key file has no immediate effect on keys and certificates already loaded into memory. At system restart, however, **key** and **certificate** commands that contain references to aliases contained in the deleted Password map will fail unless a new Password map has been created with the same aliases.

Note: The **password-map** command cannot be used in a configuration script. When found, the command is ignored.

Use the **no password-map** command to delete the Password map and host key files

Related Commands

certificate, **key**, **keygen**

Examples

- Creates a new Password map and generates a host key used to 3DES-encrypt the two plaintext passwords.

```
# password-map
#
Please enter alias-name and plaintext password pairs
- Leading and trailing white space is removed
- Enter a blank alias name to finish
Alias-name: towson
Plaintext password: Toshiro Mifune
Alias-name: dundaulk
Plaintext password: Tatsuya Nakadai
Alias-name:
SSL: password-map saved
#
```

- Confirms Password map creation with the **show password-map** command.

```
# show password-map
2 password-map aliases
  towson
  dundaulk
#
```

- Adds an additional alias-password pair to the Password map.

```
# password-map
A password-map already exists, overwrite it with a new map?
(y/n): n
SSL: Appending to current password map...
Please enter alias-name and plaintext password pairs
- Leading and trailing white space is removed
- Enter a blank alias name to finish
Alias-name: columbia
Plaintext password: Akiru Kurasawa
Alias-name:
SSL: password-map saved
#
```

- Confirms addition to the Password map with the **show password-map** command.

```
# show password-map
3 password-map aliases
  columbia
  towson
  dundaulk
#
```

- Deletes the Password map.

```
# no password-map
Are you sure you want to remove the password-map? (y/n): y
SSL: deleted saved password-map
#
```

- Confirms deletion with **show password-map** command.

```
# show password-map
0 password-map aliases
#
```

profile

Creates a Crypto Profile which specifies an SSL service level.

Syntax

profile *name idCred* [**ssl** *name*] [**ciphers** *cipher-string*] [**options** *options-mask*]

profile *name* **%none%** [**ssl** *name*] [**ciphers** *cipher-string*] [**options** *options-mask*]

no profile *name*

Parameters

name Specifies the name of the Crypto Profile.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

idCred Specifies the name of an Identification Credentials that was previously created with the **idcred** command. The assignment of an Identification Credentials is required when the Crypto Profile is used in conjunction with an SSL server (which is required by the SSL specification to authenticate itself to clients). The assignment of an Identification Credentials is optional when the Crypto Profile will be used in conjunction with an SSL client.

%none%
Indicates that no Identification Credentials is assigned to this Crypto Profile.

ssl *name*
Optionally specifies the name of an existing Validation Credentials that was created with the **valcred** command.

ciphers *cipher-string*
Optionally specifies a list of symmetric key-encryption algorithms that are supported by this Crypto Profile. Table 5 list the available keywords.

Table 5. Available algorithm keywords for the cipher string

Algorithm keyword	Meaning
`DEFAULT`	(Default) Includes all cipher suites except eNULL ciphers, cipher suites that use DH authentication, and all cipher suites that contain RC4, RSA, and SSL version 2 ciphers. This cipher list is determined at compile time. It is normally `ALL:!ADH:RC4+RSA:+SSLv2:@STRENGTH`. This keyword must be the first cipher string specified.
`ALL`	All cipher suites, except eNULL ciphers. eNULL ciphers must be explicitly enabled.
`HIGH`	All *high* encryption cipher suites. These cipher suites support a key length in excess of 128-bit.
`MEDIUM`	All *medium* encryption cipher suites. These cipher suites support a key length of 128-bit.
`LOW`	All *low* encryption cipher suites. These cipher suites support a key length of 56- or 64-bit, excluding `EXPORT` cipher suites.
`EXP` or `EXPORT`	Export encryption algorithms that are eligible for export out of the United States, including 40 and 56 bit algorithms.
`EXPORT40`	40-bit export encryption algorithms.
`EXPORT56`	56-bit export encryption algorithms.

Table 5. Available algorithm keywords for the cipher string (continued)

Algorithm keyword	Meaning
eNULL or NULL	NULL ciphers offer no encryption at all and are a security risk. These cipher suites are disabled unless explicitly included.
aNULL	The cipher suites offering no authentication. This is currently the anonymous DH algorithms. These cipher suites are vulnerable to *man-in-the-middle* attacks. Use is normally discouraged.
kRSA and RSA	Cipher suites using RSA key exchange.
kEDH	Cipher suites using ephemeral DH key agreement.
kDHr and kDHd	Cipher suites using DH key agreement and DH certificates signed by Certificate Authorities with RSA and DSS keys respectively. Not implemented.
aRSA	Cipher suites using RSA authentication. This is, the certificates carry RSA keys.
aDSS and DSS	Cipher suites using DSS authentication. This is, the certificates carry DSS keys.
aDH	Cipher suites effectively using DH authentication. This is, the certificates carry DH keys. Not implemented.
kFZA, aFZA, eFZA or, FZA	Cipher suites using FORTEZZA key exchange, authentication, encryption, or all FORTEZZA algorithms. Not implemented.
TLSv1, SSLv3, and SSLv2	TLS version 1.0, SSL version 3.0, and SSL version 2.0 cipher suites, respectively.
DH	Cipher suites using DH, including anonymous DH.
ADH	Anonymous DH cipher suites.
3DES	Cipher suites using triple DES.
DES	Cipher suites using DES, except triple DES.
RC4	Cipher suites using RC4.
RC2	Cipher suites using RC2.
IDEA	Cipher suites using IDEA.
MD5	Cipher suites using MD5.
SHA1 or SHA	Cipher suites using SHA-1.
AES	Cipher suites using AES.

The cipher string consists of one or more cipher keywords separated by colons. Commas or spaces are acceptable separators, but colons are the norm.

The cipher string can take different forms.

- A single cipher suite, such as RC4-SHA.
- A list of cipher suites that contains a certain algorithm, or cipher suites of a certain type. For example SHA1 represents all ciphers suites using the SHA-1 digest algorithm.
- A combination of single cipher string using the + character, which is used as a logical AND operation. For example SHA1+DES represents all cipher suites that contain the SHA-1 and the DES algorithms.

Optionally, each cipher keyword can be preceded by the following characters:

! Permanently deletes the cipher from the list. Even if you explicitly add the cipher to the list, it can never reappear in the list.

- Deletes the cipher from the list. You can add this cipher again.

+ Moves the cipher to the end of the list. The + character moves existing ciphers, it does not add them.

If none of these characters is present, the string is interpreted as a list of ciphers to be appended to the current list. If the list includes a cipher that is already in the list, that cipher is ignored. That is, existing ciphers are not moved to the end of the list.

Additionally, the cipher string can contain the `@STRENGTH` keyword at any point to sort the cipher list in order of encryption algorithm key length.

options *options-mask*
Optionally enables various SSL options for the Crypto Profile. Use the string or specify a hexadecimal representation of a 32-bit mask string that identifies specific supported SSL options. Table 6 lists the available options.

Table 6. SSL options as string and hexadecimal representation

String value	Hexadecimal representation	Description
`OpenSSL-default`	`0x000FFFFF`	Default value
`Disable-SSLv2`	`0x01000000`	Disallows the use of SSL version 2
`Disable-SSLv3`	`0x02000000`	Disallows the use of SSL version 3
`Disable-TLSv1`	`0x04000000`	Disallows the use of TLS version 3

When using hexadecimal representation, use a logical `OR` to modify the behavior during the SSL handshake. When using the string value, use a + character to join values. For example, to disallow both SSL version 2 and TLS version 1, enter one following values:

Hexadecimal
`0x05000000`

String `Disable-SSLv2+DisableTLSv1`

Guidelines

A Crypto Profile defines a level of SSL service. When you create an SSL Proxy Profile with the **sslproxy** command, you assign a Crypto Profile to the SSL Proxy Profile.

Before creating a Crypto Profile to use with an SSL server, use the **certificate** command with the **key** and **idcred** commands to create an Identification Credentials. This set of credentials consists of a certificate, which contains a public key, and the corresponding private key.

A Crypto Profile optionally uses a Validation Credentials to validate certificates that are received from remote SSL peers.

- The SSL client requires a Validation Credentials only when it validates the certificate that is presented by an SSL server. The SSL standard does not require the validation of the server certificate.
- The SSL server requires a Validation Credentials only when it validates certificates that are presented by SSL clients. The SSL standard does not require the validation of SSL clients.

If you want the SSL service to validate received certificates:

1. Use the **valcred** and **certificate** (Validation Credentials) commands to create a Validation Credentials.
2. Assign the Validation Credentials to the Crypto Profile.

Assignment of a Validation Credentials to a Crypto Profile mandates that SSL validates the certificate that is presented by the remote peer. If the peer fails to present a certificate on request or presents a certificate that cannot be validated, the Crypto Profile requires the termination of the SSL connection.

Note: In the absence of the **ssl** keyword, the Crypto Profile performs no SSL peer authentication.

The **no profile** command deletes only the specified Crypto Profile. The alias names that are used to create the original Crypto Profile remain available for use, as do as the files that contain the actual certificates and private keys that are used to implement the Crypto Profile.

Related Commands

certificate (Crypto), **certificate** (Validation Credentials), **idcred**, **sslproxy**, **valcred**

Examples

- Creates the `Low` Crypto Profile that uses the Identification Credentials (certificate and private key) aliased by `XSSL-1` to identify the SSL proxy. The Crypto Profile specifies no validation of received peer certificates and supports the `DEFAULT` cipher list.

```
# profile Low XSSL-1
Creating new crypto profile 'Low'
#
```

- Creates the `Low` Crypto Profile that uses the Identification Credentials aliased by `XSSL-1` to identify the SSL proxy. The Crypto Profile specifies no peer validation, supports the `DEFAULT` cipher list, and disables SSL Version 2.

```
# profile Low XSSL-1
options 0x01000000
Creating new crypto profile 'Low'
#
```

- Same as the previous example.

```
# profile Low XSSL-1
options Disable-SSLv2
Creating new crypto profile 'Low'
#
```

- Creates the `Low` Crypto Profile that uses the Identification Credentials aliased by `XSSL-1` to identify the SSL proxy. The Crypto Profile specifies no peer validation, supports the `DEFAULT` cipher list, and disables SSL Version 2 and TLS Version 1.

```
# profile Low XSSL-1
options 0x05000000
Creating new crypto profile 'Low'
#
```

- Same as the previous example.

```
# profile Low XSSL-1
options Disable-SSLv2+DisableTLSv1
Creating new crypto profile 'Low'
#
```

- Creates the `High` Crypto Profile that uses the Identification Credentials aliased by `XSSL-2` to identify the SSL proxy. The Crypto Profile validates the SSL peer with the `TSC-1` validation credentials, and supports symmetric encryption algorithms with key lengths of 128 bits or more.

```
# profile High XSSL-2
 ssl TSC-1 ciphers HIGH
Creating new crypto profile 'High'
#
```

- Deletes the `High` Crypto Profile.

```
# no profile High
Crypto Profile 'High' deleted
#
```

sign

Signs a file stored on the appliance.

Syntax

sign *URL* **idcred** *alias* **alg** *algorithm*

Parameters

URL Identifies the local file to be signed, and takes the *directory*`:///`*filename* format.

directory
Must be one of the following directory-specific keywords that reference specific directories:

`audit:` Contains the audit log

`cert:` Contains domain-specific private keys and certificates

`config:`
Contains configuration scripts

`export:`
Contains export packages

`image:` Contains primary and secondary firmware images

`local:` Contains user processing resources such as style sheets, schemas, document encryption maps, or XML mapping files

`logstore:`
Contains logging files

`logtemp:`
Contains active and rotated log files

`pubcert:`
Contains well-known (for example, VeriSign) public certificate files

`sharedcert:`
Contains private keys and certificates which are shared across domains

`store:` Contains DataPower-supplied processing resources such as style sheets, schemas and authentication/authorization files

`tasktemplates:`
Contains Task Template files

`temporary:`
Contains temporary files

filename
Specifies the name of the file to sign.

idcred *alias*
Specifies and existing alias for an Identification Credentials (a matched public/private key pair) used to identify the *identification-set-alias* references the *local* private key used to sign a file.

alg *algorithm*
Identifies the signature method.

Guidelines

Before encrypting a file, use the **sign** command to attach a digital signature to the file. You can email an encrypted file with the **send file** command.

Related Commands

validate

Examples

- Use S/MIME to sign the `FWSec-1` log file. The uses S/MIME to encrypt the signed file.

```
# sign logtemp:///FWSec-1 idcred bob alg smime
File 'FWSec-1' successfully signed
# encrypt logtemp:///FWSec-1 cert bob alg smime
File 'FWSec-1' successfully encoded
```

sskey

Creates an alias for a shared secret key.

Syntax

sskey *key-alias URL* [**password** *password*]

sskey *key-alias URL* [**password-alias** *password-alias*]

no sskey *key-alias*

Parameters

key-alias
Specifies an alias for the stored shared secret key.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

URL Specifies a local URL that identifies the file that contains the private key.

- If the private key is stored in the private cryptographic area, the URL takes the *filename* form.
- If the private key is stored in the public cryptographic area, the URL takes the pubcert:///*filename* form.

Note: Do not store private key files in the public cryptographic area. This area is intended for the storage of certificate files that are publicly available.

password *password*
Specifies the plaintext password required to access the shared secret key file. Required when a shared secret key file is password-protected.

password-alias *password-alias*
Specifies the alias for the encrypted password required to access the shared secret key file. Required when a shared secret key file is password-protected.

Guidelines

Before using the **password-alias** keyword, you must use the **password-map** command to 3DES-encrypt the shared secret key password and associate an alias with the encrypted password. An attempt to reference an encrypted password not found in the Password map results in command failure.

- In environments that utilize plaintext (unencrypted) passwords, the *password* argument is used to open and read the shared secret key file.
- In environments that utilize encrypted passwords, the *password-alias* argument is searched for in the password map file and its associated encrypted password is identified. The encrypted password in turn is then 3DES-decrypted (using the locally generated host key) to yield the plaintext password used to open and read the shared secret key file.

Use the **sskey** command in conjunction with the **certificate** and **idcred** commands to create an Identification Credentials that consists of a certificate, which contains a public key, and the corresponding private key.

The **no sskey** command deletes only the alias for the stored shared secret key. The file that contains the actual shared secret key remains on the appliance.

Related Commands

password-map

Examples

- Creates the `alice` alias for the `SS2.pem` shared secret key. The target key is contained within the private cryptographic area.

```
# sskey alice SS2.pem
Creating key 'alice'
#
```

- Creates the `alice` alias the `SS2.pem` shared secret key. The target key is contained within the private cryptographic area, and is accessed with the `oceanCity` plaintext password.

```
# sskey alice SS2.pem
password oceanCity
Creating key 'alice'
#
```

- Creates the `alice` alias the specified `SS2.pem` secret key. The target key is contained within the private cryptographic area, and is accessed with an encrypted password aliased by `HavredeGrace`.

```
# sskey alice SS2.pem
password-alias HavredeGrace
Creating key 'alice'
#
```

- Deletes the `alice` shared secret key alias.

```
# no sskey alice
Key 'alice' deleted
#
```

---

test password-map

Tests the association between an encrypted password alias and a file. Confirms or denies that the alias references the password that protects the file.

Syntax

test password-map *alias type URL*

Parameters

alias Specifies the name of the candidate alias.

type Identifies the file type. Use the value **key** or **cert**.

URL Specifies a local URL that identifies the file that contains the certificate or key.

- If stored in the public cryptographic area, takes the `pubcert:`*`filename`* form.
- If stored in the private cryptographic area, takes the *`filename`* form.

Guidelines

Assuming syntactical correctness, testing a key or certificate file that does not require a password will succeed in all cases.

Note: The **test password-map** command cannot be used in a startup configuration. If found, the script ignores the command.

Related Commands

certificate, **key**, **password-map**

Examples

- Indicates that the `towson` candidate alias does not reference the encrypted password that protects the `dpSupplied.der` certificate file.

```
# test
password-map towson cert pubcert:dpSupplied.der
Alias 'towson' with file 'pubcert:dpSupplied.der' --> FAIL
#
```

- Indicates that the `dundaulk` candidate alias does reference the encrypted password that protects the `dpSupplied.der` certificate file.

```
# test password-map dundaulk
cert pubcert:dpSupplied.der
Alias 'dundaulk' with file 'pubcert:dpSupplied.der' --> OK
#
```

- Indicates that the `columbia` candidate alias does not reference the encrypted password that protects the `K2.der` key file.

```
# test password-map columbia
key K2.der
Alias 'columbia' with file 'K2.der' --> FAIL
#
```

- Indicates that the `towson` candidate alias does reference the encrypted password that protects the `K2.der` key file.

```
# test password-map towson
key K2.der
Alias 'towson' with file 'K2.der' --> OK
#
```

valcred

Enters Validation Credentials mode.

Syntax

valcred *name*

no valcred *name*

Parameters

name Specifies the name of the Validation Credentials.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

A Crypto Profile optionally uses a Validation Credentials to validate certificates received from remote SSL peers.

- A Validation Credentials is required by an SSL client only when it validates the certificate presented by an SSL server. Validation of a server's certificate is not required by the SSL standard.
- A Validation Credentials is required by an SSL server only when it validates certificates presented by SSL clients. Validation of SSL clients is not required by the SSL standard.

If you want the SSL service to validate received certificates:

- Use the **valcred** and **certificate** (Validation Credentials) commands to create a Validation Credentials.
- Assign the Validation Credentials to the Crypto Profile.

Assignment of a Validation Credentials to a Crypto Profile mandates that SSL validate the certificate presented by the remote peer. If the peer fails to present a certificate upon request, or presents a certificate that cannot be validated, the Crypto Profile requires the termination of the SSL connection.

The **no valcred** command deletes only the named Validation Credentials. The certificate aliases that appeared in the list remain available for use, as do as the files that contain the actual certificates.

Related Commands

certificate (Validation Credentials), **profile**

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials.

```
# valcred ValCred-1
Entering Validation Credentials mode for 'ValCred-1'
#
```

- Deletes the `ValCred-1` Validation Credentials.

```
# no valcred ValCred-1
Validation Credentials 'ValCred-11' deleted
#
```

---

validate

Validates the digital signature of a specified file.

Syntax

validate *URL* **valcred** *name* **alg** *algorithm*

Parameters

URL Identifies the local file whose digital signature is to be verified, and takes the *directory*`:///`*filename* form, where:

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

filename
Specifies the name of a file in the specified directory.

valcred *name*
Specifies the name of a Validation Credentials (a certificate list).

alg *algorithm*
Identifies the signature method.

Guidelines

The validation process attempts to validate the digital signature of the target document by iterating through the certificates that are referenced by the specified Validation Credentials.

Related Commands

sign, **valcred**

Examples

- Uses the `Signers` Validation Credentials to validate the digital signature of the `store:///SchemaNew.xsd` schema with the S/MIME algorithm.

```
# validate store:///SchemaNew.xsd valcred Signers alg smime
 'SchemaNew.xsd' successfully validated.
#
```

Chapter 12. Crypto Certificate Monitor configuration mode

This chapter provides an alphabetic listing of commands that are available in Crypto Certificate Monitor configuration mode. To enter this configuration mode, use the crypto **cert-monitor** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Crypto Certificate Monitor configuration mode.

disable-expired-certs

Specifies system usage of an expired certificate.

Syntax

disable-expired-certs {**on** | **off**}

Parameters

on Specifies that on certificate expiration all objects that use the expired certificate (either directly or through inheritance) are disabled and are no longer in service. For example, certificate expiration triggers the disable of the associated crypto certificate object. Disable of the crypto certificate object triggers disable of all Firewall Credential Lists, Identification Credential Sets, and Validation Credential Lists that use the expired certificate. In turn crypto profiles that use disabled Identification Credential Sets and Validation Credential Lists are disabled, leading to the disable of SSL Proxy Profiles dependent on now-disabled crypto profiles. Ultimately XML Firewalls, XSL Proxies, and XML managers may be disabled as the result of certificate expiration.

off (Default) Specifies that the certificate object and objects using the expired certificate are not disabled upon certificate expiration.

Examples

- Specifies that all objects that use or reference a certificate are disabled on certificate expiration.

```
# disable-expired-certs on
#
```

- Restores the default state. Objects that use or refer to a certificate are not disabled on certificate expiration.

```
# disable-expired-certs off
#
```

or

```
# no disable-expired-certs
#
```

log-level

Specifies the log priority assigned to certificate monitor messages that note the impending expiration date of a certificate

Syntax

log-level *priority*

Parameters

priority
Specifies the log priority assigned to certificate expiration messages.

Guidelines

The level of log events are characterized (in descending order of criticality) as:

- `emergency`
- `alert`
- `critical`
- `error`
- `warning`
- `notice`
- `info`

The `debug` event is used for system troubleshooting and diagnostics. Do not use the `debug` event in production environments.

Related Commands

reminder

Examples

- Specifies that certificate expiration messages are logged as errors.

```
# log-level error
#
```

poll

Specifies the frequency with which the Certificate Monitor examines certificate object expiration dates.

Syntax

poll *frequency*

Parameters

frequency
Specifies the number of days between Certificate Monitor scans. Use an integer in the range of 1 through 65535.

Related Commands

reminder

Examples

- Specifies that the Certificate Monitor performs a certificate scan every 3 days.

```
# poll 3
#
```

reminder

Specifies the notification window before certificate expiration that initiates certificate expiration log messages.

Syntax

reminder *days*

Parameters

days Specifies the notification window. Use an integer in the range of 1 through 65535.

Guidelines

For example, the value 21 specifies that all scanned certificate objects due to expire in 3 weeks or less generate a log entry at the priority specified by the **log-level** command.

Related Commands

log-level, **poll**

Examples

- Specifies that the Certificate Monitor begins issuing certificate expiration messages 21 days before certificate expiration.

```
# reminder 21
#
```

Chapter 13. Crypto Firewall Credentials configuration mode

This chapter provides an alphabetic listing of commands that are available in Crypto Firewall Credentials configuration mode.

To enter this configuration mode, use the Crypto **fwcred** command. While in this configuration mode, you can create a Firewall Credentials list that identifies which keys and certificates available to a DataPower appliance to support firewall processing.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Crypto Firewall Credentials configuration mode.

certificate

Adds a certificate alias.

Syntax

certificate *alias*

Parameters

alias Specifies the alias for the target certificate. The target certificate must be previously created with the Crypto **certificate** command.

Guidelines

Prior to adding a certificate alias to the Firewall Credentials list:

1. Use the **copy** command (or the WebGUI) to transfer the actual certificate to the appliance.
2. Use the Crypto **certificate** command to create a certificate alias.

Use the **no certificate** command to delete a certificate alias from a Firewall Credentials List.

Related Commands

certificate (Crypto)

Examples

- Enters Firewall Credentials mode for the `FWCred-1` Firewall Credentials. Adds the certificate that is referenced by the `alice-3` alias.

```
# fwcred FWCred-1
Entering Firewall Credentials mode for 'FWCred-1'
# certificate alice-3
#
```

key

Adds a key alias.

Syntax

key *alias*

Parameters

alias Specifies the alias for the target private key. The target private key must be previously created with the Crypto **key** command.

Guidelines

Prior to adding a key alias to the list:

1. Use the **copy** command (or the WebGUI) to transfer the actual key to the appliance.
2. Use the Crypto **key** command to create a key alias.

Use the **no key** command to delete a key alias from a Firewall Credentials List.

Related Commands

key (Crypto)

Examples

- Enters Firewall Credentials mode for the `FWCred-1` Firewall Credentials. Adds the key that is referenced by the `alice-3` alias.

```
# fwcred FWCred-1
Entering Firewall Credentials mode for 'FWCred-1'
# key alice-3
#
```

---

sskey

Adds a shared secret key alias.

Syntax

sskey *alias*

Parameters

alias Specifies the alias for the target shared-secret key. The target shared-secret key must be previously created with the Crypto **sskey** command.

Guidelines

Prior to adding a shared secret key alias to the list:

1. Use the **copy** command (or the WebGUI) to transfer the actual shared secret key to appliance.
2. Use the Crypto **sskey** command to create a shared secret key alias.

Use the **no sskey** command to delete a shared secret key alias from a Firewall Credentials List.

Related Commands

sskey (Crypto)

Examples

- Enters Firewall Credentials mode for the `FWCred-1` Firewall Credentials. Adds the shared secret key that is referenced by the `ss-bob-alice` alias.

```
# fwcred FWCred-1
Entering Firewall Credentials mode for 'FWCred-1'
# sskey ss-bob-alice
#
```

Chapter 14. Crypto Validation Credentials configuration mode

This chapter provides an alphabetic listing of commands that are available in Crypto Validation Credentials configuration mode.

To enter this configuration mode, use the Crypto **valcred** command. While in this mode, compile a Validation Credentials List to validate credentials that are presented by an SSL peer.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

cert-validation-mode

Specifies the method, in conjunction with the current Validation Credentials List, to perform certificate validation.

Syntax

cert-validation-mode {**legacy** | **pkix**}

no cert-validation-mode

Parameters

legacy (Default) The behavior is that the Validation Credentials contains either the exact peer certificate to match or the certificate of the immediate issuer, which could be an intermediate CA or a root CA. This mode is useful when you want to match the peer certificate exactly, but that certificate is not a self-signed (root) certificate.

pkix The complete certificate chain is checked from subject to root when using this Validation Credentials for certificate validation. Validation succeeds only if the chain ends with a root certificate in the Validation Credentials. Non-root certificates in the Validation Credentials will be used as untrusted intermediate certificates. Additional untrusted intermediate certificates will be obtained dynamically from the context at hand (SSL handshake messages, PKCS#7 tokens, PKIPath tokens, and so forth).

Guidelines

The **pkix** method, as described in RFC 3280, expects the remote peer to provide all intermediate certificates to the DataPower appliance during SSL negotiation. The associated Validation Credentials List consists of self-signed certificates and certificates of trust anchors. Certificates can be a root CA or an intermediate CA.

Use the **no cert-validation-mode** command to delete a certificate alias from a Validation Credentials List.

Related Commands

certificate (Crypto)

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Specifies PKIX validation mode.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# cert-validation-mode pkix
#
```

- Restores the default setting.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# cert-validation-mode legacy
#
```

certificate

Adds a certificate alias to the current Validation Credentials List.

Syntax

certificate *alias*

no certificate *alias*

Parameters

alias Specifies the name of an existing alias for the target certificate. If the alias does not already exist, use the Crypto **certificate** command.

Guidelines

A Crypto Profile optionally uses a Validation Credentials List to authenticate a remote SSL peer.

- A Validation Credentials List is required by an SSL client only when it authenticates the certificate presented by the remote SSL server. Authentication of the server's certificate is not required by the SSL standard.
- A Validation Credentials List is required by an SSL server only when it authenticates remote SSL clients. Authentication of SSL clients is not required by the SSL standard.

Assignment of a Validation Credentials List to a Crypto Profile requires that SSL validates the certificate that is presented by the remote peer. If the peer fails to present a certificate on request or presents a certificate that cannot be validated, the Crypto Profile requires that the SSL connection be terminated.

Prior to adding a certificate-alias to the Validation Credentials List you must:

1. Use the **copy** command or the WebGUI to transfer the certificate to the appliance.
2. Use the Crypto **certificate** command to create an alias for the certificate.

Use the **no certificate** command to delete a certificate alias from a Validation Credentials List.

Related Commands

certificate (Crypto), **valcred**

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Adds the `bob-1` certificate alias to the list.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# certificate bob-1
#
```

crldp

Controls support for the X.509 Certificate Distribution Point certificate extension.

Syntax

crldp {**ignore** | **require**}

Parameters

ignore (Default) Ignores the certificate extension.

require
Indicates that a candidate certificate is deemed valid if the presented certificate chain is terminated by a trust anchor. This method is used only when if the current Validation Credentials List is used for SSL peer validation.

Guidelines

This noncritical certificate extension specifies how CRL information is obtained.

Refer to RFC 2527 *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework* and to RFC 3280 *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* for information on Certificate Policies.

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Enables support the Certificate Distribution Point extension.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# crldp require
#
```

- Restores the default state.

```
Crypto Validation Credentials configuration mode
# crldp ignore
#
```

explicit-policy

Controls support for the initial-explicit-policy variable.

Syntax

explicit-policy

no explicit-policy

Guidelines

Meaningful only if **cert-validation mode** is **pkix**; otherwise, it is not used.

If enabled, the chain validation algorithm must end with a non-empty policy tree. If disabled, the algorithm may end with an empty policy tree (unless Policy Constraints extensions in the chain require an explicit policy).

Refer to RFC 2527 *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework* and to RFC 3280 *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* for information on Certificate Policies.

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Specifies the chain validation algorithm must end with an empty tree.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# explicit-policy
#
```

- Restores the default state.

```
Crypto Validation Credentials configuration mode
# no explicit-policy
#
```

---

initial-policy-set

Identifies a Certificate Policy used by the current Validation Credentials List.

Syntax

initial-policy-set *identifier*

no initial-policy-set *identifier*

Parameters

identifier
: Specifies the unique object identifier for the certificate policy associated with the current Validation Credentials List.

Guidelines

Refer to RFC 2527 *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework* and to RFC 3280 *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* for information on Certificate Policies.

Meaningful only if **cert-validation mode** is **pkix** and otherwise unused.

RFC 2527 defines a Certificate Policy is follows:

> a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements. For example, a particular certificate policy might indicate

applicability of a type of certificate to the authentication of electronic data interchange transactions for the trading of goods within a given price range.

The certificate policies extension contains a sequence of one or more policy information terms, each of which consists of an object identifier (OID) and optional qualifiers.

Note: The use of qualifiers is not supported. If present, they are ignored.

In a host certificate, policy information terms indicate the policy under which the certificate has been issued and the purposes for which the certificate may be used.

In a CA certificate, policy information terms limit the set of policies for certification paths which include this certificate. When a CA does not wish to limit the set of policies for certification paths which include this certificate, it may assert the special policy, **anyPolicy**, with an OID of 2.5.29.32.0.

You use this command as often as needed to construct a set of Certificate Policy identifiers; by default, the initial Certificate Policy Set consists of the single OID 2.5.29.32.0, identifying **anyPolicy**.

All members of the constructed set are used in certificate chain processing as described in Section 6.1.1 of RFC 3280.

Use the **no initial-policy-set** command to remove a Certificate Policy from the Validation Credentials List.

Related Commands

explicit-policy, **inhibit-anypolicy**

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Adds the specified OID to the set of Certificate Policy identifiers associated with the current Validation Credentials List.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# initial-policy-set 1.3.6.1.4.1.14248.1.1
#
```

- Removes the specified OID from the set of Certificate Policy identifiers associated with the current Validation Credentials List.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# no initial-policy-set 1.3.6.1.4.1.14248.1.1
#
```

require-crl

Mandates the use of Certificate Revocation Lists during certificate chain processing.

Syntax

require-crl

no require-crl

Guidelines

By default, CRL usage is not required when processing certificate chains.

Use the **no require-crl** command to restore the default condition, which allows, but does not require, CRL usage when processing certificate chains.

Related Commands

use-crl

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Requires CRL usage during certificate chain processing.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# require-crl
#
```

- Restores the default setting.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# no-require-crl
#
```

---

use-crl

Enables but does not require the use of Certificate Revocation Lists during certificate chain processing.

Syntax

use-crl

no use-crl

Guidelines

By default, CRL usage is enabled when processing certificate chains.

Use the **no use-crl** command to disable the use of Certificate Revocation Lists during certificate chain processing.

Related Commands

crl (Crypto), **require-crl**

Examples

- Enters Validation Credentials Mode to create the `ValCred-1` Validation Credentials List. Disables CRL usage during certificate chain processing.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# no use-crl
#
```

- Restores the default setting.

```
# valcred ValCred-1
Crypto Validation Credentials configuration mode
# use-crl
#
```

Chapter 15. Deployment Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Deployment Policy configuration mode. A deployment policy is a sequence of accept, filter, and modify clauses that accept, filter, or modify configuration data of specific resources on the appliance.

To enter this configuration mode, use the Global **deployment-policy** command. While in this mode, select portions of imported or included configuration to add to the running configuration.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Deployment Policy configuration mode.

??? accept

Defines an accept clause.

Syntax

accept *statement*

Parameters

statement

Specifies a cumulative white list used as a pattern match in a deployment policy. This PCRE statement represents the metadata whose values the deployment policy will match, and then accept or allow into the appliance while searching the contents of the imported configuration file. Accept clauses allow properties into the appliance.

The statement takes the following form:

```
address/domain/resource[?Name=resource-name
  &Property=property-name&Value=property-value]
```

address

Specifies the IP address or host alias. The value * matches all IP addresses.

domain Specifies the name of the application domain. The value * matches all domains.

resource

Specifies the resource type. The value * matches all resource type.

Name=*resource-name*

Optionally specifies a name match for a resource. This property limits the match statement to resources of the specified name. Use a PCRE to select groups of resource instances. For example, `foo*` would match all resources with names that start with `foo`.

`Property=property-name`
Optionally specifies the name of the configuration property. This property limits the match statement to resources of the specified property.

`Value=property-value`
Optionally specifies the value for the configuration property. This property limits the match statement to resources of the specified property.

PCRE documentation is available at the following web site:

http://www.pcre.org

Guidelines

The **accept** command defines a white list of metadata that is accepted or allowed into the appliance.

Related Commands

import-execute (global)

Examples

- ??? Defines an accept clause for a deployment policy where the matching statement is made up of the `10.10.10.1:8888` IP address, the `jets` domain, the `Cessnae` Load Balancer Group. If this resource is part of the exported configuration package, this resource is accepted during the import.

```
# accept 10.10.10.1:8888/jets/network/loadbalancer-group?Name=Cessnae
#
```

??? filter

Defines a filter clause.

Syntax

filter *statement*

Parameters

statement
Specifies a cumulative black list used as a pattern match in a deployment policy. This PCRE statement represents the metadata whose values the deployment policy will match, and not accept or disallow into the appliance while searching the contents of the imported configuration file. Filter clauses disallow properties into the appliance.

The statement takes the following form:

address`/`*domain*`/`*resource*`[?Name=`*resource-name*
`&Property=`*property-name*`&Value=`*property-value*`]`

address
Specifies the IP address or host alias. The value `*` matches all IP addresses.

domain Specifies the name of the application domain. The value `*` matches all domains.

resource
Specifies the resource type. The value * matches all resource type.

Name=*resource-name*
Optionally specifies a name match for a resource. This property limits the match statement to resources of the specified name. Use a PCRE to select groups of resource instances. For example, foo* would match all resources with names that start with foo.

Property=*property-name*
Optionally specifies the name of the configuration property. This property limits the match statement to resources of the specified property.

Value=*property-value*
Optionally specifies the value for the configuration property. This property limits the match statement to resources of the specified property.

PCRE documentation is available at the following web site:

http://www.pcre.org

Guidelines

The **filter** command defines a black list of metadata that is not accepted or disallowed into the appliance.

Related Commands

import-execute (global)

Examples

- ??? Defines a filter clause for a deployment policy where the matching statement is made up of the 10.10.10.1:8888 IP address, the jets domain, the Cessnae Load Balancer Group. If this resource is part of the exported configuration package, this resource is not included (filtered) during the import.

```
# accept 10.10.10.1:8888/jets/network/loadbalancer-group?Name=Cessnae
#
```

??? modify

Defines a modify clause.

Syntax

modify *statement* **add** *property value*

modify *statement* **change** *value*

modify *statement* **delete**

Parameters

statement
Specifies a list used as a pattern match in a deployment policy. This PCRE statement represents the metadata whose values the deployment policy will match while searching the contents of the imported configuration file.

The appliance preprocesses the add statements first, the change statements second, and the delete statements last when applying the modify clause.

The statement takes the following form:

address/*domain*/*resource*[?Name=*resource-name*
  &Property=*property-name*&Value=*property-value*]

address
: Specifies the IP address or host alias. The value * matches all IP addresses.

domain Specifies the name of the application domain. The value * matches all domains.

resource
: Specifies the resource type. The value * matches all resource type.

Name=*resource-name*
: Optionally specifies a name match for a resource. This property limits the match statement to resources of the specified name. Use a PCRE to select groups of resource instances. For example, foo* would match all resources with names that start with foo.

Property=*property-name*
: Optionally specifies the name of the configuration property. This property limits the match statement to resources of the specified property.

Value=*property-value*
: Optionally specifies the value for the configuration property. This property limits the match statement to resources of the specified property.

PCRE documentation is available at the following web site:

http://www.pcre.org

add Indicates that the modify clause adds the identified property to the configuration.

change
: Indicates that the modify clause changes the value for the identified property.

delete Indicates that the modify clause deletes the identified property.

property
: Identifies the name of the property to add to the configuration.

value Identifies the value for the property to add to the configuration or to modify in the configuration.

Guidelines

The **modify** command defines a modify clause. This clause modifies properties in the configuration of an imported configuration package. Properties for a configuration can be added, can be modified, or can be deleted.

Related Commands

import-execute (global)

Examples

- ??? Adds a summary to the Turbotans host alias in the default domain. The UserSummary property with a value of BlueSkinners is added to the configuration of the Turbotans host alias during the import.

```
# modify */default/network/host-alias?Name=Turbotans
   add UserSummary BlueSkinners
#
```

- ??? Changes the value of the summary for the Turbotans host alias in the default domain to Turbotans5 during the import.

```
# modify */default/network/host-alias?Name=Turbotans
   &Property=UserSummary&Value=BlueSkinners change
   Turbotans5
#
```

- ??? Deletes the summary for the Turbotans host alias in the default domain during the import.

```
# modify */default/network/host-alias?Name=Turbotans
   &Property=UserSummary delete
#
```

Chapter 16. DNS Settings configuration mode

This chapter provides an alphabetic listing of commands that are available in DNS (Domain Name Services) configuration mode.

To enter this configuration mode, use the Global **dns** command. While in this mode, identify sources of DNS information and perform static mapping of host names to IP addresses.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in DNS Setting configuration mode.

name-server

Identifies a local DNS provider.

Syntax

name-server *address* [*UDP-port*] [*TCP-port*] [*flags*] [*count*]

no name-server {*address* | *}

Parameters

address Specifies the IP address or host name of the DNS server.

UDP-port
Optionally identifies the UDP port number that the target DNS server monitors. Use an integer in the range of 0 through 65535. The default is 53.

TCP-port
Optionally identifies the TCP port number that the target DNS server monitors. Use an integer in the range of 0 through 65535. The default is 53.

flags Optionally specifies protocol-level DNS behavior and should be set to 0.

count Optionally specifies the maximum number of times to retransmit an unacknowledged resolution request to this DNS server. The default is 3.

* Specifies all DNS servers.

Guidelines

Use the **no name-server** command to delete a DNS provider.

Note: Unless explicitly instructed, do not change the value of the DNS parameter.

Related Commands

ip name-server

Examples

- Identifies 10.10.10.240:53 (the well-known DNS port) as a DNS provider.

```
# name-server 10.10.10.240
#
```

- Identifies a DNS server at 10.10.10.240 UDP port 60000.

```
# name-server 10.10.10.240 60000
#
```

- Deletes the specified DNS provider.

```
# no name-server 10.10.10.240
#
```

- Deletes all DNS providers.

```
# no name-server *
#
```

---

search-domain

Adds an entry to the IP domain-suffix search table, thus enabling the usage of non-fully qualified domain names.

Syntax

search-domain *domain*

no search-domain *domain*

Parameters

domain Specifies a base domain name to which a host name can be prefixed.

Guidelines

The **search-domain** and **ip domain** commands both enable the usage on non-fully qualified domain names (host names) by specifying a list of one or more domain names that can be appended to a host name.

Use multiple **search-domain** or **ip domain** commands to add more than one entry to the IP domain name table.

The appliance attempts to resolve a host name in conjunction with any domains identified by the **search-domain** or **ip domain** commands. The host name is resolved as soon as a match is found.

Use the **no search-domain** command to delete an entry from the table.

Related Commands

ip domain

Examples

- Adds the `datapower.com`, `somewhereelse.com`, and `endoftheearth.com` IP domains to the IP domain table. The appliance attempts to resolve the host name `loki` in following ways:
  - `loki.datapower.com`
  - `loki.somewhereelse.com`
  - `loki.endoftheearth.com`

```
# search-domain datapower.com
# search-domain somewhereelse.com
# search-domain endoftheearth.com
# exit
```

```
# xslproxy Proxy-01
XSL proxy configuration mode
# remote-address loki 80
#
```

static-host

Maps a host name to an IP address.

Syntax

static-host *hostname address*

no static-host {*hostname* | *}

Parameters

hostname
: Identifies a specific host.

address
: Specifies the IP address of the host.

*
: Specifies all hosts.

Guidelines

Use the **no static-host** command to remove the a specific host-address map or all host-address maps.

Related Commands

ip host

Examples

- Maps IP address 10.10.10.168 to host `loki`.

  ```
  # static-host loki 10.10.10.168
  #
  ```

- Deletes the mapping between IP address 10.10.10.168 and host `loki`.

  ```
  # no static-host loki
  #
  ```

- Deletes all entries from the host mapping table.

  ```
  # no static-host *
  #
  ```

Chapter 17. Document Cache configuration mode

This chapter provides an alphabetic listing of commands that are available in Document Cache configuration mode.

To enter this configuration mode, use the Global **documentcache** command. While in this mode, define the policy that specifies which HTTP-obtained documents the associated XML Manager caches.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Document Cache configuration mode.

clear

Clears specified documents from the document cache.

Syntax

clear

clear *pattern*

Parameters

pattern An shell-style match pattern that identifies documents.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Guidelines

In the absence of the *pattern* argument, removes all files from the document cache.

Related Commands

policy

Examples

- Removes all documents from the document cache

```
# clear
Cleared documents in cache matching pattern *
#
```

- Removes all XML schemas and XSL style sheets from the document cache

```
# clear *xs[dl]
Cleared documents in cache matching pattern *xs[dl]
#
```

maxdocs

Specifies the maximum size of the document cache in documents.

Syntax

maxdocs *documents*

Parameters

documents
Specifies the maximum number of documents to retain in the document cache. Use an integer in the range of 1 through 250000. The default is 5000.

Guidelines

Retain the default value of 5000 documents.

Related Commands

size

Examples

- Specifies a maximum cache capacity of 4,000 documents.

```
# maxdocs 40000
Ok. Document cache size is now limited to 4000 documents
#
```

-

policy

Defines a Document Cache Policy.

Syntax

policy *pattern* [*priority*] [*ttl*]

policy *pattern* [*priority*] [**nocache**]

no policy [*pattern*]

Parameters

pattern A shell-style match pattern that identifies documents.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

priority
Specifies the priority of a document in the cache. The greater the value, the higher its priority. Use an integer in the range of 1 through 255. The default is 128.

ttl Specifies the maximum number of seconds to retain a document in the cache. Use an integer in the range of 5 through 86400. The default is 900.

nocache
Specifies that documents that conform to the match pattern are not cached.

Guidelines

A Document Cache Policy that defines the following characteristics:

- Which documents the associated XML manager caches

  Cache eligibility is determined by a shell-style match pattern that specifies which documents (acquired via HTTP) are cached. Candidate documents are evaluated against the expression. Only documents with a URL that match the expression are cached.
- The priority of a document in the cache

  When the cache exceeds its maximum size, the assigned priority determines which documents to delete. Documents with the lowest priority are deleted first. If multiple documents have the same priority, the least recently-used document is deleted.
- The maximum period of time that a document is maintained in the cache

  The time-to-live (TTL) or lifetime of the document in the cache. When this time expires, the document is deleted from the cache.

Policy is set as follows:

- If the document does not contain a `Data` or `Last-Modified` header, do not cache the document.
- If the document contains a `Vary` header, do not cache the document.
- If the document contains a `Cache-Control` header (HTTP 1.1), cache the document and use the contents of that field to determine a document shelf-life.
- If the document contains an `Expires` header (HTTP 1.0), cache the document and use the contents of that field to determine a document shelf-life.
- If the document contains a `Last-Modified` header, cache the document and use the contents of that field (in conjunction with the `Data` header) to heuristically determine a document shelf-life (not to exceed 8 hours).

You can specify multiple document cache policies. Candidate documents are evaluated against each policy in order and immediately cached in the event of a match. Consequently, the order of policies is important when using the **nocache** keyword.

Use the **no policy** with a match pattern to delete specific cache policies. Use **no policy** without arguments to delete all cache policies. These commands delete the policies, not the documents in the cache.

Related Commands

clear, **size**

Examples

- Caches all XML schemas with the default priority and TTL.

```
# documentcache mgr1
Document cache configuration mode
# policy *xsd
#
```

- Caches all XML schemas with a priority of 210 and the default TTL.

```
# documentcache mgr1
Document cache configuration mode
# policy *xsd 210
#
```

- Caches all style sheets and schemas with a priority of 255 and the default TTL. Caches all XML files with the default priority and TTL. Caches all GIF files with a priority of 1 and the default TTL.

```
# documentcache mgr1
Document cache configuration mode
# policy *xs[dl] 255
# policy *xml
# policy *gif 1
#
```

- Removes the cache policy for XML schema files.

```
# no policy *xsd
#
```

- Removes all cache policies.

```
# no policy
#
```

size

Defines the size of the document cache.

Syntax

size *bytes*

Parameters

bytes Specifies the size of the document cache in bytes. The default is 0.

Guidelines

The appliance calculates a maximum cache size based on available memory. Because the default is 0, you must size the document cache to enable it.

Related Commands

maxdocs

Examples

- Enables a document cache of 1 MB.

```
# documentcache mgr1
Document cache configuration mode
# size 1048576
#
```

static-document-calls

Enables (nonstandard) independent document calls.

Syntax

static-document-calls {**on** | **off**}

Parameters

on (Default) Specifies dependent document calls.

off Specifies independent document calls.

Guidelines

XSLT specifications require that multiple document calls in the same transform return the same result. However, you can disable this behavior with the **off** keyword. When disabled, all document calls as independent of each other.

Examples

- Disables static document calls.

```
# static-document-calls off
#
```

- Restores the default behavior by enabling static document calls.

```
# static-document-calls on
#
```

Chapter 18. Document Crypto Map configuration mode

This chapter provides an alphabetic listing of commands that are available in Document Crypto Map configuration mode.

To enter this configuration mode, use the Global **document-crypto-map** command. While in this mode, design a map to enable partial (field-level) document encryption or decryption.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are available in this configuration mode.

namespace-mapping

Adds XML namespace data to the map.

Syntax

namespace-mapping *prefix URI*

Parameters

prefix Specifies the namespace prefix.

URI Specifies the location of the namespace.

Related Commands

operation, **select**

Examples

- Enters Document Crypto Map mode to create the DCM-1 Document Crypto Map. Specifies the schema for SOAP 1.1 envelope namespace.

```
# document-crypto-map DCM-1
New Document Crypto Map configuration
# namespace-mapping SOAP http://schemas.xmlsoap.org/soap/envelope/
#
```

operation

Specifies the cryptographic operation to perform.

Syntax

operation {**encrypt** | **decrypt**}

Parameters

encrypt
(Default) Specifies that selected nodes are encrypted.

decrypt
Specifies that selected nodes are decrypted.

Related Commands

namespace-mapping, **select**

Examples

- Specifies document decryption.

```
# document-crypto-map DCM-1
Modify Document Crypto Map configuration
# decrypt
#
```

---

select

Specifies the document nodes to encrypt or decrypt.

Syntax

select *XPath*

Parameters

XPath Defines an XPath expression that identifies the target nodes.

Guidelines

Document nodes that match the XPath expression are encrypted or decrypted depending on the value of the **operation** command.

Related Commands

namespace-mapping, **operation**

Examples

- Specifies that all `SSN` nodes are subject to the cryptographic operation. Because an operation is not specified, the default `encrypt` operation is assumed.

```
# document-crypto-map DCM-1
Modify Document Crypto Map configuration
# select //SSN
#
```

Chapter 19. Failure Notification configuration mode

This chapter provides an alphabetic listing of commands that are available in Failure Notification configuration mode. To enter this configuration mode, use the Global **failure-notification** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Failure Notification configuration mode.

always-on-startup

Indicates whether to send an error report on each firmware restart.

Syntax

always-on-startup {**on** | **off**}

on Sends the report.

off (Default) Does not send the report.

email-address

Provides the email address to which to send error reports.

Syntax

email-address *address*

Parameters

address Specifies the full email address of the message recipient.

Related Commands

location-id, **mode**, **remote-address**

Examples

- Specifies the email address of the recipient.

```
# email-address techsupport@TeraCorp.com
#
```

internal-state

Indicates whether to include a snapshot of the internal state.

Syntax

internal-state {**on** | **off**}

on Includes the snapshot.

off (Default) Does not include the snapshot.

location-id

Specifies the subject line of the email.

Syntax

location-id *string*

Parameters

string Specifies descriptive text.

Guidelines

The **location-id** command specifies the subject line of the email. If the message contains spaces, wrap the value in double quotation marks.

Examples

- Provides an identifying string.

```
# location-id "South Campus Building 9 5th Floor"
#
```

remote-address

Identifies the remote SMTP server to which to send the failure notification.

Syntax

remote-address *server*

Parameters

server Identifies the remote SMTP server by name or by IP address.

Examples

- Identifies the remote SMTP server.

```
# remote-address smtp.TeraCorp.com
#
```

Chapter 20. Flash configuration mode

This chapter provides an alphabetic listing of commands that are available in Flash configuration mode. To enter this configuration mode, use the Global **flash** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Flash configuration mode.

boot config

Designates the startup configuration to use for the next restart.

Syntax

boot config *filename*

Parameters

filename
Specifies the name of the startup configuration file.

Guidelines

The **boot config** and **boot image** commands work in conjunction to define the restart process, with **boot config** designating the startup configuration and **boot image** designating the appliance startup firmware image.

Note: You can also use the Global **write memory** command in conjunction with the Global **save-config overwrite** command to save the appliance running configuration to memory, and designate that saved file (autoconfig.cfg) as the startup configuration.

Related Commands

boot image, **boot update**, **save-config**, **overwrite**, **shutdown**, **write memory**

Examples

- Designates `testEnvironment.cfg` as the startup configuration.

```
# boot config testEnvironment.cfg
#
```

boot delete

Deletes the secondary install.

Syntax

boot delete

Guidelines

A firmware upgrade performed with the **boot image** command retains current configuration data, allowing the appliance to be restored to a known, stable state if necessary. The previous firmware image and associated configuration data is referred to as the *secondary* install.

While, you can use the **boot delete** command to delete the secondary install, keep in mind that its deletion will prevent firmware rollback as provided by the **boot switch** command. Consequently, deletion of the secondary install is not recommended.

Related Commands

boot image, **boot switch**

Examples

- Deletes the secondary install.

```
# boot delete
Previous firmware install deleted
#
```

---

boot image

Designates the startup firmware image.

Syntax

boot image *image*

Parameters

image Specifies the name of the firmware image to restart the appliance.

Guidelines

The **boot image** and **boot config** commands work in conjunction to define the restart process. The **boot image** command designates the appliance startup image, and the **boot config** command designates the startup configuration.

Use **boot image** to decrypt and decompress scrypt image files.

Related Commands

boot config, **boot update**, **shutdown**

Examples

- Designates `tR2207-x0503-B1.scrypt` as the startup firmware image.

```
# boot image R2207-x0503-B1.scrypt
⋮
Completed!
#
```

---

boot switch

Toggles between primary and secondary installs.

Syntax

boot switch

Guidelines

A firmware upgrade performed with the **boot image** command retains current configuration data, allowing the appliance to be restored (rolled back) to a known, stable state if necessary. The previous firmware image and associated configuration data is referred to as the *secondary* install; the newly installed firmware image and associated configuration data is referred to as the *primary* install.

You can use the **boot switch** command to toggle between the newly installed and previously active firmware versions.

Configuration edits made to a selected install are local and do not transition during rollback. It is possible to issue the **boot switch** command twice, returning the appliance to the newly installed version.

While, you can use the **boot delete** command to delete the secondary install, keep in mind that its deletion will prevent firmware rollback as provided by the **boot switch** command. Consequently, deletion of the secondary install is not recommended.

Related Commands

boot delete, **boot image**

Examples

- Rollback failed because the secondary install has been deleted with the **boot delete** command.

```
# boot switch
% Firmware roll-back failed: Switch active firmware failed:
Secondary install not available
#
```

boot update

Creates a new configuration or opens an existing configuration for editing.

Syntax

boot update write *name*

boot update append *name*

Parameters

write Creates and opens a new configuration. If the file exists, erases and opens the existing configuration.

append
Opens an existing configuration to which you can append commands.

name Specifies the name of the configuration to be created or opened.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

Guidelines

After opening the newly created or existing configuration, the command prompts for command input:

```
Enter startup commands, one per line. End with a period.
```

Enter commands, terminating each command by pressing the Return or Enter key.

If appending commands to an existing configuration, make certain to start with appropriate commands to transition to the correct configuration mode.

Follow the last command of the configuration with the following sequence to signal the end of the configuration:

```
Return
. (period)
Return
```

The CLI acknowledges configuration completion:

```
Configuration completed successfully.
```

After configuration completion you can use the **boot config** and **shutdown** commands to activate the new or edited configuration.

Related Commands

boot config, **boot image**, **shutdown**

Examples

- Creates a new configuration, `jrb_03.cfg`. If it exists, erases and opens the file.

  ```
  # boot update write jrb_03.cfg
  Enter startup commands, one per line. End with a period.
  ```

- Opens an existing configuration, `jrb_03.cfg`, to which commands will be appended.

  ```
  # boot update append jrb_01.cfg
  Enter startup commands, one per line. End with a period.
  ```

---

copy

Copies a file to or from the DataPower appliance.

Syntax

copy [**-f**] *source destination*

Parameters

-f Overwrites an existing file, if one of the same name already exists. In the absence of this argument, an attempt to save a file with the same name as an existing file will result in a prompt that requests confirmation to overwrite the existing file.

source and *destination*
Specifies the URLs that identify the source file and target destination, respectively.

- If the source file or target destination reside on the appliance, these arguments take the following form:

directory:///*filename*

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

filename
Specifies the name of a file in the specified directory.

- If the source file or target destination is remote to the DataPower appliance and the transport protocol is SCP or SFTP, these arguments take the form that is compliant with RFC 1738.

  To use an absolute path:

  scp://*user@host*:*port*//*file_path*
  sftp://*user@host*:*port*//*file_path*

  To use a path that is relative to the user's home directory:

  scp://*user@host*:*port*/*file_path*
  sftp://*user@host*:*port*/*file_path*

  Where:

  host Specifies the fully-qualified host name or IP address of the remote server. If DNS is enabled, the host name.

  port Specifies the listening port on the remote server.

  After issuing the command, the system prompts you for the remote login password.

Guidelines

The **copy** command transfers files to or from the DataPower appliance. You must issue this command from the appliance. When the source file or target destination is remote to the appliance, this command supports only the following protocols:

- HTTP
- HTTPS
- Secure Copy (SCP)
- Secured File Transfer Protocol

To send a file from the appliance as an email, use the Global **send file** command.

When using the **copy** command, be aware of the following restrictions:

- You cannot copy files from the cert: directory
- You cannot copy files to the audit:, logstore:, or logtemp: directory.

Related Commands

delete, **dir**, **move**, **send file** (Global)

Examples

- Uses HTTP to copy a file from the specified URL to the `image:` directory.

```
# copy http://host/image.crypt image:///image.crypt
file copy successful (1534897 bytes transferred)
#
```

- Uses HTTP over SSL to copy a file from the specified URL to the `image:` directory.

```
# copy https://host/image.crypt image:///image.crypt
file copy successful (1534897 bytes transferred)
#
```

- Uses SCP to copy a file from the specified URL to the `store:` directory.

```
# copy scp://jrb@10.10.1.159//XML/stylesheets/InitialConvert.xsl
store:///InitialConvert.xsl
     Password: yetanotherpassword
file copy successful
#
```

- Uses SCP to copy a file from the `logstore:` directory to the specified remote target (identified by a qualified host name).

```
# copy logstore:///Week1.log scp://jrb@ragnarok.datapower.com//LOGS/Week1.log
     Password: yetanotherpassword
file copy successful
#
```

- Uses SFTP to copy a file from the specified URL to the `store:` directory.

```
# copy sftp://jrb@10.10.1.159//XML/stylesheets/InitalConvert.xsl
 store:///InitalConvert.xsl
     Password: yetanotherpassword
file copy successful
#
```

- Uses SFTP to copy a file from the `logstore:` directory to the specified remote target.

```
# copy logstore:///Week1.log sftp://jrb@10.10.1.159//LOGS/x/Week1.log
     Password: yetanotherpassword
file copy successful
#
```

- Copies a file from the `config:` directory to the `local:` directory.

```
# copy config:///startup-config local:///startup-config
file copy successful (2347 bytes transferred)
#
```

---

delete

Deletes a file from the flash.

Syntax

delete *url*

Parameters

url

Specifies a URL of the file to delete. This argument take the following form:

directory`:///`*filename*

directory

Specifies a directory on the appliance. Refer to “Directories on the appliance” on page xxiv for details.

filename

Specifies the name of a file in the specified directory.

Guidelines

The **delete** command deletes a file on the DataPower appliance. The deletion of a file is permanent. After a file is deleted, it cannot be recovered.

Note: The **delete** command does not prompt for confirmation. Be certain that you want to delete the file before issuing this command.

Related Commands

copy, **dir**, **move**

Examples

- Deletes the `startup-config-deprecated` file from the `store:` directory.

```
# delete store:\\\startup-config-deprecated
#
```

- Deletes the `betaImage` file from the `image:` directory.

```
# delete image:\\\betaImage
#
```

---

dir

Displays the contents of a directory.

Syntax

dir *directory*

Parameters

directory

Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

Related Commands

copy, **delete**, **move**

Examples

- Displays the contents of the `config:` directory.

```
# dir config:

File Name       Last Modified              Size
-----------------------------------------------
unicenter.cfg          Mon Jul  9 11:09:36 2007   3411
autoconfig.cfg         Mon Jul  9 14:20:27 2007   20907

89.2 MB available to config:
#
```

- Displays the contents of the `msgcat` subdirectory of the `config:` directory.

```
# dir store:\\\msgcat

File Name              Last Modified              Size
---------              -------------              ----
crypto.xml             Mon Jul  9 11:09:26 2007   179069
dplane.xml             Mon Jul  9 11:09:26 2007   299644
:
xslt.xml               Mon Jul  9 11:09:26 2007   10233

89.2 MB available to store:\msgcat
#
```

move

Moves a file from one directory to another.

Syntax

move [**-f**] *source destination*

Parameters

-f Overwrites an existing file, if one of the same name already exists.

In the absence of this argument, an attempt to save a file with the same name as an existing file results in a prompt that requests confirmation to overwrite the existing file.

source and *destination*
Specifies the URLs that identify the source file and target destination, respectively. These arguments take the following form:

directory:///*filename*

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

filename
Specifies the name of a file in the specified directory.

Guidelines

You can use the **move** command to transfer a file to or from a directory. However, you cannot use the **move** command to copy a file from the private cryptographic area (such as the cert: directory).

Related Commands

copy, **delete**, **dir**

Examples

- Moves a file from the `config:` directory to the `store:` directory.

```
# move config:///startup-config store:///archiveConfig-10
#
```

- Renames a file.

```
# move config:///startup-config config:///archiveConfig-10
#
```

reinitialize

Returns the appliance to a manufactured state.

Syntax

reinitialize *filename*

Parameters

filename
Specifies the name of the firmware image to re-initialize the appliance. The file must be in the image: directory.

Guidelines

Deletes user-modified or added configuration information, including data in the DataPower directories. This data consists of style sheets, object configurations, keys, certificates, and so forth. That is, the command returns a system to the manufactured state of the specified firmware image. Refer to the **boot delete** command.

After files are deleted, they cannot be recovered. If you might need any of these files after restoring the system to a manufactured state, ensure that you have copies of these files. To recreate the appliance configuration, refer to the *IBM WebSphere DataPower SOA Appliances: 9003: Installation Guide* or to the *IBM WebSphere DataPower SOA Appliances: Type 9235: Installation Guide*, depending on your model type.

Related Commands

boot delete

Examples

- Deletes all user files and data stored on the DataPower appliance and reboots the appliance.

```
# reinitialize firmware.scrypt2
WARNING - all user data and files will be deleted
Do you want to continue ("yes" or "no")? y
#
```

---

shutdown

Restarts or shuts down the appliance.

Syntax

shutdown reboot [*seconds*]

shutdown reload [*seconds*]

shutdown halt [*seconds*]

Parameters

reboot Shuts and restarts the appliance.

reload Restarts the appliance.

halt Shuts down the appliance.

seconds
Specifies the number of seconds before the start of the shutdown operation. Use an integer in the range of 0 through 65535. The default is 10.

Guidelines

The appliance restarts using the startup configuration specified by the **boot config** command and the firmware image specified by the **boot image** command. If a startup configuration or firmware image is not designated, the appliance restarts with the configuration and firmware image that were active when you invoke the **shutdown** command.

Related Commands

boot config, boot image

Examples

- Waits 10 seconds to shut down and restart the appliance.

```
# shutdown reboot
Reboot in 10 second(s).
#
```

- Waits 20 seconds to restart the appliance.

```
# shutdown reload 20
Reload in 20 second(s).
#
```

- Waits 1 minute to shut down the appliance.

```
# shutdown halt 60
Shutdown in 60 second(s).
#
```

Chapter 21. FTP Poller Front Side Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in FTP Poller Front Side Handler configuration mode.

To enter this configuration mode, use the Global **source-ftp-poller** command. While in this configuration mode, define the client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

delay-time

Specifies the time period between polling intervals.

Syntax

delay-time *interval*

Parameters

interval
Specifies the delay between polling intervals in milliseconds. Use an integer in the range of 25 through 100000. The default is 60000.

Guidelines

The **delay-time** command specifies the number of milliseconds to wait after the completion of one poll before starting the next interval. This interval is not the polling interval. The interval is the delay between polling intervals.

error-delete

Indicates whether to delete a file after a processing failure.

Syntax

error-delete {**on** | **off**}

Parameters

on Deletes the input or processing renamed file if it could not be processed.

off (Default) Does not delete the input or processing renamed file if it could not be processed.

Guidelines

The **error-delete** command indicates whether the input or processing renamed file should be deleted when it could not be processed.

error-rename-pattern

Specifies the rename pattern when a file could not be processed.

Syntax

error-rename-pattern *pattern*

Parameters

pattern Specifies a PCRE that defines the rename pattern.

Guidelines

The **error-rename-pattern** command specifies the PCRE to rename a file when it could not be processed.

This command is relevant when **error-delete** is **off**. Otherwise, it is ignored.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

error-delete

match-pattern

Specifies the file name pattern for the search criteria.

Syntax

match-pattern *pattern*

Parameters

pattern Specifies a PCRE to use as the match pattern to search the contents of the directory.

Guidelines

The **match-pattern** command specifies the PCRE used to match the contents of the directory being polled. If there is file-renaming or there is a response, this PCRE must create PCRE back references using `()` pairs.

For example, if input files are `NNNNNN.input` the **match-pattern** would be

```
([0-9] {6})\.input$
```

PCRE documentation is available at the following web site:

http://www.pcre.org

processing-rename-pattern

Specifies the rename pattern when a file could be processed.

Syntax

processing-rename-pattern *pattern*

Parameters

pattern Specifies a PCRE that defines the rename pattern.

Guidelines

The **processing-rename-pattern** command specifies the PCRE to rename a file that is being processed. This functionality allows multiple poller objects to poll the same directory with the same match pattern. There is no lack of atomicity if the rename operation on the server is atomic. The poller that succeeds in renaming the input file will proceed to process the file. Any other poller that tries to rename the file at the same time will fail to rename the file and will proceed to try the next file that matches the specified match pattern.

To ensure uniqueness, the resulting file name will be in the following format:

filename.*serial*.*domain*.*poller*.*timestamp*

where:

filename
Specifies the file name for the renamed input file.

serial
Specifies the serial number of the DataPower appliance.

domain
Specifies the domain of the polling object.

poller
Specifies the name of the polling object.

timestamp
Specifies the timestamp.

Note: File renaming cannot be used with an FTP server that supports only 8.3 file names.

For example, if the input files are `NNNNNN.input` and you want to rename them to `NNNNNN.processing`, the **match-pattern** would be `([0-9] {6{})\.input$` and the rename pattern would be `$1.processing`. The resultant file name on the server would be:

NNNNNN.processing.*serial*.*domain* .*poller*.*timestamp*

Note: If no processing rename pattern is configured, the file will still be renamed. The only difference is that the file name portion of the resulting file is the name of the original input file, not the renamed input file.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

match-pattern

---

processing-seize-pattern

Specifies the rename pattern to search for files that are being processed.

Syntax

processing-seize-pattern *pattern*

Parameters

pattern Specifies the PCRE to use as the match pattern to search for files that are being processed.

Guidelines

The **processing-seize-pattern** command specifies the PCRE to find files that were renamed to indicate that they are in the ″being processed″ state but the processing was never completed.

The processing seize pattern contains three phrases that must be in `\( \)` pairs. The first phrase is the base file name that includes the configured processing suffix. The second phrase is the host name. The third phrase is the timestamp.

PCRE documentation is available at the following web site:

http://www.pcre.org

---

processing-seize-timeout

Specifies the time to wait before processing a file that is already in the processing state.

Syntax

processing-seize-timeout *timeout*

Parameters

timeout
Specifies the number of seconds to wait before processing a file that is already in the processing state. Use an integer in the range of 0 through 1000. The default is 0, which disable this behavior.

Guidelines

The **processing-seize-timeout** command allows failure handling of a poller when multiple data routers are polling the same target. If another data router renames a file and does not process (and rename or delete) it within the specified number of seconds, this system will try to take over processing.

This system will try to take over processing when all of the following conditions are met when compared to the processing seize pattern:

- The base file name (first match phrase) is the base file name of the processing seize pattern
- The host name (second match phrase) is not the name of this system
- The timestamp (third match phrase) is further in the past than the wait time specified by this command

When these conditions are met, this system renames the file (with its host name and a fresh timestamp) and locally processes the file. This processing assumes that the rename succeeded.

Related Commands

processing-seize-pattern

---

result

Indicates whether to create a response file after processing an input file.

Syntax

result {**on** | **off**}

Parameters

on (Default) Creates a result file.

off Does not create a result file.

Guidelines

The **result** command indicates whether the appliance should create a response file after successfully processing an input file.

---

result-name-pattern

Specifies the match pattern to build the name of the response file.

Syntax

result-name-pattern *pattern*

Parameters

pattern Specifies the PCRE to use as the match pattern to build the name of the response file.

Guidelines

The **result-name-pattern** command specifies the PCRE to use as the match pattern to build the name of the result file. This PCRE will normally have a back reference to the base input file name. For instance, in input files are `NNNNNN.input` and the desired result file name is `NNNNNN.result`, then the match pattern would be `([0-9]{6})\.input$` and the result pattern would be `$1.result`.

Some servers might allow this pattern to indicate a path that puts the file in a different director, if it allows cross-directory renames. For instance, the match pattern would be `(.*)` and the result pattern would be `../result/$1`.

This command is relevant when **result** is **on**. Otherwise, it is ignored.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

result

success-delete

Indicates whether the input file is deleted after successful processing.

Syntax

success-delete on {**on** | **off**}

Parameters

on Deletes the input file.

off (Default) Does not delete the input file.

Guidelines

The **success-delete** command indicates whether the input or processing renamed files should be deleted after successful processing.

success-rename-pattern

Specifies the rename pattern for input files on success.

Syntax

success-rename-pattern *pattern*

Parameters

pattern Specifies the PCRE to use as the match pattern to rename the input file on success.

Guidelines

The **success-rename-pattern** command specifies the PCRE to rename the input file on success. This PCRE will normally have a back reference for the base input file name. For instance, in input files are `NNNNNN.input` and the desired result file name is `NNNNNN.processed`, then the match pattern would be `([0-9]{6})\.input$` and the result pattern would be `$1.processed`.

Some servers might allow this pattern to indicate a path that puts the file in a different director, if it allows cross-directory renames. For instance, the match pattern would be `(.*)` and the result pattern would be `../processed/$1`.

This command is relevant when **success-delete** is **off**. Otherwise, it is ignored.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

success-delete

target-dir

Specifies the directory to poll.

Syntax

target-dir*directory*

Parameters

directory
Specifies the directory to poll.

Guidelines

The **target-dir** command specifies a directory to poll. The path must end in a slash. The slash denotes a directory.

For a relative path to the home directory of the specified user
`ftp://user:password@host:port/path/`

For an absolute path to the root directory
`ftp://user:password@host:port/%2Fpath/`

Do not configure one FTP poller to point at a host name that is the virtual name of a load balancer group. This configuration is not the correct way to poll multiple hosts. To poll multiple hosts, use the same Multi-Protocol Gateway and configure one FTP poller object for each real host.

xml-manager

Assigns an XML Manager.

Syntax

xml-manager*name*

Parameters

name Specifies the name of the XML Manager.

Chapter 22. FTP Quoted Commands configuration mode

This chapter provides an alphabetic listing of commands that are available in FTP Quoted Commands configuration mode.

To enter this configuration mode, use the Global **ftp-quote-command-list** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

quoted-command

Adds an entry to the list of quoted FTP commands.

Syntax

quoted-command *FTP-command*

Parameters

FTP-command
Specifies the FTP command to add to the list.

Guidelines

Adds an FTP command to the end of the list of FTP commands to be sent by the FTP User Agent to an FTP server before a file transfer. Generally the **quoted-command** command is used to send FTP **SITE** commands.

Chapter 23. FTP Server Front Side Handler mode

An instance of an FTP Server Front Side Handler object defines a handler service that receives FTP request messages from clients and forwards them to the appropriate DataPower service.

To enter the configuration mode to create a new instance or to modify an existing instance, use the Global **source-ftp-server** command. To delete an existing instance, use the Global **no source-ftp-server** command. For details about these commands, refer to"source-ftp-server" on page 101.

While in this mode, use the commands listed in Table 7 to define the handler.

- To view the current configuration, use the **show** command.
- To exit this configuration mode without saving configuration changes to the running configuration, use the **cancel** command.
- To exit this configuration mode and save configuration changes to the running configuration, use the **exit** command.

Table 7. FTP Server Front Side Handler commands

Command	Purpose
acl	Assigns an Access Control List object.
address	Specifies the local IP address for the service.
admin-state	Sets the administrative state of an object.
allow-ccc	Controls the use of the FTP **CCC** command.
allow-compression	Controls the use of the FTP **MODE Z** command.
allow-restart	Controls the use of the FTP **REST** command.
allow-unique-filename	Controls the use of the FTP **STOU** command.
certificate-aaa-policy	Assigns an AAA Policy object that determines whether to require a password for secondary authentication.
data-encryption	Controls the use of encryption of data connections (file transfers).
default-directory	Specifies the working directory on the FTP server.
filesystem	Controls the file system type that the FTP server presents.
filesystem-size	Specifies the maximum size for the temporary file system.
idle-timeout	Specifies the inactivity duration of FTP control connections.
max-filename-len	Specifies the maximum length of a file name on the FTP server.
passive	Controls the use of the FTP **PASV** command.
passive-idle-timeout	Sets the idle timeout for establishing passive connections.
passive-port-max	Sets the highest port value for the passive port range.
passive-port-min	Sets the lowest port value for the passive port range.
passive-port-range	Controls whether to limit the port range for passive connections.
persistent-filesystem-timeout	Specifies the inactivity duration for a connection to a virtual persistent file system.

Table 7. FTP Server Front Side Handler commands (continued)

Command	Purpose
password-aaa-policy	Assigns an AAA Policy to evaluate the user name and password.
port	Specifies the listening port.
require-tls	Controls whether FTP client connections require TLS encryption.
response-nfs-mount	Specifies the NFS mount in which to store response files.
response-storage	Specifies where to store response files.
response-suffix	Specifies the suffix to add when generating response files.
response-type	Specifies how to make responses available to the FTP client.
restart-timeout	Specifies the amount of time in which the client has to reconnect to the server.
ssl	Assigns an SSL Proxy Profile object.
summary	Specifies a brief, object-specific comment.
unique-filename-prefix	Specifies the prefix for the file name.
virtual-directory	Creates a directory in the virtual file system on the FTP server.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifies the name of an existing Access Control List object.

Guidelines

The **acl** command defines a reference to an existing Access Control List object. The Access Control List object allows or denies access to this service based on the IP address of the client.

When attached to a service, the default for an Access Control List is to deny all access. To deny access to only select IP addresses:

1. Create an allow clause to grant access to all IP addresses; for example, `allow 0.0.0.0`.
2. Create a deny clause to deny access for select clients; for example, `deny 10.33.194.170`

address

Specifies the local address for the service.

Syntax

local-address *address*

Parameters

address Specifies the local IP address or host alias on which the service listens. The default is `0.0.0.0`.

Guidelines

The **local-address** command specifies the local IP address on which the service listens. The default of `0.0.0.0` indicates that the service is active on all IP addresses.

The use of a host aliases can help to ease migration tasks among machines.

---

allow-ccc

Controls the use of the FTP **CCC** command.

Syntax

allow-ccc {**on** | **off**}

Parameters

on (Default) Permits the use of the **CCC** command.

off Denies the use of the **CCC** command.

Guidelines

The **allow-ccc** command controls whether the FTP **CCC** command can be used to turn off TLS encryption of the FTP control connection after user authentication. If allowed, the **CCC** command can be used to turn off encryption after authentication.

Turning off encryption is necessary when the FTP control connection crosses a firewall or NAT device that needs to sniff the control connection. Turning off encryption eliminates the secrecy about the files being transferred and allows TCP packet injection attacks.

---

allow-compression

Controls the use of FTP MODE Z compression.

Syntax

allow-compression {**on** | **off**}

Parameters

on (Default) Permits the use of the FTP compression.

off Denies the use of the FTP compression.

Guidelines

The **allow-compression** command controls whether the FTP client can use FTP **MODE Z** compression. After enabling FTP compression, the FTP client can use the `zlib` method to compress data transfers.

allow-restart

Controls the use of the **REST** command for interrupted file transfers.

Syntax

allow-restart {**on** | **off**}

Parameters

on (Default) Permits the use of the **REST** command.

off Denies the use of the **REST** command.

Guidelines

The **allow-restart** command controls whether to support the **REST** command to continue the transfer of a file after an interruption in the data transfer.

For written files, the server delays the actual processing until a timeout expires or until the next FTP command (other than **SIZE** or **REST**). The FTP client can return and resume the transfer with the **SIZE**, **REST**, and **STOR** commands. The argument to the **REST** command must be the same as the byte count returned by the **SIZE** command.

allow-unique-filename

Controls the use of the FTP **STOU** command.

Syntax

allow-unique-filename on {**on** | **off**}

Parameters

on Permits the use of the **STOU** command.

off (Default) Denies the use of the **STOU** command.

Guidelines

The **allow-unique-filename** command controls whether the FTP client can use the FTP **STOU** command. When enabled, the FTP server generates a unique file name for each transferred file.

Related Commands

unique-filename-prefix

certificate-aaa-policy

Assigns an AAA Policy that determines whether a password is required for secondary authentication.

Syntax

certificate-aaa-policy *name*

Parameters

name Specifies the name of an existing AAA Policy object.

Guidelines

The **certificate-aaa-policy** command assigns the AAA policy that determines whether a password is required for secondary authentication of the information in the TLS/SSL certificate that is provided during TLS negotiation after the **AUTH TLS** command to the FTP server. Primary authentication is done by the SSL Proxy Profile, which can completely reject a certificate. This authentication stage controls whether an FTP password will be demanded or not. If the result of this authentication succeeds, the FTP client will only have to use the **USER** command to login after the **AUTH TLS**. If this authentication fails, the FTP client will have to use both the **USER** and **PASS** commands to complete the login process. If no Certificate AAA Policy is configured, **USER** and **PASS** will always be required. If the **AUTH TLS** command is not used by the FTP client, **USER** and **PASS** will always be required.

---

data-encryption

Controls the use of encryption of data connections (file transfers).

Syntax

data-encryption {**disallow** | **allow** | **require**}

Parameters

disallow
Does not allow the FTP client to encrypt data connections.

allow (Default) Allows, but does not require, the FTP client to encrypt data connections. TLS must be allowed on data connections.

require
Requires the FTP client to encrypt data connections. TLS should be required on data connections.

Guidelines

The **data-encryption** command controls whether the FTP server allows, does not allow, or requires the FTP client to encrypt data connections (file transfers).

Data encryption is controlled by the FTP **PROT P** command.

---

default-directory

Specifies the working directory on the FTP server.

Syntax

default-directory *directory*

Parameters

directory
Specifies the initial working directory for all users on this FTP server. The default is the root directory (/).

Guidelines

The **default-directory** command specifies the current working directory for all users of this FTP server. This directory will be the initial working directory after users connect and authenticate. When using a virtual file system and the working directory is not the root directory, the specified directory must be one of the configured virtual directories.

Related Commands

virtual-directory

---

filesystem

Controls the file system type that is presented by the FTP server.

Syntax

filesystem {**virtual-ephemeral** | **virtual-persistent** | **transparent**}

Parameters

virtual-ephemeral
The FTP server will have an ephemeral virtual file system with subdirectories created by configuration. The contents of this file system are private to an individual FTP control connection to the FTP server.

The contents of this file system will not persist after this FTP control connection ends.

virtual-persistent
The FTP server will have a persistent virtual file system with subdirectories created by configuration. The contents of this file system are shared by all FTP control connections to this FTP server with the same authenticated user identity. The user identity is determined by the FTP user name and, if used, by the TLS/SSL certificate.

The contents of this file system will persist (for the duration defined by the **persistent-filesystem-timeout**) after all FTP control connections end.

This mode is required to support checkpoint/restart with the **REST** command.

transparent
The FTP server has a transparent file system. The files and directories shown are those on the FTP server.

Guidelines

The **filesystem** command controls whether the FTP server presents a virtual file system that is configured locally or transparently shows the file system on the FTP server.

If **virtual-ephemeral** or **virtual-persistent** is chosen, the client can write files to all directories. These files are shown in directory listings but cannot be retrieved. For file system responses, the responses are shown and can be retrieved, renamed, or deleted.

If **transparent** is chosen, the file system will show the contents of the equivalent path of the server on the FTP server.

Related Commands

persistent-filesystem-timeout, **virtual-directory**

filesystem-size

Specifies the maximum size for the temporary file system.

Syntax

filesystem-size *megabytes*

Parameters

megabytes

Specifies the maximum size in megabytes for the temporary file system. Use an integer in the range of 1 through 2048. The default is 32.

Guidelines

The **filesystem-size** command specifies the maximum size in megabytes for the temporary file system. This command is relevant when the file system type, as defined by the **filesystem** command is **virtual-ephemeral** or **virtual-persistent** and the response storage, as defined by the **response-storage** command is **temporary**.

Related Commands

filesystem, **response-storage**

idle-timeout

Specifies the inactivity duration of FTP control connections.

Syntax

idle-timeout *seconds*

Parameters

seconds

Specifies the number of seconds that the FTP control connection can be idle. The default is 0, which disables the timeout.

Guidelines

The **idle-timeout** command specifies the number of seconds that the FTP control connection can be idle before it times out. After the specified duration elapses, the FTP server closes the control connection.

max-filename-len

Specifies the maximum file name length on the FTP server.

Syntax

max-filename-len *length*

Parameters

length Specifies the maximum length of a file name on the FTP server. Use an integer in the range of 1 through 4000. The default is 256.

---

passive

Controls the use of passive mode by the FTP client.

Syntax

passive {**disallow** | **allow** | **require**}

Parameters

disallow
Does not allow the FTP client to use the **PASV** command to open data connections to the FTP server. The FTP server will open all data connections to the FTP client (as directed by the FTP **PORT** command). Often, this mode is incompatible with firewall and proxy servers.

allow (Default) Allows, but does not require, the FTP client to use the **PASV** command to open data connections to the FTP server.

require
Requires the FTP client to use the **PASV** command to open all data connections to the FTP server. This mode is useful when the FTP server is behind a firewall device that requires clients to use passive mode.

Guidelines

The **passive** command controls whether the FTP server allows, does not allow, or requires passive mode to be used by the FTP client.

When requiring passive mode, the FTP client must use the FTP **PASV** command. Without the use of the **PASV** command, the FTP **STOR**, **STOU**, and **RETR** commands will fail when issued by the FTP client.

When not allowing passive mode, the FTP client must use the FTP **PORT** command.

---

passive-idle-timeout

Sets the idle timeout for establishing passive connections.

Syntax

passive-idle-timeout *seconds*

Parameters

seconds
Specifies the number of seconds that the server waits for a client to establish a passive connection. Use an integer in the range of 5 through 300. The default is 60.

Guidelines

The **passive-idle-timeout** command controls the amount of time in seconds between when the FTP server issues code 227 ("Entering Passive Mode") in response to the **PASV** or **EPSV** command from the FTP client and when the FTP client must establish a TCP data connection to the listening port and issue a data transfer command.

- If a data connection is not established within the timeout period, the listening port will be closed. If a data transfer command is issued after the port is closed, the command fails with code 425 and the "Failed to open data connection" message.
- If a data connection is established but no data transfer command is issued within the timeout period, the TCP data connection will be closed. Any data transfer command after the timeout will be treated as if the **PASV** or **EPSV** command was never issued. The command fails with code 425 and the "require PASV or PORT command first" message.

This command is relevant when allowing or requiring the use of passive mode and when limiting port usage to a specific range. In other words, this command is relevant when both of the following conditions are met:

- The value of the **passive** command is any keyword except **disallow**.
- The value of the **passive-port-range** command is the keyword **on**.

Related Commands

passive, **passive-port-range**

---

passive-port-max

Sets the highest port value for the passive port range.

Syntax

passive-port-max *port*

Parameters

port Specify the higher end of the passive port range. Use an integer in the range of 1024 through 65534. The default is 1050.

Guidelines

The **passive-port-max** command sets the highest port value for the passive port range. This value must be greater than the value set by the **passive-port-min** command.

This command is relevant when allowing or requiring the use of passive mode and when limiting port usage to a specific range. In other words, this command is relevant when both of the following conditions are met:

- The value of the **passive** command is any keyword except **disallow**.
- The value of the **passive-port-range** command is the keyword **on**.

Related Commands

passive, **passive-port-min**, **passive-port-range**

passive-port-min

Sets the lowest port value for the passive port range.

Syntax

passive-port-min *port*

Parameters

port Specify the lower end of the passive port range. Use an integer in the range of 1024 through 65534. The default is 1024.

Guidelines

The **passive-port-min** command sets the lowest port value for the passive port range. This value must be less than the value set by the **passive-port-max** command.

This command is relevant when allowing or requiring the use of passive mode and when limiting port usage to a specific range. In other words, this command is relevant when both of the following conditions are met:

- The value of the **passive** command is any keyword except **disallow**.
- The value of the **passive-port-range** command is the keyword **on**.

Related Commands

passive, **passive-port-max**, **passive-port-range**

passive-port-range

Controls whether to limit the port range for passive connections.

Syntax

passive-port-range {**on** | **off**}

Parameters

on Enables the use of a limited port range.

off (Default) Disables the use of a limited port range.

Guidelines

The **passive-port-range** controls whether to use a limited, TCP port range for passive connections. This command is useful when a firewall or proxy server want to allow incoming FTP data connections for a limited port range only. This behavior is common with the FTP server that cannot use a packet analyzer (sometimes known as a *packet sniffer*) on the control connection.

The range limits the maximum number of FTP clients that can be in the state between when the FTP server issues a 227 ("Entering Passive Mode") in response to the **PASV** or **EPSV** command from the FTP client and when the FTP client must establish a TCP data connection to the listening port and issue a data transfer command. To control the pressure on this limited resource, use the **passive-idle-timeout** command to adjust the idle timeout value for passive data connections.

Note: While multiple FTP servers on the same system can use the same or overlapping passive port ranges, this configuration could introduce contention for a common resource in the TCP implementation.

Because of contention, do not use a port range that overlaps with other services that are on the same system as the FTP server. Because there is no configuration check for such conflicts, another service might allocate the ports first. If this happens, the port will not be available for the FTP server when it attempts to allocate a listening port dynamically.

When limiting the port range, use the following commands to define the range and the idle timeout for data connections:

- **passive-idle-timeout**
- **passive-port-max**
- **passive-port-min**

This command is relevant when allowing or requiring the use of passive mode. In other words, this command is relevant when the value of the **passive** command is any keyword except **disallow**.

Related Commands

passive, **passive-idle-timeout**, **passive-port-max**, **passive-port-min**

persistent-filesystem-timeout

Specifies the inactivity duration for a connection to a virtual persistent file system.

Syntax

persistent-filesystem-timeout *seconds*

Parameters

seconds

Specifies the inactivity duration in seconds. Use an integer in the range of 1 through 43000. The default is 600.

Guidelines

The **persistent-filesystem-timeout** command specifies the duration in seconds that a connection to a virtual file system is retained after all FTP control connections from user identities are disconnected. When the timeout expires, the virtual file system object is destroyed. All of the response files that were not deleted by the FTP client are deleted from their storage area.

This command is relevant when **filesystem** is **virtual-persistent**. Otherwise, it is ignored. Otherwise, it is ignored.

Related Commands

filesystem

password-aaa-policy

Assigns an AAA Policy to evaluate the user name and password.

Syntax

password-aaa-policy *name*

Parameters

name Specifies the name of an existing AAA Policy object.

Guidelines

The **password-aaa-policy** command assigns the AAA policy to perform authentication of user names and passwords provided to the FTP server by the client with the **USER** and **PASS** commands.

- If authentication succeeds, the FTP client can use all of the features of the FTP server.
- If authentication fails, a 530 error is returned. The user can attempt to authenticate again.

Without an AAA password policy, any user name and password is accepted but evaluation does not occur until there is a data transfer operation. Therefore, until a full AAA policy succeeds, an FTP client cannot perform any of the following FTP operations:

- Read files with the **RETR** command
- Write files with the **STOU** command
- Delete files with the **DELE** command
- Take a directory with the **NLST** command or with the **LIST** command.

---

port

Specifies the listening port.

Syntax

port *port*

Parameters

port Specifies the TCP listening port for the service. The default is 21.

Guidelines

The **port** command specifies the port that is monitored by the FTP service. This port is the port on which FTP control connections can be established. This port does not control the TCP port that is used for the data connections. If the FTP client uses the FTP **PASV** command, data connections will use an arbitrary, unused TCP port.

---

require-tls

Controls whether FTP control connections require TLS encryption.

Syntax

require-tls {**on** | **off**}

Parameters

on Requires TLS encryption.

off (Default) Does not require TLS encryption.

Guidelines

The **require-tls** command controls whether FTP control connections require TLS encryption. If required, the FTP client must use the FTP **AUTH TLS** command before any other command.

To support TLS encryption, ensure that the configuration of the associated instance of the User Agent object defines the relevant information to contact the FTP server. Use the **ftp-policy** command in User Agent configuration mode to define this configuration.

This setting does not control encryption of data transfers.

Related Commands

ftp-policy (User Agent)

---

response-nfs-mount

Specifies the NFS mount in which to store response files.

Syntax

response-nfs-mount *name*

Parameters

name Specifies the name of an existing NFS static mount.

Guidelines

The **response-nfs-mount** command specifies the NFS static mount in which to store response files. Each response file will have a unique file name in the NFS directory. The name of the response file is not related to the file name that the virtual file system presents to the FTP client.

Generally, this NFS directory is not made available through the FTP server. This directory should not be used for any other purpose.

This command is relevant when the **response-storage** is **nfs**. Otherwise, it is ignored.

Related Commands

response-storage

---

response-storage

Specifies where response files are stored.

Syntax

response-storage {**temporary** | **nfs**}

Parameters

temporary
(Default) Stores response files in temporary storage on the system. This storage space has limited size.

nfs Stores response files on the top level directory of the specified NFS server. Only the NFS server limits the storage space.

Guidelines

The **response-storage** command specifies the storage for response file. In normal operation, the FTP client should delete response files after reading them. After a system restart or power cycle, response files do not remain available.

The default is **temporary**, where the response files are stored in temporary storage on the system. This storage is limited in size. Using temporary storage reduces the available system storage for processing. Temporary storage should not be used when using a virtual persistent file system. In virtual persistent file systems, it is highly unlikely that there would be enough space.

Selecting **nfs** uses an NFS server to store these files, which eliminates storage space constraints.

This command is relevant when **response-type** is **virtual-filesystem**. Otherwise, it is ignored.

Related Commands

filesystem, **response-nfs-mount**, **response-type**

---

response-suffix

Specifies the suffix to add when generating response files.

Syntax

response-suffix *suffix*

Parameters

suffix Specifies the suffix to add. Use a regular expression in the ^[^/]*$ form. Note that the directory separator (/) is not allowed.

The default is an empty string.

Guidelines

The **response-suffix** command specifies the suffix to add to the URL file name portion before using it as the response file name or URL. If none specified, no suffix is added.

This command is relevant when **response-type** is **virtual-filesystem** or **ftp-client**. Otherwise, it is ignored.

Related Commands

response-type

response-type

Selects how to make a response available for gateway transactions started by an FTP **STOR** or **SOUT** operation.

Syntax

response-type {**none** | **virtual-filesystem** | **ftp-client**}

Parameters

none (Default) Indicates that no response is made available to the client. Any response from the server is dropped.

virtual-filesystem
Indicates that the response is made available as a file in the virtual file system that can be read by the FTP client.

ftp-client
Indicates that the response is written by the FTP client.

Guidelines

The **response-type** command selects how to make a response available for gateway transactions started by an FTP **STOR** or **SOUT** operation.

If **virtual-filesystem** is chosen, the response is made available as a file in the virtual file system that can be read by the FTP client. The directory for responses is configured on a per-virtual directory basis. The suffix specified by the **response-suffix** command is added to the input file name. The **virtual-directory** command controls where the files are stored.

If **ftp-client** is chosen, the response is not made available using the FTP server. The response is written using the FTP client. The response is written to the URL specified by the **response-url** command. The file name consists of the incoming file name plus the suffix specified by the **response-suffix** command.

This command is relevant when **filesystem** is **virtual-ephemeral** or **virtual-persistent**. Otherwise, it is ignored.

Related Commands

filesystem, **response-suffix**, **response-url**, **virtual-directory**

response-url

Selects the generated response URL.

Syntax

response-url *URL*

Parameters

URL Specifies the URL that is used for the generated response. The default is to have no response generated, which is an empty string. Use a regular expression in the `^(|ftp://[^/]+(/[^/]+)*)$` form.

Guidelines

The **response-url** command selects the URL that is used in generating a response. This URL enables a response to be written using FTP commands. The URL must be an FTP URL that starts with `ftp://`. The URL should include a directory, but not a file name. The URL cannot include query parameters.

This command is relevant when **response-type** is **ftp-client**. Otherwise, it is ignored.

Related Commands

response-type

---

restart-timeout

Specifies the amount of time the client has to reconnect to the server.

Syntax

restart-timeout *seconds*

Parameters

seconds
Specifies the number of seconds that the client has to reconnect to the server.

Guidelines

The **restart-timeout** command specifies the number of seconds that the FTP client has to reconnect to the server. The FTP client needs to use **SIZE**, **REST**, and **STOR** commands to continue an interrupted file transfer. If this period of time elapses, the data that was received to this point on the TCP data connection will be passed to the DataPower service. This timeout is canceled if another command (other than **SIZE** or **REST**) is received on the FTP control connection.

---

ssl

Assigns an SSL Proxy Profile object.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ssl** command indicates the existing SSL proxy profile to assign to the object.

---

unique-filename-prefix

Defines a file name prefix.

Syntax

unique-filename-prefix *variable*

Parameters

variable

Defines the prefix for file names that are generated when using the FTP **STOU** command. When defining the prefix, the directory separator (/) is not allowed. The default is to not add a prefix, which is an empty string. Use a regular expression in the ^[^/]*$ form.

Guidelines

The **unique-filename-prefix** define the prefix for file names that are generated when using the FTP **STOU** command. When defining the prefix, the directory separator (/) is not allowed. A numeric suffix is generated.

This command is relevant when **allow-unique-filename** is **on**. Otherwise, it is ignored.

Related Commands

allow-unique-filename

---

virtual-directory

Creates a directory in the virtual file system on the FTP server.

Syntax

virtual-directory *path directory*

Parameters

path Specifies the directory in the virtual file system of the FTP server where the FTP client can find this directory. Use a regular expression in the ^/[^/]+(/[^/]+)*$ form.

directory

Specifies the directory in the virtual file system of the FTP server where the responses to files that are stored in this directory will go. Use a regular expression in the ^/[^/]+(/[^/]+)*$ form.

Guidelines

The **virtual-directory** command creates a directory in the virtual file system that is presented by the FTP server. The FTP client can use all of these directories to write to be processed.

The root directory (/) is always present and cannot be created, and its response directory is always the root directory.

This command is relevant when the file system type is **virtual-ephemeral** or **virtual-persistent**. Otherwise, it is ignored.

Related Commands

filesystem

Chapter 24. Hard Disk Array configuration mode (Type 9235)

This chapter provides an alphabetic listing of commands that are available in Hard Disk Array configuration mode. To enter this configuration mode, use the Global **raid-volume** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Hard Disk Array configuration mode.

directory

Specifies the name of the directory.

Syntax

directory *name*

Parameters

name Specifies the name of the subdirectory.

Guidelines

The **directory** command specifies the directory under which to make the files on the hard disk array available in the local: and logstore: directories in each application domain.

Examples

- Makes the files on the hard disk array accessible in the `local:///disk` and `logstore:///disk` directories.

```
# raid-volume raid0
Hard Disk Array configuration mode
# directory disk
#
```

read-only

Sets the files on the hard disk array to read-only access.

Syntax

read-only

no read-only

Guidelines

The **read-only** command sets the files on the hard disk array to read-only access. The default is read-write.

Examples

- Makes the file system read-only.

```
# raid-volume raid0
Hard Disk Array configuration mode
# read-only
#
```

- Makes the file system read-write, the default state.

```
# raid-volume raid0
Hard Disk Array configuration mode
# no read-only
#
```

Chapter 25. Host Alias configuration mode

This chapter provides an alphabetic listing of commands available in Host Alias configuration mode.

To enter this configuration mode, use the Global **host-alias** command.

Many of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Host Alias configuration mode.

ip-address

Creates an alias for an IP address on a Ethernet port.

Syntax

ip-address *address*

Parameters

address Specifies the IP address to map.

Guidelines

The **ip-address** commands creates an alias for a local IP address of the DataPower appliance. Instead of providing the IP address, you can specify this alias.

Examples

- Creates the `Ragnarok` alias. Maps `Ragnarok` to IP address 192.168.12.12.

```
# host-alias Ragnarok
New Host Alias configuration
# ip-address 192.168.12.12
# exit
#
```

Chapter 26. HTTP Front Side Handler mode

An instance of an HTTP Front Side Handler object defines a handler service that receives HTTP request messages from clients and forwards them to the appropriate DataPower service.

To enter the configuration mode to create a new instance or to modify an existing instance, use the Global **source-http** command. To delete an existing instance, use the Global **no source-http** command. For details about these commands, refer to "source-http" on page 102.

While in this mode, use the commands listed in Table 8 to define the handler.

- To view the current configuration, use the **show** command.
- To exit this configuration mode without saving configuration changes to the running configuration, use the **cancel** command.
- To exit this configuration mode and save configuration changes to the running configuration, use the **exit** command.

Table 8. HTTP Front Side Handler commands

Command	Purpose
acl	Assigns an Access Control List object.
admin-state	Sets the administrative state of an object.
allowed-features	Specifies the methods and versions to allow in incoming HTTP requests.
compression	Controls the negotiation of GZIP compression.
local-address	Specifies the local IP address for the service.
http-client-version	Sets the HTTP version for the connection.
max-header-count	Specifies the maximum number of headers to allow.
max-header-name-len	Specifies the maximum length of header names to allow.
max-header-value-len	Specifies the maximum length of header values to allow.
max-querystring-len	Specifies the maximum length of the query string to allow.
max-total-header-len	Specifies the maximum aggregate length of HTTP headers to allow.
max-url-len	Specifies the maximum length of URLs to allow.
persistent-connections	Controls the negotiation of persistent connections.
port	Specifies the listening port.
summary	Specifies a brief, object-specific comment.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifies the name of an existing Access Control List object.

Guidelines

The **acl** command defines a reference to an existing Access Control List object. The Access Control List object allows or denies access to this service based on the IP address of the client.

When attached to a service, the default for an Access Control List is to deny all access. To deny access to only select IP addresses:

1. Create an allow clause to grant access to all IP addresses; for example, `allow 0.0.0.0`.
2. Create a deny clause to deny access for select clients; for example, `deny 10.33.194.170`

---

allowed-features

Specifies the methods and versions to allow in incoming HTTP requests.

Syntax

allowed-features *feature*[+*feature*]...

Parameters

feature[+*feature*]...
Specifies a list of features to allow in requests. Concatenate features with the plus sign (+). The following feature tokens are available:

- `CmdExe`
- `DELETE`
- `DotDot`
- `FragmentIdentifiers`
- `GET`
- `HEAD`
- `HTTP-1.0`
- `HTTP-1.1`
- `OPTIONS`
- `POST`
- `PUT`
- `QueryString`
- `TRACE`

Guidelines

The **allowed-features** command specifies the methods and versions to allow in incoming HTTP requests. This handler will accept HTTP requests that contains any of the allowed features. If the feature is not in the list, the handler rejects the request. By default, this handler permits the following features:

- `FragmentIdentifiers`
- `HTTP-1.0`
- `HTTP-1.1`
- `POST`
- `PUT`
- `QueryString`

Examples

- Limits features to `HTTP-1.0`, `HTTP-1.1`, `POST`, and `QueryString`.

```
# allowed-features HTTP-1.0+HTTP-1.1+POST+QueryString
#
```

compression

Controls the negotiation of GZIP compression.

Syntax

compression {**on** | **off**}

Parameters

on Enables compression negotiation.

off (Default) Disables compression negotiation.

Guidelines

The **compression** command controls whether to enable or to disable GZIP compression negotiation.

local-address

Specifies the local address for the service.

Syntax

local-address *address*

Parameters

address Specifies the local IP address or host alias on which the service listens. The default is `0.0.0.0`.

Guidelines

The **local-address** command specifies the local IP address on which the service listens. The default of `0.0.0.0` indicates that the service is active on all IP addresses.

The use of a host aliases can help to ease migration tasks among machines.

http-client-version

Sets the HTTP version for the connection.

Syntax

http-client-version {**http/1.0** | **http/1.1**}

Parameters

http/1.0
Uses HTTP 1.0.

http/1.1
(Default) Uses HTTP 1.1.

Guidelines

The **http-client-version** command set the HTTP version for the connection. The specified version should not conflict with the HTTP version that is allowed by the **allowed-features** command.

Related Commands

allowed-features

max-header-count

Specifies the maximum number of headers to allow.

Syntax

max-header-count *count*

Parameters

count Specifies the maximum number of headers. The default is 0, which indicates no limit.

Guidelines

The **max-header-count** command specifies the maximum number of HTTP header to allow in incoming request messages.

Examples

- Limits the number HTTP headers to 20.

```
# max-header-count 20
#
```

max-header-name-len

Specifies the maximum length of header names to allow.

Syntax

max-header-name-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. The default is 0, which indicates no limit.

Guidelines

The **max-header-name-len** command specifies the maximum length of header names to allow for HTTP headers in request messages. Each HTTP header is expressed as a name-value pair. This command specifies the maximum length of the name portion for HTTP headers. The use **max-header-value-len** command to specify the maximum length of the value portion for HTTP headers.

Related Commands

max-header-value-len

max-header-value-len

Specifies the maximum length of header values to allow.

Syntax

max-header-value-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. The default is 0, which indicates no limit.

Guidelines

The **max-header-value-len** command specifies the maximum length of header values to allow for HTTP headers in request messages. Each HTTP header is expressed as a name-value pair. This command specifies the maximum length of the value portion for HTTP headers. The use **max-header-name-len** command to specify the maximum length of the name portion for HTTP headers.

Related Commands

max-header-name-len

max-querystring-len

Specifies the maximum length of the query string to allow.

Syntax

max-querystring-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. The default is 0, which indicates no limit.

Guidelines

The **max-querystring-len** command specifies the maximum length of the query string to allow for HTTP headers in request messages. The query string is the portion of the URL after the question mark (?) character.

Examples

- Limits the query string to 1024 bytes.

```
# max-querystring-len 1024
#
```

max-total-header-len

Specifies the maximum aggregate length of HTTP headers to allow.

Syntax

max-total-header-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. Use an integer in the range of 5 through 128000. The default is 128000.

Guidelines

The **max-total-header-len** command specifies the maximum aggregate length of incoming HTTP headers to allow in request messages.

Examples

- Limits aggregated HTTP headers to 65535 bytes.

```
# max-total-header-len 65535
#
```

max-url-len

Specifies the maximum length of URLs to allow.

Syntax

max-url-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. Use an integer in the range of 1 through 128000. The default is 16384.

Guidelines

The **max-url-len** command specifies the maximum length of the URL to allow in request messages. The URL includes the query string and fragment identifiers.

Examples

- Limits the URL to 32000 bytes.

```
# max-url-len 32000
#
```

persistent-connections

Controls the negotiation of persistent connections.

Syntax

persistent-connections {**on** | **off**}

Parameters

on (Default) Enables the establishment of persistent connections.

off Disables the establishment of persistent connections.

Guidelines

The **persistent-connections** command controls the negotiation of persistent connections.

- When enabled, the handler negotiates with the remote peer and establishes a persistent connection if agreeable to the peer.
- When disabled, the handler does not attempt to negotiate the establishment of persistent connections.

---

port

Specifies the TCP listening port.

Syntax

port *port*

Parameters

port Specifies the TCP listening port for the service. The default is 80.

Guidelines

The **port** command specifies the port that is monitored by the DataPower service.

Chapter 27. HTTP Input Conversion Map configuration mode

This chapter provides an alphabetic listing of commands that are available in HTTP Input Conversion Map configuration mode.

To enter this configuration mode, use the Global **input-conversion-map** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in HTTP Input Conversion Map configuration mode.

default-encoding

Specifies the input default encoding.

Syntax

default-encoding {**base64** | **plain** | **urlencoded** | **xml**}

Parameters

base64
Treats input literally. Adds `encoding='base64'` to input element.

plain XML escapes the input.

urlencoded
(Default) URL unescapes, then XML escapes the input.

xml Treats input literally, no processing.

Related Commands

rule

Guidelines

The input conversion map describes the way an incoming document is expected to be encoded. For each input in a request, the name is compared to the list of rules. Each rule contains a regular expression and the resulting encoding. The first matching regular expression indicates the encoding to be used for the input. If no rules match, the encoding specified by the default-encoding property is used.

Examples

- Defines an HTTP Input Conversion Map: default encoding is `urlencoded`. Any input that ends with `xml` is treated as XML. Any input that ends with `base64` is treated and tagged as Base64.

```
# input-conversion-map ICM-1
New HTTP Input Conversion Map configuration
# default-encoding urlencoded
# rule xml$ xml
# rule base64$ base64
```

rule

Adds a processing rule to the current HTTP conversion map.

Syntax

rule *expression* {**base64** | **plain** | **urlencoded** | **xml**}

Parameters

expression
Defines a PCRE regular expression that defines an input element.

base64
Treats input literally. Adds `encoding='base64'` to input element.

plain XML escapes the input.

urlencoded
(Default) URL unescapes, then XML escapes the input.

xml Treats input literally, no processing.

Related Commands

default-encoding

Guidelines

The input conversion map describes the way an incoming document is expected to be encoded. For each input in a request, the name is compared to the list of rules. Each rule contains a regular expression and the resulting encoding. The first matching regular expression indicates the encoding to be used for the input. If no rules match, the encoding specified by the default-encoding property is used.

PCRE documentation is available at the following web site:

http://www.pcre.org

Examples

- Defines an HTTP conversion map: default encoding is `urlencoded`. Any input that ends with `xml` is treated as XML. Any input that ends with `base64` is treated and tagged as Base64.

```
# input-conversion-map ICM-1
New HTTP Input Conversion Map configuration
# default-encoding urlencoded
# rule xml$ xml
# rule base64$ base64
```

Chapter 28. HTTP Service configuration mode

This chapter provides an alphabetic listing of commands that are available in HTTP Service configuration mode.

To enter this configuration mode, use the Global **httpserv** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in HTTP Service configuration mode.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifics the name of an ACL.

Guidelines

Assignment of an ACL to a HTTP service is optional. If an ACL is assigned to the service, only those IP addresses specifically allowed by the ACL can initiate access to the HTTP Service on the appliance; if an ACL is not assigned, access to the HTTP service on the appliance is unrestricted.

Related Commands

acl (Global), **allow** (ACL), **deny** (ACL)

identifier

Specifies the contents of the `Server` response header field.

Syntax

identifier *string*

no identifier

Parameters

string Specifies the contents of the `Server` response header field. Use double quotes to bracket strings that contain spaces.

Guidelines

The `Server` response header field generally contains information (name and version) that describes the server application software. By default, inclusion of the `Server` response header field is suppressed.

Note: Users should consider security implications before revealing software version information.

Use the **no identifier** command to suppress the `Server` response header field.

Examples

- Specifies `Release 3.7.1` as the contents of the `Server` response header field.

```
# identifier "Release 3.7.1"
#
```

- Suppresses the transmission of the `Server` response header field.

```
# no identifier
#
```

ip-address

Specifies the local IP address to monitor for incoming traffic.

Syntax

ip-address {*address* | **0**}

Parameters

address Specifies the IP address (primary or secondary) of a DataPower Ethernet interface.

0 Indicates all DataPower Ethernet interfaces.

Guidelines

In conjunction with the **port** command, identifies the IP addresses and ports that the HTTP service monitors.

Related Commands

port

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current HTTP service monitor.

```
# http telnet-1
Telnet Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

local-directory

Specifies the directory from which the HTTP service serves documents.

Syntax

local-directory {**config:** | **image:** | **store:** | **temporary:**}

Parameters

config:
Serves documents from the configuration (config:) directory

image: Serves documents from the firmware image (image:) directory

store: (Default) Serves documents from the general storage (store:) directory

temporary:
Serves documents from the temporary (temporary:) directory

Examples

- Specifies that the current HTTP service serves documents from the `temporary:` directory.

```
# local-directory temporary:
#
```

---

mode

Specifies the operational mode of the current HTTP service.

Syntax

mode {**normal** | **echo**}

Parameters

echo Places the HTTP service in loopback mode.

normal
Returns the requested document.

Examples

- Creates the `echoServer` loopback HTTP service that monitors port 8888 on all active Ethernet interfaces.

  Creates two XSL Proxy services (`Proxy-1` and `Proxy-2`) with a remote address of 127.0.0.1 (standard IP loopback address) and port 8888 (the `echoServer` loopback HTTP service).

```
# httpserv echoServer
New HTTP Service configuration
# mode echo
# ip-address 0
# port 8888
# exit
#

# xslproxy Proxy-1
New XSL Proxy configuration
# ip-address 0
# port 9222
# remote-address 127.0.0.1 8888
# xml-manager mgr1
# stylesheet-policy Expedite
# exit
#

# xslproxy Proxy-2
New XSL Proxy configuration
# ip-address 0
# port 9223
# remote-address 127.0.0.1 8888
# xml-manager mgr1
# stylesheet-policy Process
# exit
#
```

port

Specifies the local port monitored by the HTTP service for incoming traffic.

Syntax

port *port*

Parameters

port Specifies the port. The default is 80.

Guidelines

Use the **port** command to change the port that is assigned with the **ip-address** command.

Related Commands

ip-address

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current HTTP service monitor.

```
# http telnet-1
Telnet Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal
(Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

start-page

Specifies the file to load when a client first accesses the HTTP service.

Syntax

start-page *filename*

Parameters

filename
Specifies the name of the file to display.

Guidelines

In the absence of this command, the HTTP service displays the directory listing that is specified by the **local-directory** command.

Related Commands

local-directory

Examples

- Specifies `Welcome.html` as the start page.

```
# start-page Welcome.html
#
```

Chapter 29. HTTPS Front Side Handler mode

An instance of an HTTPS Front Side Handler object defines a handler service that receives HTTP request messages from clients and forwards them to the appropriate DataPower service.

To enter the configuration mode to create a new instance or to modify an existing instance, use the Global **source-https** command. To delete an existing instance, use the Global **no source-https** command. For details about these commands, refer to "source-https" on page 102.

While in this mode, use the commands listed in Table 9 to define the handler.

- To view the current configuration, use the **show** command.
- To exit this configuration mode without saving configuration changes to the running configuration, use the **cancel** command.
- To exit this configuration mode and save configuration changes to the running configuration, use the **exit** command.

Table 9. HTTPS Front Side Handler commands

Command	Purpose
acl	Assigns an Access Control List object.
admin-state	Sets the administrative state of an object.
allowed-features	Specifies the methods and versions to allow in incoming HTTP requests.
compression	Controls the negotiation of GZIP compression.
local-address	Specifies the local IP address for the service.
http-client-version	Sets the HTTP version for the connection.
max-header-count	Specifies the maximum number of headers to allow.
max-header-name-len	Specifies the maximum length of header names to allow.
max-header-value-len	Specifies the maximum length of header values to allow.
max-querystring-len	Specifies the maximum length of the query string to allow.
max-total-header-len	Specifies the maximum aggregate length of HTTP headers to allow.
max-url-len	Specifies the maximum length of URLs to allow.
persistent-connections	Controls the negotiation of persistent connections.
port	Specifies the listening port.
summary	Specifies a brief, object-specific comment.
ssl	Assigns an SSL Proxy Profile object.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifies the name of an existing Access Control List object.

Guidelines

The **acl** command defines a reference to an existing Access Control List object. The Access Control List object allows or denies access to this service based on the IP address of the client.

When attached to a service, the default for an Access Control List is to deny all access. To deny access to only select IP addresses:

1. Create an allow clause to grant access to all IP addresses; for example, `allow 0.0.0.0`.
2. Create a deny clause to deny access for select clients; for example, `deny 10.33.194.170`

---

allowed-features

Specifies the methods and versions to allow in incoming HTTP requests.

Syntax

allowed-features *feature*[+*feature*]...

Parameters

feature[+*feature*]...
Specifies a list of features to allow in requests. Concatenate features with the plus sign (+). The following feature tokens are available:

- `CmdExe`
- `DELETE`
- `DotDot`
- `FragmentIdentifiers`
- `GET`
- `HEAD`
- `HTTP-1.0`
- `HTTP-1.1`
- `OPTIONS`
- `POST`
- `PUT`
- `QueryString`
- `TRACE`

Guidelines

The **allowed-features** command specifies the methods and versions to allow in incoming HTTP requests. This handler will accept HTTP requests that contains any of the allowed features. If the feature is not in the list, the handler rejects the request. By default, this handler permits the following features:

- `FragmentIdentifiers`
- `HTTP-1.0`
- `HTTP-1.1`
- `POST`
- `PUT`
- `QueryString`

Examples

- Limits features to `HTTP-1.0`, `HTTP-1.1`, `POST`, and `QueryString`.

```
# allowed-features HTTP-1.0+HTTP-1.1+POST+QueryString
#
```

compression

Controls the negotiation of GZIP compression.

Syntax

compression {**on** | **off**}

Parameters

on Enables compression negotiation.

off (Default) Disables compression negotiation.

Guidelines

The **compression** command controls whether to enable or to disable GZIP compression negotiation.

local-address

Specifies the local address for the service.

Syntax

local-address *address*

Parameters

address Specifies the local IP address or host alias on which the service listens. The default is `0.0.0.0`.

Guidelines

The **local-address** command specifies the local IP address on which the service listens. The default of `0.0.0.0` indicates that the service is active on all IP addresses.

The use of a host aliases can help to ease migration tasks among machines.

http-client-version

Sets the HTTP version for the connection.

Syntax

http-client-version {**http/1.0** | **http/1.1**}

Parameters

http/1.0
Uses HTTP 1.0.

http/1.1
(Default) Uses HTTP 1.1.

Guidelines

The **http-client-version** command set the HTTP version for the connection. The specified version should not conflict with the HTTP version that is allowed by the **allowed-features** command.

Related Commands

allowed-features

max-header-count

Specifies the maximum number of headers to allow.

Syntax

max-header-count *count*

Parameters

count Specifies the maximum number of headers. The default is 0, which indicates no limit.

Guidelines

The **max-header-count** command specifies the maximum number of HTTP header to allow in incoming request messages.

Examples

- Limits the number HTTP headers to 20.

```
# max-header-count 20
#
```

max-header-name-len

Specifies the maximum length of header names to allow.

Syntax

max-header-name-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. The default is 0, which indicates no limit.

Guidelines

The **max-header-name-len** command specifies the maximum length of header names to allow for HTTP headers in request messages. Each HTTP header is expressed as a name-value pair. This command specifies the maximum length of the name portion for HTTP headers. The use **max-header-value-len** command to specify the maximum length of the value portion for HTTP headers.

Related Commands

max-header-value-len

max-header-value-len

Specifies the maximum length of header values to allow.

Syntax

max-header-value-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. The default is 0, which indicates no limit.

Guidelines

The **max-header-value-len** command specifies the maximum length of header values to allow for HTTP headers in request messages. Each HTTP header is expressed as a name-value pair. This command specifies the maximum length of the value portion for HTTP headers. The use **max-header-name-len** command to specify the maximum length of the name portion for HTTP headers.

Related Commands

max-header-name-len

max-querystring-len

Specifies the maximum length of the query string to allow.

Syntax

max-querystring-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. The default is 0, which indicates no limit.

Guidelines

The **max-querystring-len** command specifies the maximum length of the query string to allow for HTTP headers in request messages. The query string is the portion of the URL after the question mark (?) character.

Examples

- Limits the query string to 1024 bytes.

```
# max-querystring-len 1024
#
```

max-total-header-len

Specifies the maximum aggregate length of HTTP headers to allow.

Syntax

max-total-header-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. Use an integer in the range of 5 through 128000. The default is 128000.

Guidelines

The **max-total-header-len** command specifies the maximum aggregate length of incoming HTTP headers to allow in request messages.

Examples

- Limits aggregated HTTP headers to 65535 bytes.

```
# max-total-header-len 65535
#
```

max-url-len

Specifies the maximum length of URLs to allow.

Syntax

max-url-len *bytes*

Parameters

bytes Specifies the maximum length in bytes. Use an integer in the range of 1 through 128000. The default is 16384.

Guidelines

The **max-url-len** command specifies the maximum length of the URL to allow in request messages. The URL includes the query string and fragment identifiers.

Examples

- Limits the URL to 32000 bytes.

```
# max-url-len 32000
#
```

persistent-connections

Controls the negotiation of persistent connections.

Syntax

persistent-connections {**on** | **off**}

Parameters

on (Default) Enables the establishment of persistent connections.

off Disables the establishment of persistent connections.

Guidelines

The **persistent-connections** command controls the negotiation of persistent connections.

- When enabled, the handler negotiates with the remote peer and establishes a persistent connection if agreeable to the peer.
- When disabled, the handler does not attempt to negotiate the establishment of persistent connections.

port

Specifies the TCP listening port.

Syntax

port *port*

Parameters

port Specifies the TCP listening port for the service. The default is 80.

Guidelines

The **port** command specifies the port that is monitored by the DataPower service.

ssl

Assigns an SSL Proxy Profile object.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ssl** command indicates the existing SSL proxy profile to assign to the object.

Chapter 30. Import Configuration File configuration mode

This chapter provides an alphabetic listing of commands that are available in Import Configuration File configuration mode.

To enter this configuration mode, use the Global **import-package** command. While in this mode, identify the location and type of an configuration file to import to the running configuration.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

auto-execute

Indicates whether to import the configuration package at startup.

Syntax

auto-execute {**on** | **off**}

Parameters

on (Default) Imports the configuration package at startup. The configuration is marked `external` and cannot be saved to the startup configuration. This behavior is equivalent to always importing the configuration.

off Imports the configuration package when manually triggered. The configuration is not marked `external` and can be saved to the startup configuration. This behavior is equivalent to importing the configuration one time.

Guidelines

The **auto-execute** indicates whether to import the configuration package at startup or to import the configuration only when manually triggered. When configuration data are marked `external`, they cannot be saved to the startup configuration.

If the configuration package defines objects that are part of the startup configuration and the value for the **overwrite-objects** command is **on**, the object in the configuration package overwrites the object in the startup configuration. In this case, a warning is written to the log.

If the configuration package contains files that are on the system and the value for the **overwrite-files** command is **on**, files in the configuration package overwrite files on the system. In this case, a warning is written to the log.

Related Commands

overwrite-files, **overwrite-objects**

Examples

- Disables automatic importation at startup.

```
# import-package Englewood
New Import Configuration File configuration
# auto-execute off
#
```

deployment-policy

Specifies the name of an existing deployment policy that preprocesses the configuration package.

Syntax

deployment-policy *name*

Parameters

name Specifies the name of an existing Deployment Policy object.

Related Commands

deployment-policy

Guidelines

The **deployment-policy** command specifies the name of an existing Deployment Policy object that preprocesses the configuration package. To create a new Deployment Policy object, use the Global **deployment-policy** command.

Examples

- Enter Import Configuration File configuration mode to create the `Norwood` configuration package and applies the existing `Policy-3` Deployment policy on the package being imported.

```
# import-package Norwood
New Import Configuration File configuration
deployment-policy Policy-3
source-url http://10.10.10.10
#
```

import-format

Specifies the file format of the configuration package.

Syntax

import-format {**xml** | **zip**}

Parameters

xml Indicates that the format as an XML tree.

zip (Default) Indicates that the format as a ZIP file.

Examples

- Enter Import Configuration File configuration mode to create the `Goodies` configuration package. Identifies the format as XML.

```
# import-package Goodies
New Import Configuration File configuration
# import-format xml
#
```

local-ip-rewrite

Indicates whether to rewrite local IP addresses.

Syntax

local-ip-rewrite {**on** | **off**}

Parameters

on (Default) Rewrites IP addresses to match the local configuration when imported.

off Retains the original IP address in the configuration package.

Guidelines

The **local-ip-rewrite** command indicates whether to rewrite local IP addresses on import. When rewriting IP addresses, a service in the configuration package that binds to `eth1` is rewritten to bind to `eth1` when imported.

overwrite-files

Indicates whether to overwrite files when the configuration package contains the same file.

Syntax

overwrite-files {**on** | **off**}

Parameters

on (Default) Overwrites files of the same name.

off Does not import the file if a file of the same name exists.

Guidelines

The **overwrite-files** command indicates whether to overwrite files when the configuration package contains the same file. If files in the configuration package overwrite files on the system, a warning is written to the log.

Related Commands

destination-domain, **overwrite-objects**

Examples

- Specifies that existing files in the destination domain are not overwritten by imported files.

```
# overwrite-files off
#
```

overwrite-objects

Indicates whether to overwrite objects when the configuration package contains the same object.

Syntax

overwrite-objects {**on** | **off**}

Parameters

on (Default) Overwrites objects of the same name.

off Does not import the objects if an objects of the same name exists.

Guidelines

The **overwrite-objects** command indicates whether to objects when the configuration package contains the same object. If objects in the configuration package overwrite objects on the system, a warning is written to the log.

Related Commands

destination-domain, **overwrite-files**

Examples

- Specifies that existing objects in the destination domain are not overwritten by imported objects.

```
# overwrite-objects off
#
```

---

source-url

Specifies the location of the Import Package.

Syntax

source-url *URL*

Parameters

URL Specifies the location of the Import Package.

Guidelines

The import tool does not support SCP and SFTP URL protocols. All other URL protocols are available; for example, HTTP, HTTPS, or FTP.

Examples

- Identifies the location configuration package.

```
# source-url http://192.168.32:8088/Imports/Goodies
#
```

Chapter 31. IMS Connect configuration mode

This chapter provides an alphabetic listing of commands that are available in IMS Connect configuration mode.

To enter this configuration mode, use the Global **ims** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

client-id-prefix

Sets the two-letter prefix for the generated client ID.

Syntax

client-id-prefix *prefix*

Parameters

prefix Specifies the two-letter prefix for the generated client ID. If not specified, the prefix is set to `DP`.

clientid

Specifies the name of the IMS client identifier.

Syntax

clientid *identifier*

Parameters

Specifies the name of the IMS client identifier. If not supplied by the client, the user exit must generate it.

datastore

Specifies the name of the data store.

Syntax

datastore *name*

Parameters

name Specifies the name of the data store.

Guidelines

Specifies the name of the data store (IMS destination ID). This property must be specified by the client. The IMS Connect returns the data store name from the exit in the `OMUSR_DESTID` OTMA header field. This value can be overridden by specifying it in the backend URL.

ebcdic-conversion

Indicates whether to enable EBCDIC header conversion.

Syntax

ebcdic-conversion {**on** | **off**}

Parameters

on Converts IMS headers to EBCDIC.

off (Default) Does not convert IMS headers to EBCDIC.

Guidelines

Indicates whether to enable EBCDIC header conversion. Conversion affects only the header, not the payload. To convert the payload, use a transformation in a processing policy. The user message exit should be able to process EBCDIC data. Some use message exits can handle both UTF-8 and EBCDIC.

encoding-scheme

Sets the Unicode encoding scheme.

Syntax

encoding-scheme {**NotSet** | **Default** | **UCS2** | **UTF8** | **UTF16**}

Parameters

Default
Uses the encoding that is set by the message.

NotSet
(Default) Uses the encoding that is set by the IMS Connect Handler object or by a transform action in the processing policy.

UCS2 Uses UCS-2 encoding.

UTF8 Uses UTF-8 encoding.

UTF16 Uses UTF-16 encoding.

exit-program

Specifies the exit program.

Syntax

exit-program *program*

Parameters

program
Specifies the exit program to use for all IMS connections.

group

Identifies the RACF® security group.

Syntax

group *group*

Parameters

group Specifies the group to which the security ID belongs.

hostname

Specifies the host of the target IMS Connect.

Syntax

hostname *host*

Parameters

host Specifies the host name or IP address of the machine running the target IMS Connect.

irm-timer

Sets the wait time to return data.

Syntax

irm-timer *timeout*

Parameters

timeout
Specifies an appropriate wait time for IMS server to return data to IMS Connect. This value sets the `IRM_TIMER`. Refer to the IMS Connect documentation for details. For example, a value of 21 sets the value to 0.21 seconds.

lterm-name

Sets the logical terminal name.

Syntax

lterm-name *name*

Parameters

name Specifies the logical terminal (LTERM) name.

Guidelines

For IMS host applications, the value for this field is set by the user message exit. The user exit message either moves this value to the `OMHDRLTM` OTMA field or sets `OMHDRLTM` with a predetermined value. If you specify an override value, OTMA places this value in the `IOPCB` field. If you do not specify an override value, OTMA places the IMS Connect-defined `TPIPE` name in the `IOPCB` field.

The `TPIPE` name is set to one of the following values based on the commit mode:

- If the commit mode is `0`, sets the value to the client identifier (`CLIENT ID`).

- If the commit mode is 1, sets the value to the port identifier (PORT ID).

If you use the LTERM value in the IOPCB to make logic decisions, be aware of the naming conventions of the IOPCB LTERM name.

password

Sets the connection password.

Syntax

password *password*

Parameters

password
Specifies the RACF password to log in to IMS Connect.

port

Specifies the port on the target IMS Connect.

Syntax

port *port*

Parameters

port Specifies the port that the target IMS Connect uses for TCP/IP connections.

tran-code

Specifies the transaction code to invoke.

Syntax

tran-code *code*

Parameters

code Specifies the transaction code.

Guidelines

Specifies the transaction code to invoke. This value can be overridden by specifying it in the backend URL.

username

Sets the RACF identifier.

Syntax

username *identifier*

Parameters

identifier
Specifies the RACF identifier.

Guidelines

Specifies the plaintext string that is sent to the server to identify the client.

Chapter 32. IMS Connect Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in IMS Connect Handler configuration mode.

To enter this configuration mode, use the Global **source-imsconnect** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifies the name of an existing ACL.

Guidelines

Only those IP addresses that are explicitly granted access by the assigned ACL can access the IMS Connect Handler.

ebcdic-input

Sets the encoding for input headers as EBCDIC or ASCII.

Syntax

ebcdic-input {**on** | **off**}

Parameters

on Indicates the input headers are in EBCDIC encoding.

off (Default) Indicates that input headers are in ASCII encoding.

Guidelines

The **ebcdic-input** command sets the encoding for input headers as EBCDIC or ASCII. This setting does not affect payload processing. Payload is not automatically processed.

local-address

Specifies the appliance interface.

Syntax

local-address {*address* | **0**}

Parameters

address Binds to a single interface on the specified port. Specify the interface by IP address or by host alias.

0 Binds to all enabled interfaces on the specified port.

Related Commands

port

persistent-connections

Controls the negotiation of persistent connections.

Syntax

persistent-connections {**on** | **off**}

Parameters

on (Default) Enables the establishment of persistent connections.

off Disables the establishment of persistent connections.

Guidelines

When enabled, the IMS Connect Handler negotiates with the remote target to establish a persistent connection, if agreeable with the target.

port

Specifies the appliance TCP port.

Syntax

port *port*

Parameters

port Specify the port that the IMS listener monitors. The default is 3000.

Related Commands

local-address

ssl

Assigns an SSL Proxy Profile

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

An SSL Proxy Profile must be assigned to the IMS Connect Handler to use secure communication. The SSL Proxy Profile must exist in the current application domain. To create an SSL Proxy Profile, use the Global **sslproxy** command.

Related Commands

sslproxy (Global)

Chapter 33. Include Configuration File configuration mode

This chapter provides an alphabetic list of commands in Include Configuration File configuration mode.

To enter this configuration mode, use the Global **include-config** command. While in this mode, specify the location of configuration files to appended to other configuration files.

All of the commands in "Common commands" on page 2 and most, but not all, of the commands in Chapter 129, "Monitoring commands," on page 1053 are available in this configuration mode.

auto-execute

Specifies whether the included-configuration is executed at appliance startup.

Syntax

auto-execute {**on** | **off**}

Parameters

on (Default) Imports the configuration package at startup. The configuration is marked `external` and cannot be saved to the startup configuration. This behavior is equivalent to always importing the configuration.

off Imports the configuration package when manually triggered. The configuration is not marked `external` and can be saved to the startup configuration. This behavior is equivalent to importing the configuration one time.

Guidelines

The **auto-execute** indicates whether to import the configuration package at startup or to import the configuration only when manually triggered. When configuration data are marked `external`, they cannot be saved to the startup configuration.

Examples

- Disables automatic execution at appliance startup.

```
# include-config StdSvcProxy
New Include Configuration File configuration
# auto-execute off
#
```

config-url

Specifies the location of a configuration file to include in another configuration file.

Syntax

config-url *URL*

Parameters

URL Identifies the location of the configuration file to include.

- If the file resides on the appliance, this parameter takes the form *directory*:///*filename*, where:

  directory
  Identifies a local directory. Generally, the directory is one of the following keywords:
  - config
  - local
  - temporary

  filename
  Identifies the file in the directory.

- If the file is remote to the appliance and the transport protocol is HTTP, HTTPS, SCP, or SFTP, this parameter takes one of the following forms:
  - http://*user*:*password*@*host*/*file*
  - https://*user*:*password*@*host*/*file*
  - scp://*user*:*password*@*host*/*file*
  - sftp://*user*:*password*@*host*/*file*

  The host name can be specified as an IP address or as a qualified host name when DNS services were previously enabled.

Guidelines

The **config-url** command specifies the location of a configuration file to include in another configuration file.

Examples

- Specifies the location of a remote configuration file to include.

```
# include-config StdSvcProxy
New Include Configuration File configuration
# config-url scp://jrb:passWoRd@baldar.ibm.com/configs/Proxy1.cfg
```

- Specifies the location of a local configuration file to include.

```
# include-config StdSvcProxy
Modify Include Configuration File configuration
# config-url local:///Proxy2.cfg
```

interface-detection

Specifies when to retrieve the Include Configuration File.

Syntax

interface-detection {**on** | **off**}

Parameters

on Retrieves the file after the local interface is up (synchronous).

off (Default) Retrieves the file at system reload without waiting for the local interface to be up (asynchronous).

Guidelines

The **interface-detection** command determine when to retrieve the Include Configuration File in relationship to the state of the local interface. This command is meaningful only when **auto-execute** is **on**.

Related Commands

auto-execute

Examples

- Specifies synchronous execution of the Include Configuration File.

```
# include-config StdSvcProxy
New Include Configuration File configuration
# interface-detection on
#
```

Chapter 34. Interface configuration mode

This chapter provides an alphabetic listing of commands that are available in Interface configuration mode.

To enter this configuration mode, use the Global **interface** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Interface configuration mode.

arp

Enables the Address Resolution Protocol (ARP) on all IP interfaces provided by the current Ethernet port.

Syntax

arp

no arp

Guidelines

ARP is enabled by default on all IP interfaces. Certain network topologies and load-balancing configurations might require that you disable ARP.

If required, use the **no arp** command to disable ARP on all IP interfaces supported by the current Ethernet port. Use the `arp` command to restore default ARP processing.

Related Commands

interface, **show netarp**

Examples

- Disables ARP on the current interface.

```
# no arp
#
```

- Enables ARP on the current interface, restoring the default state.

```
# arp
#
```

dhcp

Enables a DHCP (Dynamic Host Configuration Protocol) client on the current interface.

Syntax

dhcp

no dhcp

Guidelines

You can use DHCP to obtain the following parameters from a DHCP server:
- Interface IP address
- Default Gateway IP address
- DNS IP address

Use the **no dhcp** command to disable the DHCP client.

Examples

- Enables a DHCP client on Ethernet 2.

```
# interface eth2
# dhcp
# exit
#
```

- Disables the DHCP client on Ethernet 2.

```
# interface eth2
# no dhcp
# exit
#
```

---

ip address

Assigns an IP address and subnet mask to an Ethernet port, and enables the interface.

Syntax

ip address *address/netmask* [**secondary**]

no ip address*address/netmask* [**secondary**]

no ip address secondary

Parameters

address Specifies an IP address to be assigned to the current Ethernet port.

netmask
Identifies the network portion of the interface address. Can be expressed in CIDR (slash) format, which is an integer that specifies the length of the network portion of the address, or in dotted decimal format.

secondary
Optionally identifies the address as a secondary address.

Guidelines

After assigning an initial IP address (referred to as the *primary* address), you can use the **secondary** keyword to assign additional addresses (referred to as secondary addresses) to the same Ethernet port. An Ethernet port can have a single primary address and multiple secondary addresses

Use the **no ip address secondary** to remove all secondary addresses from an interface.

Related Commands

interface, **show ip address**, **standby**

Examples

- Assigns a primary IP address to Ethernet port 0.

```
# ip address 192.168.7.6/27
#
```

- Functionally equivalent to the previous example.

```
# ip address 192.168.7.6 255.255.224.0
#
```

- Assigns a secondary IP address to Ethernet port 0.

```
# ip address 192.168.7.7/27 secondary
#
```

- Removes the primary IP address from Ethernet port 0.

```
# no ip address 192.168.7.7/27
#
```

- Removes the specified secondary IP address from Ethernet port 0.

```
# no ip address 192.168.7.7/27 secondary
#
```

- Removes all secondary addresses from Ethernet port 0.

```
# no ip address secondary
#
```

ip default-gateway

Designates the system default gateway reachable from the current interface.

Syntax

ip default-gateway *address*

no ip default-gateway *address*

Parameters

address Specifies the IP address of the default gateway.

Guidelines

Use the **no ip default-gateway** command, to delete the default gateway.

Related Commands

show ip default-gateway

Examples

- Identifies the system default gateway reachable from Ethernet interface 0.

```
# ip default-gateway 10.10.10.100
#
```

- Deletes the system default gateway.

```
# no ip default-gateway 10.10.10.100
#
```

ip route

Adds a static route to the routing table.

Syntax

ip route *address*/*netmask next-hop-address* [*metric*]

no ip route *address*/*netmask next-hop-address*

Parameters

address Specifies the address of the destination network.

netmask
Identifies the network portion of the address. Can be expressed in CIDR (slash) format, which is an integer that specifies the length of the network portion of the address, or in dotted decimal format.

next-hop-address
Specifies the IP address of the next-hop router.

metric Assigns a routing metric to this static route. The greater the metric value, the more preferred the route. Use a value in the range of 0 to 255. The default is 0.

Guidelines

Use the **no ip route** command to delete a static route.

Examples

- Adds a static route (subnet 10.10.10.224 reached via next-hop router 192.168.1.100) to the routing table.

```
# ip route 10.10.10.0/27 192.168.1.100
#
```

- Deletes the same static route (subnet 10.10.10.224 reached via next-hop router 192.168.1.100) from the routing table.

```
# no ip route 10.10.10.0/27 192.168.1.100
#
```

mac-address

Specifies the MAC address of the current interface.

Syntax

mac-address *MAC-address*

Parameters

MAC-address
Specifies the 48-bit MAC address in hexadecimal format.

Guidelines

By default, the operating software uses the “burned-in” MAC addresses from the network interface media.

Examples

- Sets a nondefault MAC address for Ethernet interface 0.

```
# mac-address 00:11:22:aa:bb:cc
#
```

mode

Specifies the operational mode (speed and duplex) for the current Ethernet interface.

Syntax

mode *mode*

Parameters

mode Specifies the Ethernet mode using one of the following keywords:

10baseT-FD or **10baseT-HD**
Indicates standard Ethernet configuration options.

100baseTx-FD or **100baseTx-HD**
Indicates Fast Ethernet configuration options.

1000baseTxFD
Indicates Gigabit Ethernet configuration options.

Auto (Default) Indicates auto-negotiated speed and duplex.

Guidelines

Specifies the operational mode (speed and duplex) for the current Ethernet interface. On Type 9235 appliances, you cannot change the operational mode associated with `eth1` and `eth2`.

Related Commands

interface, **show interface mode**

Examples

- Modifies the Ethernet 0 interface by placing it in `100baseTx` full duplex mode.

```
# interface eth0
Interface configuration mode (eth0)
(config-if[eth0]) # mode 100baseTx-FD
Interface operational parameters set (100baseTx-FD)
(config-if[eth0])#
#
```

- Modifies the Ethernet 0 interface by placing it in auto-negotiation mode (the default mode).

```
# interface eth0
Interface configuration mode (eth0)
(config-if[eth0]) # mode Auto
Interface operational parameters set (Auto)
(config-if[eth0])#
#
```

mtu

Sets an interface-specific maximum transmission unit (MTU).

Syntax

mtu *size*

Parameters

size Specifies the maximum size of an MTU.

Specifies the MTU for the current interface in bytes. Use an integer in the range of 576 to 16128. The default is 1500.

Guidelines

The MTU is determined without regard to the length of the layer 2 encapsulation.

Examples

- Sets the MTU for the current interface to 4 kilobytes.

```
# mtu 4096
#
```

---

packet-capture

Starts or stops a packet-capture session.

Syntax

Starts a package capture
packet-capture *filename duration kilobytes*

Stops a package capture
no packet-capture *filename*

Parameters

filename
Specifies the file to which packet-capture data is written.

duration
Specifies the maximum duration of the packet-capture session in seconds. Use a value in the range of 5 through 3600. The special value of **-**1 indicates that the packet capture completes when the maximum file size is reached or until you invoke the **no packet-capture**.

kilobytes
Specifies the maximum size, in kilobytes, of the packet-capture file in kilobytes. Use a value in the range of 10 through 50000.

Guidelines

Packet-capture date is saved in a *pcap* format. Use a utility such as **tcpdump** or **ethereal** to interpret the packet-capture file.

Use the **no packet-capture** command to immediately end a packet-capture session, regardless of the duration specified by the duration parameter.

Examples

- Initiates a packet-capture session on Ethernet 0. Packet-capture data is written to the file `Eth0Trace` in the general storage directory. The session terminates after 30 seconds or when `Eth0Trace` contains 2500 kilobytes of data (whichever occurs first).

```
# packet-capture store://Eth0Trace 1800 2500
Trace begun.
⋮
#
```

- Initiates and then terminates a packet-capture session.

```
# packet-capture store://Eth0Trace 1800 2500
Trace begun.
⋮
# no packet-capture store://Eth0Trace
#
```

standby

Implements a failover configuration

Syntax

To assign both interfaces to a group using a Virtual IP address (VIP)
standby *group-number* **ip** *address*

To assign a priority to a standby member of a group
standby *group-number* **priority** *priority-value*

To place the active interface in preempt mode
standby *group-number* **preempt**

To disable preemption on a standby group
no standby *group-number* **preempt**

To delete a standby group
no standby *group-number*

To delete all standby groups on the current interface
no standby

Parameters

group-number
Specifies the number of the standby group. Use a value in the range of 1 through 255.

ip *address*
Specifies the virtual IP address (VIP) to assign to the standby group. Both interfaces must use the same VIP.

priority *priority-value*
Specifies the priority of the interface. Use a value in the range of 0 through 255. The default is 100.

For the active interface, specify a value of 100 or greater. For the standby interface, specify a value less than 100.

preempt
Indicates preemption. When in preempt mode, the designated active interface returns to service after a failure, it resumes its active role and the standby interface resumes its standby role. Otherwise, the designated active interface functions in standby capacity after it returns to service.

Guidelines

The **standby** command implements a failover configuration to ensure that an interface on another DataPower appliance is available if an active interface becomes unresponsive.

There are two types of failover configurations:

- An *active* interface is backed up by a *warm standby* interface. This configuration is known as an `active-standby` topology.

  In this topology, the warm standby is inactive on the network except that it monitors the active interface. If the warm standby detects that the active interface has become unresponsive, it assumes the IP address of the formerly active interface and enables all its own network services. In this scenario, assured failure protection is provided, but at a cost of one idle standby interface for each active one.
- An active interface backs up another active interface. This configuration is known as an `active-active` topology.

  In an `active-active` topology, two standby groups are configured. An interface is assigned the active role on one group and the standby role on the second. The active and standby roles are reversed for the second interface. In this scenario, assured failure protection is provided, without the cost of idle standby interfaces.

  Note: In an `active-active` topology, the standby groups must be configured on different Ethernet ports.

To implement a failover configuration, use the **standby** command to configure both active and standby interfaces.

1. Assign both interfaces to the same group.
2. Assign a priority to the standby interface.
3. Optionally place the active interface in preempt mode.

The redundant topology ensures that, in the event an interface becomes unresponsive (perhaps as a result of internal hardware failure or intermittent network failure), a standby interface takes the place of the non-functioning interface.

Use the **no standby** command to disable a failover configuration or to disable preemption.

Related Commands

interface, **ip address**

Examples

- Assigns Ethernet 0 to standby group 2 and specifies a VIP of 10.10.66.66. Not specifying a priority (accepting the default of 100) ensures that the interface is the active member of the group. Places the interface in preempt mode meaning that it resumes the active role following a failure and subsequent restoration to service.

```
# interface ethernet 0
Interface configuration mode (ethernet 0)
# ip address 10.10.66.1 255.0.0.0
# ip default-gateway 10.20.1.1
```

```
# standby 2 ip 10.10.66.66
# standby 2 preempt
# exit
#
```

- Assigns Ethernet 0 to standby group 2 and specifies a VIP of 10.10.66.66. The priority value of 90 ensures that the interface is the standby member of the group. Because it is the standby member, it is not placed in preempt mode.

  ```
  # interface ethernet 0
  Interface configuration mode (ethernet 0)
  # 10.10.66.3 255.0.0.0
  # ip default-gateway 10.30.1.1
  # standby 2 ip 10.10.66.66
  # standby 2 priority 90
  # exit
  #
  ```

- Assigns Ethernet 0 to standby group 5 in the active role and specifies a VIP of 10.10.66.66. Not specifying a priority (accepting the default of 100) ensures that the interface is the active member of this group. Places the interface in preempt mode meaning that it resumes the active role following a failure and subsequent restoration to service.

  Assigns Ethernet 1 to standby group 7 in the standby role and specifies a VIP of 10.10.66.67. The priority value of 90 ensures that the interface is the standby member of the group. Because it is the standby member, it is not placed in preempt mode.

  ```
  # interface ethernet 0
  Interface configuration mode (ethernet 0)
  # ip address 10.10.66.1 255.0.0.0
  # ip default-gateway 10.20.1.1
  # standby 5 ip 10.10.66.66
  # standby 2 preempt
  # exit
  # interface ethernet 1
  Interface configuration mode (ethernet 1)
  # ip address 10.10.66.2 255.0.0.0
  # standby 7 ip 10.10.66.67
  # standby 7 priority 90
  # exit
  #
  ```

- Disables preempt mode for standby group 2 on Ethernet 0.

  ```
  # interface ethernet 0
  Interface configuration mode (ethernet 0)
  # no standby 2 preempt
  # exit
  #
  ```

- Deletes standby group 2 on Ethernet 0.

  ```
  # interface ethernet 0
  Interface configuration mode (ethernet 0)
  # no standby 2
  # exit
  #
  ```

- Deletes all standby groups on Ethernet 0.

  ```
  # interface ethernet 0
  Interface configuration mode (ethernet 0)
  # no standby
  # exit
  #
  ```

Chapter 35. iSCSI CHAP configuration mode (Type 9235)

This chapter provides an alphabetic listing of commands that are available in iSCSI CHAP configuration mode. The CHAP is the challenge handshake authentication protocol.

To enter this configuration mode, use the Global **iscsi-chap** command. While in this mode, define the credential to use to access the iSCSI volume.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in iSCSI CHAP configuration mode.

password

Specifies the password for the CHAP user.

Syntax

password *password*

Parameters

password
Specifies the password for the CHAP user.

Guidelines

The **password** command specifies the password for the CHAP user.

Examples

- Sets `Gerry` as the user with the password `BigSecret` as the credentials for the `CHAP-2` CHAP.

```
# iscsi-chap CHAP-2
New iSCSI CHAP configuration mode
# username Gerry
# password BigSecret
#
```

username

Specifies the user for the CHAP.

Syntax

username *user*

Parameters

user Specifies a user name.

Guidelines

The **username** command specifies the user for the CHAP.

Examples

- Sets `Gerry` as the user with the password `BigSecret` as the credentials for the `CHAP-2` CHAP.

```
# iscsi-chap CHAP-2
New iSCSI CHAP configuration mode
# username Gerry
# password BigSecret
#
```

Chapter 36. iSCSI Host Bus Adapter configuration mode (Type 9235)

This chapter provides an alphabetic listing of commands that are available in iSCSI Host Bus Adapter configuration mode. To enter this configuration mode, use the Global **iscsi-hba** command.

All Type 9235 appliances have two Host Bus Adapters. Each adapter has a "burned in" iSCSI qualified name (IQN). The only required configuration is to define the network connectivity. Use either DHCP or specify the IP address and default gateway.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in iSCSI Host Bus Adapter configuration mode.

dhcp

Indicates whether to use DHCP.

Syntax

dhcp {**on** | **off**}

Parameters

on Enables DHCP.

off (Default) Disables DHCP.

Guidelines

The **dhcp** command specifies whether to use DHCP.

- When enabled, values for the **ip-address** and **ip default-gateway** commands are ignored.
- When disabled, define the network connection for the HBA with the **ip-address** and **ip default-gateway** commands.

Related Commands

ip-address, **ip default-gateway**

Examples

- Enables DHCP for the `iscsi-1` HBA.

```
# iscsi-hba iscsi-1
Modify iSCSI Host Bus Adapter configuration
# dhcp on
#
```

- Disables DHCP for the `iscsi-1` HBA.

```
# iscsi-hba iscsi-1
Modify iSCSI Host Bus Adapter configuration
# dhcp off
#
```

iname

Changes the iSCSI qualified name.

Syntax

iname *IQN*

Parameters

IQN Specifies the IQN.

Guidelines

The **iname** command changes the "burned in" value for the iSCSI qualified name (IQN). If you need to change this value, specify an IQN in the following format:

- `iqn.2001-04.com.example`
- `iqn.2001-04.com.example:storage:diskarrays-sn-a8675309`
- `iqn.2001-04.com.example:storage.tape1.sys1.xyz`
- `iqn.2001-04.com.example:storage.disk2.sys1.xyz`

Examples

- Makes the iSCSI qualified name `iqn.2001-04.com.example:storage.disk6.Balboa` for `iscsi-2` HBA .

```
# iscsi-hba iscsi-2
Modify iSCSI Host Bus Adapter configuration
# iname iqn.2001-04.com.example:storage.disk6.Balboa
#
```

ip-address

Specifies the IP address for the HBA.

Syntax

ip-address *address*

Parameters

address Specifies the IP address.

Guidelines

The **ip-address** command specifies the IP address for the HBA. Use this command with the **ip default-gateway** command to define the network connection.

Do not use the **ip-address** command or the **ip default-gateway** command when **dhcp** is **on**.

Related Commands

dhcp, **ip default-gateway**

Examples

- Sets `10.10.10.44` as the IP address and `10.10.10.46` as the default gateway for the `iscsi-2` HBA .

```
# iscsi-hba iscsi-2
Modify iSCSI Host Bus Adapter configuration
# ip-address 10.10.10.44
# ip default-gateway 10.10.10.46
#
```

ip default-gateway

Specifies the default gateway for the HBA.

Syntax

ip default-gateway *address*

Parameters

address Specifies the IP address of the default gateway.

Guidelines

The **ip default-gateway** command specifies the IP address of the default gateway for the HBA. Use this command with the **ip-address** command to define the network connection.

Do not use the **ip-address** command or the **ip default-gateway** command when **dhcp** is **on**.

Related Commands

dhcp, **ip-address**

Examples

- Sets `10.10.10.44` as the IP address and `10.10.10.46` as the default gateway for the `iscsi-2` HBA .

```
# iscsi-hba iscsi-2
Modify iSCSI Host Bus Adapter configuration
# ip-address 10.10.10.44
# ip default-gateway 10.10.10.46
#
```

Chapter 37. iSCSI Target configuration mode (Type 9235)

This chapter provides an alphabetic listing of commands that are available in iSCSI Target configuration mode.

To enter this configuration mode, use the Global **iscsi-target** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in iSCSI Target configuration mode.

chap

Assigns an iSCSI CHAP.

Syntax

chap *name*

Parameters

name Specifies the name of the existing iSCSI CHAP.

Guidelines

The **chap** command specifies the name of the existing iSCSI CHAP.

Examples

- Assigns the `CHAP-1` CHAP to the `Target-1` iSCSI target.

```
# iscsi-target Target-1
NEW iSCSI Target configuration mode
# chap CHAP-1
#
```

hba

Assigns an iSCSI HBA.

Syntax

hba *name*

hba{**iscsi1** | **iscsi2**}

Parameters

iscsi1 Specifies the existing iSCSI HBA keyword that identifies the **eth1** Ethernet interface.

iscsi2 Specifies the existing iSCSI HBA keyword that identifies the **eth2** Ethernet interface.

Guidelines

The **hba** command assigns an existing iSCSI HBA to which to bind this target instance.

Examples

- Assigns the `iscsi1` HBA to the `Target-2` iSCSI target.

```
# iscsi-target Target-2
New iSCSI Target configuration mode
# hba iscsi1
#
```

---

hostname

Specifies the host of the iSCSI target.

Syntax

hostname *host*

Parameters

host Specifies the host name or IP address of the remote iSCSI server.

Guidelines

The **hostname** command identifies the remote iSCSI target by host name or IP address.

Examples

- Identifies the`Target-3` iSCSI target with the `10.10.10.33` IP address.

```
# iscsi-target Target-3
New iSCSI Target configuration mode
# hostname 10.10.10.33
#
```

---

port

Specifies the listening port.

Syntax

port *port*

Parameters

port Specifies the listening port. The default is 3260.

Guidelines

The **port** command specifies the listening port on the remote iSCSI target.

Examples

- Defines port 2222 as the listening port on the `Target-4` iSCSI target.

```
# iscsi-target Target-4
New iSCSI Target configuration mode
# port 2222
#
```

target-name

Specifies a name of the remote iSCSI target.

Syntax

target-name *name*

Parameters

name Specifies the iSCSI qualified name (IQN) or IEEE Extended Unique Identifier (EUI) for the iSCSI target.

Guidelines

The **target-name** specifies the iSCSI qualified name (IQN) or IEEE Extended Unique Identifier (EUI) for the iSCSI target.

- To specify an IQN, use the following format:
  - `iqn.2001-04.com.example`
  - `iqn.2001-04.com.example:storage:diskarrays-sn-a8675309`
  - `iqn.2001-04.com.example:storage.tape1.sys1.xyz`
  - `iqn.2001-04.com.example:storage.disk2.sys1.xyz`
- To specify an EUI, use the `eui.02004567A425678D` format.

Examples

- Sets the target name using an IQN.

```
# iscsi-target Target-7
New iSCSI Target configuration mode
# target-name iqn.2001-04.com.example:disk7.Balboa
#
```

- Sets the target name using an EUI.

```
# iscsi-target Target-8
New iSCSI Target configuration mode
# target-name eui.02004567A425678D
#
```

Chapter 38. iSCSI Volume configuration mode (Type 9235)

This chapter provides an alphabetic listing of commands that are available in iSCSI Volume configuration mode. To enter this configuration mode, use the Global **iscsi-volume** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in iSCSI Volume configuration mode.

directory

Specifies the name of the directory.

Syntax

directory *name*

Parameters

name Specifies the name of the subdirectory.

Guidelines

The **directory** command specifies the name of the directory under which to make the files on the iSCSI volume available in the local: and logstore: directories in each application domain.

Examples

- Makes the files on the `VOL1` iSCSI volume accessible in the `local:` and `logstore:` directories, under the `StoreDisk-1` subdirectory.

```
# iscsi-volume VOL1
NEW iSCSI Volume configuration mode
# directory StoreDisk-1
#
```

lun

Specifies the logical unit number.

Syntax

lun *LUN*

Parameters

LUN Specifies the logical unit number.

Guidelines

The **lun** command specifies the logical unit number (LUN). Use an integer in the range of 0 through 255.

Examples

- Makes LUN 33 the VOL2 iSCSI volume .

```
# iscsi-volume VOL2
New iSCSI Volume configuration mode
# lun 22
#
```

---

read-only

Defines whether to makes the files on the iSCSI volume read-only.

Syntax

read-only {**on** | **off**}

Parameters

on Sets the file to read-only.

off (Default) Sets the files to read-write.

Guidelines

The **read-only** command indicates whether the files on the iSCSI volume are read-only or read-write.

Examples

- Sets the files on VOL3 to read-only.

```
# iscsi-volume VOL3
New iSCSI Volume configuration mode
# read-only on
#
```

---

target

Specifies the name of the iSCSI target to which to bind.

Syntax

target *name*

Parameters

name Specifies the name of the iSCSI target.

Guidelines

The **target** command specifies the name of the iSCSI target instance to which to bind the iSCSI volume.

Examples

- Makes the T-100 target for the VOL5 iSCSI volume.

```
# iscsi-volume VOL5
New iSCSI Volume configuration mode
# target T-100
#
```

Chapter 39. Kerberos KDC Server configuration mode

This chapter provides an alphabetic listing of commands that are available in Kerberos KDC Server configuration mode.

To enter this configuration mode, use the Global **kerberos-kdc** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Kerberos KDC Server configuration mode.

port

Specifies the UDP or TCP port that the target Kerberos KDC server monitors.

Syntax

port *port*

Parameters

port Identifies the listening port on the Kerberos server. The default is 88.

Related Commands

server, **tcp**

Examples

- Specifies port 8888 as the monitored port.

```
# port 8888
#
```

realm

Specifies the realm (administrative domain) to support.

Syntax

realm *name*

Parameters

name Specifies the name of the Kerberos realm.

Guidelines

You must specify a Kerberos realm to complete KDC configuration.

Related Commands

server

Examples

- Provides the name of the Kerberos realm.

```
# realm us.ibm.com
#
```

server

Identifies the server by domain name or IP address.

Syntax

server *server*

Parameters

server Specifies the host name or IP address of the Kerberos KDC server.

Guidelines

You must specify a Kerberos KDC Server to complete the configuration.

Related Commands

port, **tcp**

Examples

- Identifies the Kerberos KDC Server by domain name.

```
# server Furio
#
```

- Identifies the Kerberos KDC Server by IP address.

```
# server 192.168.12.12
#
```

tcp

Enables the use of TCP as the transport layer protocol.

Syntax

tcp

no tcp

Guidelines

UDP is the default transport protocol to access the Kerberos KDC server. Use the **no tcp** command to restore the default state, which is to use UDP as the transport layer protocol.

Related Commands

port, **server**, **udp-timeout**

Examples

- Specifies TCP as the transport layer protocol.

```
# tcp
#
```

- Restores UDP, the default, as the transport layer protocol.

```
# no tcp
#
```

udp-timeout

When using UDP as the transport protocol, specifies the number of seconds to wait for a server response.

Syntax

udp-timeout *time*

Parameters

time Specifies the maximum time to wait for a Kerberos KDC Server response. Use an interval in the range of 1 through 60. The default is 5.

Guidelines

Only meaningful when the default UDP protocol is used to access the Kerberos KDC server.

Related Commands

tcp

Examples

- Specifies a 7 second timeout value.

```
# udp-timeout 7
#
```

Chapter 40. Kerberos Keytab configuration mode

This chapter provides an alphabetic listing of commands that are available in Kerberos Keytab configuration mode.

To enter this configuration mode, use the Crypto **kerberos-keytab** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

filename

Specifies the file that contains the Kerberos keytab, a file that contains keys used to decrypt Kerberos tickets.

Syntax

filename *URL*

Parameters

URL Identities the fully qualified name of the keytab file in the cert: directory.

Guidelines

The **filename** command specifies the location of the keytab file. The keytab file is Kerberos-generated and must be uploaded to the cert: directory on the appliance.

Examples

- Identifies the `KKTab` keytab file in the `cert:` directory.

```
# filename cert:///KKTab
#
```

use-replay-cache

Controls the caching of Kerberos authenticators on tickets for Kerberos principals in this keytab.

Syntax

use-replay-cache {**on** | **off**}

Parameters

on (Default) Enables caching of Kerberos authenticators.

off Disables caching of Kerberos authenticators.

Examples

- Disables the authenticators cache.

```
# use-replay-cache off
#
```

Chapter 41. LDAP Search Parameters configuration mode

This chapter provides an alphabetic listing of commands that are available in LDAP Search Parameters configuration mode.

To enter this configuration mode, use the Global **ldap-search-parameters** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

base-dn

Specifies the base DN to begin the search.

Syntax

base-dn *DN*

Parameters

DN Specifies the base DN for the search.

Guidelines

The **base-dn** command specifies the distinguished name (DN) relative to which the LDAP search is to be performed. This value identifies the entry level of the tree used by the **scope** command.

Related Commands

scope

filter-prefix

Specifies the prefix of the LDAP filter expression.

Syntax

filter-prefix *prefix*

Parameters

prefix Specifies the prefix of the filter expression.

Guidelines

The **filter-prefix** command specifies the string prefix to construct an LDAP filter expression, as defined in *LDAP: String Representations of Search Filters*. This string is added before the user name to construct the LDAP filter to search for the DN of the user.

If the prefix is `(&(mail=` and the user name is `bob@example.com` and the suffix is `)(c=US))`, the LDAP search filter would be `(&(mail=bob@example.com)(c=US))`.

You can use the **filter-suffix** to append a string to the LDAP filter expression to complete the search filter.

Related Commands

filter-suffix

Examples

Creates the LDAP filter expression `(&(mail=bob@example.com)(c=US))` based on `bob@example.com` as the user name.

```
# filter-prefix "(&(mail="
# filter-suffix ")(c=US))"
#
```

---

filter-suffix

Specifies the suffix of the LDAP filter expression.

Syntax

filter-prefix *suffix*

Parameters

suffix Specifies the suffix of the filter expression.

Guidelines

The **filter-suffix** command specifies the string suffix to construct an LDAP filter expression, as defined in *LDAP: String Representations of Search Filters*. This string is added after the user name to construct the LDAP filter to search for the DN of the user.

If the prefix is `(&(mail=` and the user name is `bob@example.com` and the suffix is `)(c=US))`, the LDAP search filter would be `(&(mail=bob@example.com)(c=US))`.

You must use the **filter-prefix** to add the prefix string to the LDAP filter expression to complete the search filter.

Related Commands

filter-prefix

Examples

Creates the LDAP filter expression `(&(mail=bob@example.com)(c=US))` based on `bob@example.com` as the user name.

```
# filter-prefix "(&(mail="
# filter-suffix ")(c=US))"
#
```

---

returned-attribute

Specifies the attribute to return for each match.

Syntax

returned-attribute *attribute*

Parameters

attribute
Specifies the name of the attribute to return. The default is `dn`.

Guidelines

The **returned-attribute** command specifies the name of the attribute to return for each entry that matches the search criteria.

---

scope

Indicates the depth of the search

Syntax

scope {**base** | **one-level** | **subtree**}

Parameters

base Searches the entry level of the tree only.

one-level
Searches the entry level of the tree and any object that is one-level below the input.

subtree
(Default) Search the entry level of the tree and all of its descendents.

Guidelines

The **scope** command indicates the depth of the LDAP search. The entry level of the tree is defined by the **base-dn** command.

Related Commands

base-dn

Chapter 42. Load Balancer Group configuration mode

This chapter provides an alphabetic listing of commands that are available in Load Balancer Group configuration mode.

To enter this configuration mode, use the Global **loadbalancer-group** command. The global command creates the Load Balancer Group if it does not exist. While in this mode, define the Load Balancer Group.

To use a Load Balancer Group, provide the name of a group as the backend destination for a top-level DataPower service.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Load Balancer Group configuration mode.

algorithm

Specifies the server selection algorithm.

Syntax

algorithm {**first-alive** | **hash** | **least-connections** | **round-robin** | **weighted-round-robin**}

Parameters

first-alive
Uses the concept of a primary server and backup servers. When the primary server is healthy, all connections are forwarded to this server. When the primary server is quarantined or convalescent, connections are forwarded to backup servers. The primary server is the first server in the members list.

hash Uses the IP address of the client or the value of an HTTP header as the basis for server selection.

When using an HTTP header, use the **load-balancer-hash-header** command to identify the header to read. This command is available only for Multi-Protocol Gateway and Web Service Proxy services.

The same client is served by the same server. This algorithm is used in applications that require the storage of server-side state information, such as cookies.

least-connections
Maintains a record of active server connections and forward a new connection to the server with the least number of active connections.

round-robin
(Default) Maintains a list of servers and forwards a new connection to the next server on the list.

weighted-round-robin
Maintains a weighted list of servers and forwards new connections in proportion to the weight (or preference) of each server.

Guidelines

The **algorithm** command specifies the server selection algorithm. A request to connect to a Load Balancer Group results in a healthy server being selected from the pool according to the server selection algorithm. The algorithm provides a method for selecting which healthy server receives an incoming client request.

Related Commands

load-balancer-hash-header (Multi-Protocol Gateway), **load-balancer-hash-header** (Web Service Proxy)

Examples

- Specifies that server selection is accomplished by a weighted-round-robin algorithm.

```
# algorithm weighted-round-robin
#
```

---

damp

Specifies the dampening period for a quarantined server.

Syntax

damp *interval*

Parameters

interval
Specifies the number of seconds that a server remains in an `softdown` state. Use a value in the range of 1 through 86400. The default is 120.

Guidelines

The **damp** command specifies the dampening period for a member server. The dampening period is the amount of time that a server is removed from the Load Balancer Group because of a failure to connect during a normal HTTP or TCP transaction. Such a server is consider to be *quarantined* and is placed in the `softdown` state. When this specified interval expires, the server is restored to the Load Balancer Group and placed in the `up` state.

This command does not impact servers that are in the `down` state.

Related Commands

algorithm, **server**

Examples

- Specifies a dampening period of 5 minutes.

```
# damp 600
#
```

giveup-when-all-members-down

Specifies the connection-behavior when no member is `up`.

Syntax

giveup-when-all-members-down {**on** | **off**}

Parameters

on Does not forward the connection to any member. Makes the next attempt when at least one members is in the `up` state.

off (Default) Selects the first member in the `down` state and forwards the connection to this server.

Examples

- Disables connection attempts if all members are in the `down` state. Subsequently restores the default value.

```
# giveup-when-all-members-down on
⋮
# giveup-when-all-members-down off
#
```

health-check

Defines a periodic health check procedure.

Syntax

health-check *state URI port type use-SOAP send-SOAP timeout frequency XPath filter SSL*

Parameters

state Controls whether to perform the periodic health check.

on Enables the health check.

off (Default) Disables the health check.

URI When the check type is **Standard**, specifies the non-server (file path) portion of the target URI. That is, specify the URI to receive the client request that is generated by the rule. The default is /.

This URI is used with the specified remote port.

port Specify the port on the target server to receive the query. The default is 80.

You can override this value for one or more members of the Load Balancer Group with the *health-port* argument of the **server** command.

The response from the server is evaluated to determine the health status of each member server in the group. The request is sent to the target URI and remote port.

type Controls the type of check to perform.

IMSConnect
Specifies that the group consists of IMS Connect servers. Performs a TCP ping.

LDAP Specifies that the group consists of LDAP servers. Performs a TCP ping.

Standard
(Default) Specifies that the group does not consist of LDAP or IMS Connect servers.

use-SOAP
When the check type is **Standard**, specifies the HTTP method used to access the target URI.

on (Default) Accesses the target URI with an HTTP **POST** operation (by posting a SOAP message).

off Accesses the target URI with an HTTP **GET** operation.

send-SOAP
When *use-SOAP* is **on**, specifies the SOAP message to send as a client request. The default is the healthcheck.xml file in the store: directory (`store:///healthcheck.xml`). When *use-SOAP* is **off**, use two double quotation marks (`""`).

timeout
Specifies the number of seconds for the completion of the health check. Use a value in the range of 2 through 86400. The default is 10.

If successful, the server is deemed healthy and is marked as `up`; otherwise, the server is deemed convalescent and is marked as `down`.

frequency
Specifies the number of seconds between health checks. Use a value in the range of 5 through 86400. The default is 180.

XPath When the check type is **Standard**, use with the *filter* argument to specify the XPath expression that must be found in a valid server response.

filter When the check type is **Standard**, specifies the style sheet to filter the server response. The default is the healthcheck.xsl file in the store: directory (`store:///healthcheck.xsl`).

This style sheet uses the specified *XPath* argument as input and scans the server response for its presence. If found, the server is deemed healthy and is marked as `up`; otherwise, the server is deemed convalescent and is marked as `down`.

SSL When the check type is **Standard**, specifies the name of an existing SSL Proxy Profile to provide the resources for a secure connection.

Guidelines

A health check is essentially a scheduled rule that sends the same request to each server member. The successful completion of the health check requires that the server passes normal TCP and HTTP connection criteria, depending on check type (servers in the load balancer group). Optionally, the health check contains a filter to test the response from the server. If the filter accepts the response, the server is considered to be healthy; otherwise, it is considered to be convalescent.

Related Commands

server

Examples

- Specifies a periodic health check for members.

```
# health-check on cgi-bin/x.cgi 80 Standard
on store:///identity.xsl 4 60 / store:///healthcheck.xsl sslProxy1
#
```

masquerade

Specifies the host name to provide to the backend server.

Syntax

masquerade {**on** | **off**}

Parameters

on Passes the name of the Load Balanced Group name to the backend server.

off (Default) Passes the name of the member server to the backend server.

Examples

- Passes the name of the Load Balancer Group as the host name to the backend server.

```
# masquerade on
#
```

server

Adds a server to a Load Balancer Group.

Syntax

server *address* [*weight*] [*mapped-port*] [“”] [*health-port*]

Parameters

address Specifies the name or IP address of the server.

weight If the algorithm is **weighted-round-robin** only, specifies the relative weight (preference). Use a value in the range of 1 through 65000. The default is 1.

mapped-port
Specifies the member-specific target port or retain the default value (0) to use the DataPower service-defined port. Use a value in the range of 0 through 65535. The default is 0.

health-port
Specifies the member-specific health port or retain the default value (0) to use the Load Balancer Group-defined port. Use a value in the range of 0 through 65535. The default is 0.

Guidelines

The **server** command adds a server to the Load Balancer Group. When defining a member server, you can optionally specify the ports to use for sending client requests for send a health check.

If the server selection algorithm is **first-alive**, the order is significant. The first server is the primary server, while subsequent entries serve as backup servers. For all other algorithms, the order is not significant.

If the server selection algorithm is **weighted-round-robin**, specify the relative preference of a server. The greater the value, the more likely this server is to receive a connection request. For example there are three members (`A`, `B`, and `C`) that have the assigned weights of 100, 60, and 40, respectively. Because of the weighting, `A` receives 50% of requests, `B` receives 30% of requests, and `C` receives 20% of requests.

Related Commands

algorithm, **health-check**

Examples

- Adds `ragnarok.datapower.com` with a weight of 5 to the Load Balancer Group.

```
# server ragnarok.datapower.com 5
#
```

---

try-every-server

Specifies the retry-behavior for a failed attempt.

Syntax

try-every-server {**on** | **off**}

Parameters

on Sends the requests to each server until one responds or all fail. Each server that fails is set to the `softdown` state.

off (Default) Does not attempt to contact other members.

Guidelines

When enabled, each server in the group, including those in a dampened state, are tried before failing the connection attempt.

Examples

- Retries all member server before failing.

```
# try-every-server on
#
```

Chapter 43. Log Target configuration mode

This chapter provides an alphabetic listing of commands that are available in Log Target configuration mode. To enter this configuration mode, use the Global **logging target** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Log Target configuration mode.

ansi-color

Enables or disables a multicolored console log display.

Syntax

ansi-color {**on** | **off**}

Parameters

on enables a multicolored log display in which events of different priorities are displayed in different colors.

off provides a monochrome display.

Guidelines

Meaningful only when the logging type is **console**, and otherwise unused.

archive-mode

Specifies an archival behavior for file-based logs.

Syntax

archive-mode {**rotate** | **upload**}

Parameters

rotate (default) Specifies that when a log file reaches its maximum size, the log is rotated as specified by the **rotate** command.

upload
Specifies that when a log file reaches its maximum size, the file is uploaded to a specified site for remote storage.

Guidelines

The **archive-mode** command is required when the log type is either **file** or **nfs**; otherwise, it is not used.

After specifying upload mode, you must use the **remote-address**, **remote-directory**, **remote-login**, and **upload-method** commands to enable transfer of the log file to the remote site.

Related Commands

backup, **email-addr**, **encrypt**, **format**, **local-file**, **local-ident**, **remote-addr**, **remote-login**, **rotate**, **sender-addr**, **sign**, **size**, **timestamp**, **upload-method**

Examples

- Specifies an archive type of `upload`.

```
# archive-mode upload
#
```

- Specifies an archive type of `rotate`, which restores the default state.

```
# archive-mode rotate
#
```

---

backup

Specifies a backup for the current log.

Syntax

backup *log*

Parameters

log Specifies the name of an existing log, of any log type).

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

---

email-address

Specifies the email address of a remote recipient of SMTP log messages.

Syntax

e-mail-address *string*

Parameters

string Specifies the remote email address.

Guidelines

The **email-address** command is only used when the log type is **smtp**.

Related Commands

interval, **local-address**, **local-ident**, **remote-address**

Examples

- Assigns an email address.

```
# email-address techDesk@datapower.com
#
```

---

encrypt

Enables S/MIME encryption of logs.

Syntax

encrypt *certAlias* **smime**

Parameters

certAlias
Specifies a string that contains the alias for a certificate file that contains the public key of the message recipient.

smime
Specifies the required keyword for the encryption method.

Guidelines

The **encrypt** command is only used when the log type is **file**, **nfs**, or **smtp** to enable S/MIME (Secure Multipurpose Internet Mail Extension) encryption.

- When enabled and the log type is **file** or **nfs**, the log is encrypted (and optionally signed) when the log is rotated or uploaded.
- When enabled and the log type is **smtp**, each log entry is encrypted (and optionally signed) upon transmission.

Related Commands

sign, **type**

---

event

Adds an event class and a priority to the current log.

Syntax

event *event-class priority*

Parameters

event-class
Specifies the name of an event-class (a set of related events).

priority
Identifies the event priority.

Guidelines

Log priority is characterized (in descending order of importance) as `emergency`, `alert`, `critic`, `error`, `warn`, `notice`, `info`, and `debug`. Identification of a priority value specifies that all events of greater or equal criticality to the argument are logged.

You can use the **show logging priority** command to display a list of event priorities.

You can use the **show logging event** command to display a list of event classes.

Related Commands

show logging event, **show logging priority**

Examples

- Specifies which event classes and which event priorities to log.

```
# event schema error
# event xmlfilter error
# event crypto error
# event ssl error
# event auth warning
#
```

event-code

Specifies an event code included in the current log.

Syntax

event-code *value*

Parameters

value Identifies the hexadecimal value of the event code.

Guidelines

This command allows only log message that contain specified event codes to be written to the current log target. Thus, it is possible to create a log target that only collects log messages for a particular set of event codes, for example, Operational State down.

Use the **View List of Event Codes** from the WebGUI to view a list of all event codes.

Related Commands

event-filter

Examples

- Creates a file-based log that contains only XML parser events.

```
# type file
# event-code 0x00030001
# event-code 0x00030002
# event-code 0x00030003
# event-code 0x00030004
# event-code 0x00030005
# event-code 0x00030006
# event-code 0x00030007
# event-code 0x00030008
# event-code 0x00030009
# event-code 0x0003000a
#
```

event-detection

Suppresses identical events.

Syntax

event-detection {**on** | **off**}

Parameters

on Suppresses the writing of identical events to the log for the specified suppression period.

off (Default) Identical events are written to the log.

Guidelines

The **event-detection** command allows for the suppression of identical log events that are generated by the same configuration object over a configurable time period. When enabled, the Log Target retains a reference to each processed event for a configurable period of time. Until this period expires, the Log Target will not process the same event from the same configuration object.

Related Commands

suppression-period

---

event-filter

Specifies an event code excluded from the current log.

Syntax

event-filter *value*

Parameters

value Specifies the hexadecimal value of the event code.

Guidelines

Event filters provide for the exclusion of log message that contain specified event codes from the current log target.

Use the **View List of Event Codes** from the WebGUI to view a list of all event codes.

Related Commands

event-code

Examples

- Creates a file-based log excluding XML parser events.

```
# type file
# event-filter 0x00030001
# event-filter 0x00030002
# event-filter 0x00030003
# event-filter 0x00030004
# event-filter 0x00030005
# event-filter 0x00030006
# event-filter 0x00030007
# event-filter 0x00030008
# event-filter 0x00030009
# event-filter 0x0003000a
#
```

facility

Specifies the syslog facility.

Syntax

facility *facility*

Parameters

facility Identifies the syslog facility.

Guidelines

facility is used only when the logging type is **syslog** or **syslog-ng**.

Related Commands

local-address, **local-ident**, **remote-address**

Examples

- Specifies the syslog facility, `local0`.

```
# type syslog
# local address 10.10.13.4
# remote-address 172.16.100.1
# facility local0
#
```

feedback-detection

Suppresses events from the log subsystem itself.

Syntax

feedback-detection {**on** | **off**}

Parameters

on Suppresses all log events triggered by the logging subsystem.

off (Default) Suppresses log events triggered by the target itself, but writes events that are generated by other log targets.

Guidelines

The **feedback-detection** command allows for the suppression of log events that are triggered by the log subsystem itself. Log targets always suppress log events triggered by the target itself, but write events that are generated by other log targets. Under certain circumstances with two or more log targets, those events can create a positive feedback loop that might cause resource contention. Enabling feedback detection suppresses all log events triggered by the logging subsystem.

format

Specifies the format in which events are added to the log.

Syntax

format {**text** | **raw** | **xml** | **cbe** | **csv**}

Parameters

text Specifies the log format as formatted text

raw Specifies the log format as unformatted text

xml Specifies the log format as XML

cbe Specifies the log format as IBM Common Base Event

csv Specifies the log format as comma-separated

Guidelines

Use the **show logging format** command to display a list of available log formats.

group (deprecated)

Comments

Deprecated command. Has no effect.

local-address

Specifies the local address over which log events are transmitted to a remote recipient.

Syntax

local-address *address* [*port*]

Parameters

address is the IP address of the interface over which log events are transmitted.

port is the optional UDP or TCP port number used to transmit log events. For TCP, the default is 25. For UDP, the default is 514.

Guidelines

When the log type is **smtp**, the use of the **local-address** command is required. For this log type, identification of a TCP port is optional.

When the log type is **syslog** or **syslog-ng**, the use of the **local-address** command is optional. For these log types, identification of a UDP port is optional.

For all other log types, the **local-address** command is not used.

Related Commands

type

Examples

- Specifies the local interface used to transmit log messages to an SMTP server. Uses a default port value of TCP port 25

```
# type smtp
# local-address 10.10.13.4
#
```

local-file

Specifies a local file that will store log messages.

Syntax

local-file *URL*

Parameters

URL Specifies the file to store log messages and takes the `logstore:///`*`filename`* form.

Guidelines

When the log type is **file**, the use of the **local-file** command is required. For all other log types, it is not used.

Related Commands

type

local-ident

Specifies a string that can be used by a remote recipient to identify this log.

Syntax

local-ident *id*

Parameters

id Identifies the current log.

Guidelines

When the log type is **smtp**, **syslog**, or **syslog-ng**, the use of the **local-ident** command is optional. For all other log types, it is not used.

Related Commands

type

nfs-file

Specifies the path to the mount file.

Syntax

nfs-file *filename*

Parameters

filename
Specifies the path to the log file relative to the NFS mount point.

Guidelines

The file name can only use the characters a-z, A-Z, 0-9, or an underscore. The path can have subdirectories that are delimited by a slash.

The file must have write permission.

Related Commands

nfs-static-mount, **type**

nfs-static-mount

Assigns an static mount.

Syntax

nfs-static-mount *name*

Parameters

name Specifies the name of an existing NFS Static Mount.

Guidelines

When the log type is **nfs**, specifies the NFS Static Mount point to write the log over NFS.

Related Commands

nfs-file, **type**

object

Adds an object filter to the current log target.

Syntax

object *type name follow-references*

Parameters

type Specifies an object type. This filter will restrict messages to only messages generated by the selected object type.

name Specifies the name of an existing object instance of the selected object type.

follow-references
Indicates whether to log messages for objects that the selected object instance references. For example, an XSL Proxy references an XML Manager object as well as many other objects.

on Logs messages for all objects that the selected object instance references.

off (Default) Logs messages for the selected object instance only.

Guidelines

Use the **object** command to enable a finer granularity of log content.

Object filters allow only those log messages that are generated by specific objects to be written to the log target. Object filters are based on object type, such as XSL Proxy, and based on specific instances of that object type. Therefore, you can only

create a log target to collect log messages for a particular instance of a particular object type. For example, you can create a log target to write messages associated with the xyz XSL Proxy only.

Examples

- Adds an object filter to the current log to log messages for the Proxy-1 XSL Proxy only.

```
# object XSLProxy Proxy-1
#
```

- Adds an object filter to the current log to log messages for the Proxy-2 XSL Proxy object and all object that the Proxy-2 XSL Proxy object references.

```
# object XSLProxy Proxy-2 on
#
```

rate-limit

Limits the number of log transactions per second

Syntax

rate-limit *seconds*

Parameters

seconds

Specifies the maximum number of transactions per second. Use an integer in the range of 1 through 1000. The default is 100.

Guidelines

Meaningful when the log type is **nfs**, **smtp**, **snmp**, **soap**, **syslog**, or **syslog-ng**. Otherwise, it is not used.

Examples

- Limits transactions to a maximum of 50 per second.

```
# rate-limit 50
#
```

remote-address

Specifies the destination address of log messages or the log itself.

Syntax

remote-address *host*

Parameters

host Identifies the host name or IP address of the remote server.

Guidelines

The **remote-address** command specifies the destination of transmitted log messages or the location of an uploaded log file. This command is relevant in the following situations:

- When the log type, as specified by the **type** command, is **smtp**, **syslog**, or **syslog-ng**
- When the log type, as specified by the **type** command, is **file** and the archive mode, as specified by the **archive-mode** command, is **upload**

Use the **remote-address** command with the **remote-port** command to define the destination of transmitted log messages.

When using TCP-based, network log targets, instead of specifying the IP address or host name of a remote server, you can specify the name of an existing load balancer group. In this situation, the same load balancer group must be assigned to the `default` XML Manager in the domain with the XML Manager **loadbalancer-group** command. To create a load balancer group, use the Global **load-balancer-group** command.

You can use SSL to establish a secure connection to a remote server by setting the values of the **remote-server** and the **remote-port** commands to the values of a local SSL Proxy service on the appliance. The local SSL Proxy service, as defined by the Global **sslforwarder** command, can then forward log messages over a secure connection to the remote server.

Related Commands

archive-mode, **load-balancer-group** (Global), **loadbalancer-group** (XML Manager), **remote-port**, **sslforwarder** (Global), **type**

Examples

- Specifies the address of an SMTP server. Uses a default TCP port value of 25.

```
# type smtp
# local address 10.10.13.4
# remote-address ragnarok.datapower.com
#
```

- Specifies the address of a syslog daemon. Uses a default UDP port value of 514.

```
# type syslog
# local address 10.10.13.4
# remote-address 172.16.100.1
#
```

- Specifies the recipient address for an uploaded log file.

```
# type file
# archive-mode upload
# remote-address 172.16.100.1
#
```

remote-directory

Specifies the remote directory where uploaded logs are stored.

Syntax

remote-directory *file-path*

Parameters

file-path
Identifies the writable remote directory that stores uploaded log files.

Guidelines

remote-directory is used only in the following situations:

- The log type is **file**.
- The archive mode is **upload**.
- The upload mode is **scp**, **ftp**, or **sftp**.

To denote an absolute directory from the root directory, specify a single forward slash character or equivalent encoded character (`%2F`) before the fully-qualified file name (for SCP or SFTP, specify `/`*`file-path`*; for FTP, specify `%2F`*`file-path`*). The path in the URL resolves to `//`*`file-path`* for SCP or SFTP and `/%2F`*`file-path`* for FTP.

To denote a directory that is relative to the user's home directory, do not specify a forward slash character or equivalent encoded character before the fully-qualified file name (for example, specify *`file-path`*). The path in the URL resolves to `/`*`file-path`*.

Related Commands

archive-mode, **remote-address**, **type**, **upload-mode**

Examples

- Specifies the remote directory for an uploaded log file that is relative to the user's home directory.

```
# type file
# archive-mode upload
# upload-method sftp
# remote-address 172.16.100.1:2121
# remote-directory logs/
#
```

- Specifies the remote directory for an uploaded log file that is absolute to the root directory.

```
# type file
# archive-mode upload
# upload-method ftp
# remote-address 172.16.300.254:2123
# remote-directory %2Flogs/
#
```

remote-login

Specifies the user name to use when uploading a log file to a remote server.

Syntax

remote-login *username* [*password*]

Parameters

username
: Specifies the user name to access a recipient of uploaded logs.

password
: Specifies the password for the user name account.

Guidelines

The **remote-login** command is used only if the log type is **file** and the archive-mode is **upload**.

If a password is not specified, it must be provided during the upload session.

Related Commands

archive-mode, **remote-address**, **remote-directory**, **type**

Examples

- Specifies the recipient address, username and password, and remote directory for an uploaded log file.

```
# type file
# remote-address 172.16.100.1
# remote-login jrb brj
# remote-directory logs/
#
```

---

remote-port

Specifies the listening port on the remote server.

Syntax

remote-port *port*

Parameters

port Specifies the destination port that monitors traffic. The default is 514.

Guidelines

The **remote-port** command specifies the listening port on the remote server. This command is relevant only when the log type, as specified by the **type** command, is **smtp**, **syslog**, or **syslog-ng**.

Use the **remote-port** command with the **remote-address** command to define the destination of transmitted log messages.

You can use SSL to establish a secure connection to a remote server by setting the values of the **remote-server** and the **remote-port** commands to the values of a local SSL Proxy service on the appliance. The local SSL Proxy service, as defined by the Global **sslforwarder** command, can then forward log messages over a secure connection to the remote server.

Related Commands

remote-address, **sslforwarder** (Global), **type**

Examples

- Specifies the address of an SMTP server that listens on port 5400.

```
# type smtp
# local address 10.10.13.4
# remote-address ragnarok.datapower.com
# remote-port 5400
#
```

retry (deprecated)

Comments

Deprecated command. Has no effect.

rotate

Sets the maximum number of file rotations.

Syntax

rotate *count*

Parameters

count Specifies how many times to rotate a log file. Use an integer in the range of 1 through 100. The default is 3.

Guidelines

The **rotate** command specifies the maximum number of rotations for the log file. Depending on the Machine Type of the appliance, the location of the file can be the local file system, the compact flash, or the hard disk array.

Type 7993 (9003)
Supports only the local file system.

Type 9235
Supports the local file system and the model-specific, auxiliary data storage (compact flash or hard disk array).

- The compact flash provides 512 MB of storage.
- The hard disk array provides 70 GB of storage.

Assuming a file name of `CryptoLog` and three rotations, the directory that contains the log file can consist of up to the following four local files:

`CryptoLog`
The current log file

`CryptoLog1`
The log file that was most recently archived.

`CryptoLog2`
The log file that was next most recently archived.

`CryptoLog3`
The oldest log file

This command is meaningful only when the following conditions are met:

- The log type, as specified by the **type** command is **file**.
- The archival mode, as specified by the **archive-mode** command, is **rotate**.

Related Commands

archive-mode, **type**

sender-address

Specifies the email address of the sender

Syntax

sender-address *string*

Parameters

string Specifies the local email address.

Guidelines

The **sender-address** command is only used when the log type is **smtp**.

Related Commands

type

sign

Enables the S/MIME signing of logs.

Syntax

sign *idCred* **smime**

Parameters

idCred Identifies an existing Identification Credentials List, which is a matched public certificate, private key pair.

smime
Specifies the signature method with the required keyword.

Guidelines

The **sign** command is only used when the log type is **file** or **smtp** to enable S/MIME (Secure Multipurpose Internet Mail Extensions) signing.

- When enabled and the log type is **file**, the log is signed (and optionally encrypted) when the log is rotated or uploaded.
- When enabled and the log type is **smtp**, each log entry is signed (and optionally encrypted) on transmission.

Related Commands

encrypt, **idcred**, **type**

Examples

- Enables the signature of `file` and `smtp` logs with the S/MIME signature algorithm and the `Bob` Identification Credentials Set.

```
# sign Bob smime
#
```

size

Sets the maximum size of a local log file.

Syntax

size *log-size*

Parameters

log-size

Specifies the maximum size of the file in kilobytes. Use an integer in the range of 100 through 50000. The default is 500.

Guidelines

The **size** command sets the maximum size of a local log file in kilobytes. Depending on the Machine Type of the appliance, the location of the file can be the local file system, the compact flash, or the hard disk array.

Type 7993 (9003)

Supports only the local file system.

Type 9235

Supports the local file system and the model-specific, auxiliary data storage (compact flash or hard disk array).

- The compact flash provides 512 MB of storage.
- The hard disk array provides 70 GB of storage.

This command is only meaningful when the log type, as specified by the **type** command, is **file**.

Related Commands

type

---

smtp-domain

Specifies the fully-qualified domain name of the SMTP client.

Syntax

smtp-domain *domain*

Parameters

domain Specifies the domain name of the SMTP client.

Guidelines

The **smtp-domain** command specifies the fully-qualified domain name of the SMTP client. This domain name is sent as part of the SMTP session invitation (**HELO** command).

This command is meaningful only in the following situations:

- The log type, as specified by the **type** command, is **smtp**.
- The log type, as specified by the **type** command, is **file**, and the upload method, as specified by the **upload-method** command is **smtp**.

Related Commands

type, **upload-method**

Examples

- Specifies the recipient of SMTP domain.

```
# type smtp
# smtp-domain popServer-1.datapower.com
#
```

soap-version

Specifies the version of SOAP to use.

Syntax

soap-version {**soap11** | **soap12**}

Parameters

soap11
SOAP targets use SOAP 1.1.

soap12
SOAP targets use SOAP 1.2.

Guidelines

When the log type is **soap**, specifies the version of SOAP for use by SOAP log targets.

Related Commands

type

ssl

Assigns an SSL Proxy Profile for SOAP-based log over HTTPS.

Syntax

ssl *name*

Parameters

name Is an existing SSL Proxy Profile to use.

Guidelines

Use the **ssl** command to assign an SSL proxy profile to use when target type is **soap** and URL uses HTTPS.

Related Commands

type, **url**

suppression-period

Interval to suppress identical events.

Syntax

suppression-period *interval*

Parameters

interval
Specifies the interval to suppress identical events in seconds. The default is 10.

Related Commands

event-detection

timeout (deprecated)

Comments

Deprecated command. Has not effect.

timestamp

Specifies the timestamp format.

Syntax

timestamp {**numeric** | **syslog**}

Parameters

numeric
(default) Specifies a numeric timestamp format.

syslog Specifies a syslog timestamp format.

type

Identifies the logging model.

Syntax

type {**cache** | **console** | **file** | **nfs** | **smtp** | **snmp** | **soap** | **syslog** | **syslog-ng**}

Parameters

cache Writes log entries to system memory.

console
Writes log entries to the console screen.

file Writes log entries to a file on the appliance.

nfs Writes log entries to a file on a remote NFS mount.

smtp Forwards log entries via email to a specified recipient.

snmp Forwards log entries as SNMP traps.

soap Forwards log entries as SOAP messages to a specified recipient.

syslog Forwards log entries to a remote syslog daemon.

syslog-ng
Forwards log entries to a remote syslog daemon.

Guidelines

For all log types, use the **event** command to specify log contents.

Cache logs require no configuration beyond the identification of the logging type. You can, however, optionally use the **format**, **size**, and **timestamp** commands to customize log behavior.

- For a *console* log, no additional configuration is required. You can, however, optionally use the **format** and **timestamp** commands to customize log behavior.
- For a *file-based* log, you must specify the name of the log file with the **local-file** command. You can optionally use the **size** command to specify the file size (in kilobytes), the **rotate** command to specify the number of file rotations, the **backup** command to specify a backup log, the **format** command to specify the log format, the **timestamp** command to specify the timestamp method, the **sign** and **encrypt** commands to sign and encrypt logs, the **archive-mode**, **upload-method**, **local-ident**, **remote-addr**, **email-addr**, and **sender-addr** commands to specify an archival and retrieval method.
- For an *SMTP-based* log, you must use the **local-addr** and **remote-addr** commands to specify source and destination IP addresses and the **email-addr** command to specify a destination email address. You can optionally use the **rate-limit** command to control event message flow, the **local-port** and **remote-port** commands to identify nonstandard source or destination ports, the **sender-addr** command to specify a pseudo recipient email address, the **local-ident** command to specify a local identifier, the **backup** command to specify a backup log, the **format** command to specify the log format, and the **sign** and **encrypt** commands to sign and encrypt logs.
- For a *SOAP-based* log, you must use the **url** command to specify the recipient of SOAP-enveloped log events and use the **rate-limit** command to throttle the event flow. You may optionally use the **backup** command to specify a backup log.
- For a *system* log, you must use the **remote-addr** command to specify a remote IP address. You may optionally use the **remote-port** command to specify a nonstandard destination port number, the **local-addr** and **local-port** commands to specify a specific local interface used for syslog transmissions, the **local-ident** command to specify a local identifier, and the **facility** command to identify the syslog facility.

Refer to the Global **sslforwarder** command for information on enabling the transmission of syslog-ng events via a secure SSL connection.

Related Commands

email-address, **facility**, **interval**, **local-address**, **local-file**, **local-ident**, **remote-address**, **rotate**, **size**, **sslforwarder** (Global)

---

upload-method

Specifies the protocol to upload a file-based log to a remote storage site.

Syntax

upload-method {**ftp** | **scp** | **sftp** | **smtp**}

Parameters

ftp Identifies the File Transfer Protocol.

scp (Default) Identifies the Secure Copy Protocol.

sftp Identifies the Secure File Transfer Protocol.

smtp Identifies the Simple Mail Transfer Protocol.

Guidelines

upload-method is used only if the log type is **file** and the archive-mode is **upload**.

Related Commands

archive-mode, **backup**, **email-addr**, **encrypt**, **format**, **local-file**, **local-ident**, **remote-addr**, **remote-login**, **rotate**, **sender-addr**, **sign**, **size**, **timestamp**

Examples

- Provides the required information (transfer protocol, recipient address, username and password, and remote directory) to upload a file-based log to a remote storage site.

```
# type file
# upload-method sftp
# remote-address 172.16.100.1
# remote-login jrb brj
# remote-directory logs/
#
```

---

url

Specifies the URL to which SOAP-based log entries are posted.

Syntax

url *URL*

Parameters

URL Identifies the destination to which SOAP-based logs entries are sent.

Guidelines

url is used only if the log type is **soap**.

Related Commands

email-address, **facility**, **interval**, **local-address**, **local-file**, **local-ident**, **remote-address**, **rotate**, **size**, **sslforwarder**

Examples

- Specifies the recipient of SOAP log messages.

```
# type soap
# url http;//ragnarok.datapower.com/logs
#
```

Chapter 44. Matching Rule configuration mode

This chapter provides an alphabetic listing of commands that are available in Matching Rule configuration mode. To enter this configuration mode, use the Global **matching** command.

Matching Rules are used in a processing policy. A processing policy enables an DataPower service to select a style sheet to filter or transform an input document. The selected style sheet can be used with, or instead of, processing instructions in the input document.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about creating and implementing Matching Rules and processing policies.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Matching Rule configuration mode.

combine-with-or

Indicates whether to combine the match criteria with OR semantics or with AND semantics.

Syntax

combine-with-or {**on** | **off**

Parameters

on Uses OR semantics to evaluate the criteria. Only a single match condition needs to be true for the match to succeed.

off (Default) Uses AND semantics to evaluate the criteria. All match conditions must be true for the match to succeed.

errorcode

Adds a pattern to match error codes.

Syntax

errorcode *pattern*

Parameters

pattern Defines a match pattern that defines the error code set.

Guidelines

The **errorcode** command adds a pattern to match error codes. To determine whether the pattern is a PCRE expression or shell style expression, use the **match-with-pcre** command.

Related Commands

match-with-pcre

Examples

- Enters Matching Rule configuration mode to create the `allErrors` Matching Rule. Adds a pattern to match all error codes.

```
# matching allErrors
Matching configuration mode
# errorcode *
#
```

fullurlmatch (deprecated)

Comments

The **fullurlmatch** command is deprecated. Use the **urlmatch** command.

hostmatch (deprecated)

Comments

The **hostmatch** command is deprecated. Use the **httpmatch** command.

httpmatch

Adds a pattern to match HTTP headers.

Syntax

httpmatch *field pattern*

Parameters

field Specifies an HTTP header as defined in sections 4.5, 5.3, 6.2, and 7.1 of RFC 2616.

pattern Defines a match pattern that defines the value for the HTTP header.

Guidelines

The **httpmatch** command adds a pattern to match HTTP headers. To determine whether the pattern is a PCRE expression or shell style expression, use the **match-with-pcre** command.

Related Commands

match-with-pcre

Examples

- Adds a match pattern to the current Matching Rule. The match succeeds if the `Content-Language` field of a candidate HTTP header contains `da`.

```
# httpmatch Content-Language da
#
```

match-with-pcre

Indicates whether expression uses PCRE or shell-style expression.

Syntax

match-with-pcre {**on** | **off**

Parameters

on Uses PCRE expressions.

off (Default) Uses shell style expressions.

Guidelines

The **match-with-pcre** command indicates whether match patterns use PCRE expression or shell-style expressions. This command applies to patterns defined by the following commands:

- **errorcode**
- **httpmatch**
- **urlmatch**

You can use wildcards to define a match pattern as follows:

`*` The string wildcard matches 0 or more occurrences of any character.

For a PCRE expression, use `.*` rather than `*` to match any number of any characters.

`?` The single character wildcard matches one occurrence of any single character.

`[]` The delimiter pair to bracket a character or numeric range.

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Related Commands

errorcode, **httpmatch**, **urlmatch**

no match

Removes all matches from the rule.

Syntax

no match

Examples

- Clears the `Star` template.

```
# matching Star
Matching configuration mode
# no match
#
```

urlmatch

Adds a pattern to match URLs.

Syntax

urlmatch *pattern*

Parameters

pattern Defines a shell-style match pattern that defines the URL set subject.

Guidelines

The **urlmatch** command adds a pattern to match URLs. To determine whether the pattern is a PCRE expression or shell style expression, use the **match-with-pcre** command.

Related Commands

match-with-pcre

Examples

- Enters Matching Rule configuration mode to create the `Star` Matching Rule. Adds a pattern to match all URLs.

```
# matching Star
Matching configuration mode
# urlmatch *
#
```

- Enters Matching Rule configuration mode to create the `Product` Matching Rule. Adds a pattern to match only candidate URLs that start with `http://www.datapower.com/products/` followed by zero or more characters.

```
# matching Product
Matching configuration mode
# urlmatch http://www.datapower.com/products/*
#
```

---

xpathmatch

Adds an expression to match an XPath.

Syntax

xpathmatch *expression*

Parameters

expression
Specifies an XPath expression that defines the document set subject.

Examples

- Enters Matching Rule configuration mode to create the `xpath` Matching Rule. Adds an expression to match attributes with the name of `destination`.

```
# matching xpath
Matching configuration mode
# xpathmatch @destination
#
```

Chapter 45. Message Count Monitor configuration mode

This chapter provides an alphabetic listing of commands that are available in Message Count Monitor configuration mode.

To enter this configuration mode, use the Global **monitor-count** command.

Message Count Monitor configuration mode enables the creation of a count monitor, a counter-based object that monitors and controls a target message class.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Message Count Monitor configuration mode.

distinct-sources

Specifies the number of distinct IP addresses to track.

Syntax

distinct-sources *count*

Parameters

count Specifies the maximum number of IP addresses to track. The default is 10000.

Guidelines

The **distinct-sources** command specifies the maximum number of IP addresses to track. When too many distinct counts are observed, the addresses not observed in the longest amount of time are discarded.

filter

Specifies a threshold value and control procedure to implement should the threshold value be exceeded.

Syntax

filter *name interval threshold burst-limit control-procedure*

no filter *name*

Parameters

name Specifies the name of the object.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

interval

Specifies the measurement interval in milliseconds.

threshold
Specifies the threshold value. Exceeding this value triggers the specified the control procedure.

burst-limit
Specifies an acceptable traffic burst. The value should be approximately twice the threshold value.

control-procedure
Specifies the name of a control procedure that was created with the **monitor-action** command. This control procedure is triggered when the threshold value is exceeded.

Guidelines

You can add multiple filters to a count monitor.

After completing the configuration of a count monitor, activate the monitor by assigning it to a DataPower service.

Use the **no filter** command to remove a filter from a count monitor.

Related Commands

measure, message-type, show Message Count-filters

Examples

- Defines the `LogSquelch` Message Count monitor. If the `Extranet` message class exceeds 50 client requests (the default counter) per second, implement the `Squelch` control procedure which logs an error and imposes a 2.5 second blackout on the `Extranet` message class.

```
# monitor-count LogSquelch
Message count monitor Configuration mode
# message-type Extranet
# filter Filter1 1000 50 100 Squelch
#
```

header

Specifies the HTTP header to read to determine the client IP address.

Syntax

header *name*

Parameters

name Specifies the name of the HTTP header to read.

Related Commands

source

Examples

- Designates the `X-Client-IP` HTTP Header to read to determine the client IP address.

```
# monitor-count LogSquelch
Message count monitor Configuration mode
# header X-Client-IP
```

measure

Specifies how to increment the counter.

Syntax

measure {**requests** | **responses** | **xpath** | **error**}

Parameters

requests
(Default) Indicates that the receipt of a client request increments the counter.

responses
Indicates that the receipt of a server response increments the counter.

xpath Indicates that a style sheet increments the counter. Use the `dp:increment-integer` extension element in a style sheet. This extension element increments the counter that the count monitor maintains. For example, if the name of the count monitor is `monitor1`, the style sheet must contain the following statement:

```
<dp:increment-integer name="'/monitor-count/monitor1'"/>
```

error Indicates that the receipt of an HTTP error response increments the counter. Processing the Error Rule can increment this counter.

Guidelines

To activate a count monitor, assign it to a DataPower service.

Related Commands

filter, **message-type**

Examples

- Specifies that the `LogSquelch` monitor counter increments by a server responses from the target message class.

```
# monitor-count LogSquelch
Message count monitor Configuration mode
# measure responses
```

message-type

Specifies the target message class for this count monitor.

Syntax

message-type *name*

Parameters

name Specifies the name of the target message class that was configured with the **message-type** (Global) command.

Guidelines

You can assign only a single message class to an count monitor.

After completing the configuration of a count monitor, activate the monitor by assigning it to a DataPower service.

Related Commands

message-matching (Global), **message-type** (Global)

Examples

- Specifies the `Extranet` message class as the target for the `LogSquelch` count monitor.

```
# monitor-count LogSquelch
Message count monitor Configuration mode
# message-type Extranet
```

---

source

Enables IP address-specific monitoring and information gathering.

Syntax

source {**all** | **ip-each** | **ip-from-header**}

all Specifies that IP source address monitoring and information gathering is aggregated for the address range defined by the Message Matching **ip** command.

ip-each
Specifies that IP source address monitoring and information gathering is individualized for all IP addresses (up to 1000) within the range defined by the Message Matching **ip** command.

ip-from-header
Specifies that IP source address monitoring and information gathering is individualized for all IP addresses (up to 1000) within the range defined by the Message Matching **ip** command. IP addresses are read from the Header field.

Guidelines

The **source** command is used only if a traffic-flow definition (Message Matching) contains an **ip** or **ip-exclude** command.

In the absence of a **source** command, source address monitoring and information gathering (if defined) is aggregated for the entire address range defined by the **ip** or **ip-exclude** commands.

Related Commands

ip (Message Matching), **ip-exclude** (Message Matching)

Examples

- Enables individualized information gathering for each IP address specified within the traffic-flow definition.

```
# monitor-count Loose
Message count monitor Configuration mode
# source ip-each
#
```

Chapter 46. Message Duration Monitor configuration mode

This chapter provides an alphabetic listing of commands that are available in Message Duration Monitor configuration mode.

To enter this configuration mode, use the Global **monitor-duration** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Message Duration Monitor configuration mode.

Message Duration Monitor configuration mode enables the creation of a duration monitor, an object that monitors certain transactions associated with a target message class.

filter

Specifies both a threshold value and a control procedure to implement if the threshold value is exceeded.

Syntax

filter *name* **average** *threshold control-procedure*

no filter *name*

Parameters

name Specifies the name of the filter.

The name can contain a maximum of 32 characters. For restrictions, refer to "Object name conventions" on page xxvi.

average
Indicates a required keyword.

threshold
Specifies the threshold processing interval in milliseconds. Exceeding this value triggers the specified control procedure.

control-procedure
Specifies the name of a control procedure that was created with the Global **monitor-action** command. This control procedure is triggered if the threshold value is exceeded.

Guidelines

You can add multiple filters to a duration monitor.

After completing the configuration of a duration message monitor, activate the monitor by assigning it to a DataPower service.

Use the **no filter** command to remove a filter from an incremental message monitor.

Related Commands

monitor-action (Global), **show message-durations**, **show message-duration-filters**

Examples

- Defines the `RateLimit1` duration message monitor. If the average server processing time of the `Extranet` message class exceeds 500 milliseconds, implement the `Yell` control procedure.

```
# monitor-count RateLimit1
Message duration monitor Configuration mode
# message-type Extranet
# measure server
# filter Filter3 average 500 Yell
#
```

- Deletes the filter named `Filter3` from the `RateLimit1` duration message monitor.

```
# monitor-count RateLimit1
Message duration monitor Configuration mode
# no filter Filter3
#
```

measure

Specifies the transaction type of interest.

Syntax

measure {**messages** | **requests** | **responses** | **server**}

Parameters

messages
Specifies the time interval between the receipt of a client request and the transmission of the associated server response.

requests
Specifies the time spent by the appliance in processing a client request, that is the interval between the receipt of a client request and its transmission to the target server.

responses
Specifies the time spent by the appliance in processing a server response, that is the interval between the receipt of a server response and its transmission to the target client.

server Specifies the server processing time, that is the interval between the transmission of a client request to the server and the receipt of the associated server response.

Guidelines

Each transaction type represents a portion of the client-to-server roundtrip. The **requests** and **responses** types are internal to the appliance. The **requests** type measure the time required to perform the configured actions on client requests. The **responses** type measure the time required to perform similar services on server responses.

The **server** and **messages** types deal with external processing, specifically the processing performed by the web or application server. The **server** type measures the actual server processing time. The **messages** type approximates the sum of **requests**, **server**, and **responses** types.

After completing the configuration of a duration monitor, activate the monitor by assigning it to a DataPower service.

Examples

- Specifies that the `RateLimit1` duration monitor gathers data on message transactions.

```
# monitor-duration RateLimit1
Message duration monitor Configuration mode
# measure messages
```

---

message-type

Specifies the target message class.

Syntax

message-type *name*

Parameters

name Specifies the name of the target message class that was configured with the **message-type** (Global) command.

Guidelines

You can assign only a single message class to an incremental monitor.

After completing the configuration of a duration monitor, activate the monitor by assigning it to a DataPower service.

Related Commands

message-matching (Global), **message-type** (Global)

Examples

- Specifies the `Extranet` message class as the target for the `RateLimit1` duration monitor.

```
# monitor-duration RateLimit1
Message duration monitor Configuration mode
# message-type Extranet
```

Chapter 47. Message Filter Action configuration mode

This chapter provides an alphabetic listing of commands that are available in Message Filter Action configuration mode.

To enter this configuration mode, use either the Global **monitor-action** command.

While in this mode, define a filter action. This action enables the configuration of control procedures. A filter action is an action or set of actions to implement when a monitored message class exceeds a configured threshold value.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Message Filter Action configuration mode.

block-interval

Specifies the time period during which a message class is denied service as a consequence of exceeding a configured threshold.

Syntax

block-interval *milliseconds*

Parameters

milliseconds
Specifies the blackout in milliseconds.

Guidelines

This command is meaningful only if the **type** command is set to **block**, and is otherwise not used.

This command is required when the **type** command is set to **block**.

Related Commands

log-priority, **type**

Examples

- Enters Message Filter Action configuration mode to create the `Squelch` control procedure.

```
# monitor-action Squelch
Message filter action Configuration mode
#
```

- Specifies a block interval of 2½ seconds.

```
# type block
# block-interval 2500
#
```

log-priority

Enables the generation of a log entry when a control procedure is triggered.

Syntax

log-priority *priority*

Parameters

priority

Identifies the event priority. The priority indicates that all events that are greater than or equal to this value are logged. Events use the following priority in descending order:

- **emerg** (Emergency)
- **alert** (Alert)
- **critic** (Critical)
- **error** (Error)
- **warn** (Warning)
- **notice** (Notice)
- **info** (Information)
- **debug** (Debug)

Guidelines

This command is required when the **type** command is set to **notify**; otherwise, it is optional.

Related Commands

block-interval, **type**

Examples

- Enters Message Filter Action configuration mode to create the `PaperTrail` control procedure.

```
# monitor-action PaperTrail
Message filter action Configuration mode
#
```

- Defines the `PaperTrail` control procedure as being of the `notify` type. Log a warning and take no further action.

```
# type notify
# log-priority warning
#
```

type

Specifies the behavior of a control procedure when a monitored message class exceeds a configured threshold value.

Syntax

type {**block** | **notify** | **reject**}

Parameters

block Temporarily denies service to a message class and optionally adds a log entry, when a message class exceeds a configured threshold.

notify Adds a log entry when a message class exceeds a configured threshold.

reject Drops all over-threshold traffic originating from a message class, and optionally adds a log entry, when a message class exceeds the configured threshold.

Guidelines

Conditional tests that trigger the execution of control procedures are defined by the **monitor-count** and **monitor-duration** commands.

Related Commands

block-interval, **log-priority**, **message-count**, **message-duration**

Examples

- Enters Message Filter Action configuration mode to create the `Squelch` control procedure. Defines the `Squelch` control procedure as being of the `block` type. Logs an error and deny service to the traffic stream defined by the `TFDef1` traffic flow definition for 2½ seconds.

```
# monitor-action Squelch
Message filter action Configuration mode
# type block
# block-interval 2500
# block-messages TFDef1
# log-priority error
# exit
#
```

- Enters Message Filter Action configuration mode to create the `Restrain` control procedure. Defines the `Restrain` control procedure as being of the `reject` type. Logs an error and reject (drop) the over-threshold message.

```
# monitor-action Restrain
Message filter action Configuration mode
# type reject
# log-priority error
# exit
#
```

- Enters Message Filter Action configuration mode to create the `LogIt` control procedure. Defines the `LogIt` control procedure as being of the `notify` type. Logs a warning and take no further action.

```
# monitor-action LogIt
Message filter action Configuration mode
# type notify
# log-priority warning
#
```

Chapter 48. Message Matching configuration mode

This chapter provides an alphabetic listing of commands that are available in Message Matching configuration mode.

To enter this configuration mode, use the Global **message-matching** command. While in Message Matching configuration mode, you specify traffic-flow definitions. A traffic-flow definition describes a traffic stream that is subject to administrative monitoring and control. A candidate message is considered part of a traffic-stream only if the candidate matches all configured properties of a traffic-flow definition.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Message Matching configuration mode.

http-header

Specifies an HTTP header field and associated header field value to be included in the traffic-flow definition.

Syntax

http-header *field pattern*

no http-header *field*

Parameters

field Identifies an HTTP header field as defined in sections 4.5, 5.3, 6.2, and 7.1 of RFC 2616.

pattern Defines a shell-style match pattern that defines the contents of the HTTP header field. You can use wildcard characters when identifying the HTTP header field contents.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Guidelines

A traffic-flow definition may contain multiple **http-header** commands.

In the absence of an **http-header** command, HTTP header contents are not considered when evaluating a candidate message against a traffic-flow definition.

Use the **no http-header** command to remove a HTTP header field match from a traffic-flow definition.

Related Commands

http-header-exclude

Examples

- Creates the `TFDef1` traffic-flow definition. HTTP traffic that contains a `From` request header field with the string `@businessPartner.com` is defined as part of the target traffic flow.

```
# message-matching TFDef1
Message matching configuration mode
# http-header From *@businessParter.com
#
```

- Removes HTTP traffic that contains a `From` request header field from the `TFDef1` traffic-flow definition.

```
# message-matching TFDef1
Message matching configuration mode
# no http-header From
#
```

http-header-exclude

Specifies an HTTP header field and associated header field value to be excluded from the traffic-flow definition.

Syntax

http-header-exclude *field pattern*

no http-header-exclude *field*

Parameters

field Identifies an HTTP header field as defined in sections 4.5, 5.3, 6.2, and 7.1 of RFC 2616.

pattern Defines a shell-style match pattern that defines the contents of the HTTP header field.

Guidelines

A traffic flow definition may contain multiple **http-header-exclude** commands.

In the absence of an **http-header** or **http-header-exclude** command, HTTP header contents are not considered when evaluating an individual message against a traffic-flow definition.

Use the **no http-header-exclude** command to remove a HTTP header field exclusion from a traffic-flow definition.

Related Commands

http-header

Examples

- Creates the TFDef1 traffic-flow definition. HTTP traffic that contains a From request header field with the string @businessPartner.com is excluded from the target traffic flow.

```
# message-matching TFDef1
Message matching configuration mode
# http-header-exclude From *businessParter.com
#
```

- Removes HTTP traffic that contains a From request header field from the TFDef1 traffic-flow definition.

```
# message-matching TFDef1
Message matching configuration mode
# no http-header-exclude from
#
```

ip

Specifies a range of IP source addresses to include in the traffic-flow definition.

Syntax

ip *address/prefix-length*

Parameters

address Specifies a dotted decimal IP address that, with the prefix length, defines a range of included IP addresses.

prefix-length
Defines a range of included IP addresses. Use an integer in the range of 1 through 32.

Guidelines

A traffic flow definition can contain only a single **ip** command.

In the absence of an **ip** or **ip-exclude** command, source address is not considered when evaluating an individual message against a traffic-flow definition.

Related Commands

ip-exclude

Examples

- Creates the TFDef1 traffic-flow definition. IP traffic that originates from the specified address range is included in the target traffic flow.

```
# message-matching TFDef1
Message matching configuration mode
# ip 10.10.1.0/24
#
```

ip-exclude

Specifies a range of IP source addresses to exclude from the traffic-flow definition.

Syntax

ip-exclude *address/prefix-length*

Parameters

address Specifies a dotted decimal IP address that, with the prefix length, defines a range of excluded IP addresses.

prefix-length
Defines a range of excluded IP addresses. Use an integer in the range of 1 through 32.

Guidelines

A traffic flow definition can contain a single **ip-exclude** command.

In the absence of an **ip** or **ip-exclude** command, source address is not considered when evaluating an individual message against a traffic-flow definition.

Related Commands

ip

Examples

- Creates the `TFDef1` traffic-flow definition. IP traffic that originates from the specified host address is excluded from the target traffic flow.

```
# message-matching TFDef1
Message matching configuration mode
# ip-exclude 10.10.1.0/24
#
```

---

method

Specifies an HTTP method type to be included in the traffic-flow definition.

Syntax

method {**connect** | **delete** | **get** | **head** | **options** | **post** | **put** | **trace**}

Parameters

connect, **delete**, **get**, **head**, **options**, **post**, **put**, and **trace**
Identifies keywords for the HTTP methods that are defined in RFC 2616.

Guidelines

A traffic-flow definition may contain a single **method** command.

In the absence of a **method** command, HTTP method type is not considered when evaluating an individual message against a traffic-flow definition.

Examples

- Creates the `TFDef1` traffic-flow definition. Identifies HTTP posts that originate from the specified IP address range are part of the target traffic flow.

```
# message-matching TFDef1
Message matching configuration mode
# ip 10.10.1.0/24
# method post
#
```

request-url

Specifies a requested URL set to include in the traffic-flow definition.

Syntax

request-url *pattern*

Parameters

pattern Defines a shell-style match pattern that defines the requested URL. You can use wildcard characters when identifying the target URL.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Guidelines

A traffic flow definition can contain a single **request-url** command.

In the absence of a **request-url** command, the target URL is not considered when evaluating an individual message against a traffic-flow definition.

Examples

- Creates the `TFDef1` traffic-flow definition. Identifies all requests for XML documents in the specified directory that originate from the specified IP address range are part of the target traffic flow.

```
# message-matching TFDef1
Message matching configuration mode
# ip 10.10.1.0/24
# request-url http://www.datapower.com/XS40/*.xml
#
```

Chapter 49. Message Type configuration mode

This chapter provides an alphabetic listing of commands that are available in Message Type configuration mode.

To enter this configuration mode, use the Global **message-type** command.

While in this mode, create a message class. A message class is a list of one or more traffic-flow definitions. Message classes identify traffic streams that are subject to administrative control procedures.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Message Type configuration mode.

message-matching

Adds a traffic-flow definition to a message class.

Syntax

message-matching *name*

no message-matching *name*

Parameters

name Specifies the name of a traffic-flow definition previously created with the **message-matching** (Global) command.

Guidelines

You can add multiple traffic-flow definitions to a message class.

Use the **no message-matching** command to delete a traffic-flow definition from a message class.

Related Commands

message-matching (Global)

Examples

- Enters Message Type configuration mode and creates the `Extranet` message class. Adds the `TFDef2` and `TFDef2` traffic-flow definitions to the `Extranet` message class.

```
# message-type Extranet
Message type configuration mode
# message-matching TFDef1
# message-matching TFDef2
#
```

- Deletes the `TFDef2` traffic-flow definition from the `Extranet` message class.

```
# message-type Extranet
Message type configuration mode
# no message-matching TFDef2
#
```

Chapter 50. MQ Front Side Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in MQ Front Side Handler configuration mode.

To enter this configuration mode, use the Global **source-mq** command. While in these modes, you can configure properties that define a client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

ccsi

Specifies the default output encoding for documents processing.

Syntax

ccsi *id*

Parameters

id Specifies the output encoding. Refer to the following Web site for the list of coded character set identifiers:

http://www.ibm.com/software/globalization/ccsid/ccsid_registered.jsp

Guidelines

Meaningful only when the configuration of the MQ Queue Manager object defines the value of the **convert** command to **on**, the default.

With this property **on**, the remote MQ Queue Manager converts output data to `ISO-8859-1` (`latin-1`).

To convert the output data to an encoding other than `latin-1`, use this command to identify that CCSI.

Related Commands

convert (MQ Queue Manager)

Examples

- Enter MQ Front Side Handler configuration mode to create the `MQ-FSH` handler. Specifies UTF-8 as the default output encoding.

```
# source-mq MQ-FSH
MQ Front Side Handler configuration mode
# ccsi 1209
#
```

concurrent-connections

Sets the number of concurrent connections.

Syntax

concurrent-connections *limit*

Parameters

limit Specifies the number of concurrent MQ connections to allocate. The minimum is 1. The default is 1.

content-type-header

Specifies the header to extract.

Syntax

content-type-header {**None** | **MQRFH** | **MQRFH2**}

Parameters

None (Default) No header

MQRFH
The Content-Type of the MQRFH header

MQRFH2
The Content-Type of the MQRFH2 header

Related Commands

content-type-xpath

content-type-xpath

Specifies the XPath expression to extract the header.

Syntax

content-type-xpath *XPath*

Parameters

XPath Specifies the XPath expression to extract.

Related Commands

content-type-header

exclude-headers

Identifies the MQ headers to remove from the message.

Syntax

exclude-headers *value*[+*value*]

Parameters

value Identifies the headers to remove. When specifying multiple headers to remove, use a plus sign (+) as the separator. The following values are valid:

`MQCIH` CICS Bridge Header

MQDLH Dead Letter Header

MQIIH IMS Information Header

MQRFH Rules and Formatting Header

MQRFH2
Rules and Formatting Header

MQWIH Working Information Header

Examples

- Identifies that the MQRFH header should be removed from the message before processing.

```
# exclude-headers MQRFH
#
```

- Identifies that the MQRFH and MQRFH2 headers should be removed from the message before processing.

```
# exclude-headers MQRFH+MQRFH2
#
```

---

get-message-options

Specifies the **MQGET** options.

Syntax

get-message-options *cumulative-value*

Parameters

cumulative-value
Specifies the cumulative value of the **MQGET** options that are applicable to an MQ message in decimal or hexadecimal format. The value is passed directly to the MQ API. The default is 4097 (decimal value for the MQGMO_WAIT and the MQGMO_SYNCPOINT_IF_PERSISTENT options).

Refer to the topic "MQGMO_* (Get Message Options)" in the WebSphere MQ Information Center for details.

http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp

Guidelines

When a message is too large for a queue, an attempt to put the message on the queue fails. Segmentation is a technique that allows the queue manager or application to split the message into smaller pieces, and place each segment on a queue as a separate physical message. The application that retrieves the message can either handle the multiple segments one-by-one, or request the queue manager to reassemble the segments into a single message that is returned by the **MQGET** call. The reassembly request is achieved by specifying the MQGMO_COMPLETE_MSG option (65536) on the **MQGET** call, and by providing a buffer large enough to accommodate the entire message.

Note: Ensure that the associated queue manager supports the MQGMO_COMPLETE_MSG option. On z/OS, MQ queue managers do not support segmentation. On other operating systems, MQ queue managers might not be configured to support segmentation.

get-queue

Identifies the GET queue to retrieve messages from an MQ Queue Manager.

Syntax

get-queue *name*

Parameters

name Specifies the name of a GET queue.

Guidelines

The **get-queue** command specifies the GET queue on the MQ Queue Manager.

Examples

- Identifies the GET queue as `GETSVCQ`.

```
# get-queue GETSVCQ
#
```

polling-interval

Specifies the wait interval for an **MQGET** operation.

Syntax

polling-interval *interval*

Parameters

interval

Specifies the number of seconds to wait for an **MQGET** operation. The minimum is 1. The default is 30.

Guidelines

The **polling-interval** command specifies the number of seconds before an **MQGET** operation returns if no messages are available. The next attempt will be performed immediately. Setting a low value will help to detect quickly network connectivity issues and queue manager power shutdown. At the same time, a low value will increase network traffic.

put-queue

Identifies the PUT queue to transfer messages to an MQ Queue Manager.

Syntax

put-queue *name*

Parameters

name Specifies the name of a PUT queue.

Guidelines

The **put-queue** command specifies the PUT queue on the MQ Queue Manager.

Examples

- Identifies the PUT queue as PUTSVCQ.

```
# put-queue PUTSVCQ
#
```

queue-manager

Identifies the Queue Manager with which to communicate.

Syntax

queue-manager *name*

Parameters

name Specifies the name of a Queue Manager object in the current application domain.

Guidelines

This Queue Manager object must already exist in the current application domain. Use the global **mq-qm** command to create a new queue manager.

Related Commands

mq-qm (Global)

Examples

- Sets the Queue Manager to the SVCQM Queue Manager object.

```
# queue-manager SVCQM
# exit
```

retrieve-backout-setting

Determines whether to retrieve backout settings from the MQ server.

Syntax

retrieve-backout-setting {**on** | **off**}

Parameters

on Retrieves backout settings from the MQ server.

off (Default) Does not retrieve backout settings from the MQ server.

Guidelines

The **retrieve-backout-setting** command determines whether to retrieve backout settings from the MQ server and compares to the backout values set in the MQ Queue Manager object. The backup settings for an MQ Queue Manager object are set with the following commands:

- **backout-queue**

- **backout-threshold**

On retry, the appliance uses the higher priority backout settings from the MQ server. If the MQ server does not contain backout settings, the appliance uses existing backout properties, either empty or populated, that are defined for the MQ Queue Manager object. If there are no backout settings, the backout functionality is disabled.

Related Commands

backout-queue (MQ Queue Manager), **backout-threshold** (MQ Queue Manager)

Examples

- Enables and subsequently disables the retrieval of backup settings from the MQ server.

```
# retrieve-backout-setting on
:
# retrieve-backout-setting off
#
```

Chapter 51. MQ Queue Manager configuration mode

This chapter provides an alphabetic list of commands that are available in MQ Queue Manager configuration mode. Most, but not all, of the commands in "Common commands" on page 2 and in Chapter 129, "Monitoring commands," on page 1053 are available in this configuration mode.

To enter this configuration mode, use the Global **mq-qm** command.

alternate-user

Determines whether to use `MQOD.AlternateUserId`.

Syntax

alternate-user {**on** | **off**}

Parameters

on (Default) Enables the use of `MQOD.AlternateUserId`.

off Disables the use of `MQOD.AlternateUserId`.

Guidelines

The **alternate-user** command determines whether to use `MQOD.AlternateUserId` during an open **MQOPEN** operation or put **MQPUT** operation.

- When enabled, the MQ operation uses the value for `MQOD.AlternateUserId`.
- When disabled, the MQ operation uses the value for `MQMD.UserIdentifier`.

Alternatively, use the **no alternate-user** command the use of `MQOD.AlternateUserId`.

Related Commands

username

automatic-backout

Determines whether to enable the backout count for the backout queue.

Syntax

automatic-backout {**on** | **off**}

Parameters

on Enables the backout count for the backup queue.

off (Default) Disables the backout count for the backup queue.

Guidelines

Use the **automatic-backout** command determines whether to enable the backout count for the backup queue. When enabled:

- Use the **backout-threshold** command to define the maximum number of times to attempt an **MQGET** operation before transferring the message to the backout queue.
- Use **backout-queue** command to define the backout queue for *poison* messages. A poison message is any message that the receiving application does not know how to process. This command is meaningful only if **units-of-work** is 1.

Usually an application rolls back the GET of this message, which leaves the message on the input queue. However, the backout count (`MQMD.BackoutCount`) increments. `MQMD.BackoutCount` counts the number of times that the message was returned by the **MQGET** call as part of the unit of work and subsequently backed out. When the backout count exceeds the backout threshold, the queue manager moves the message to the backout queue.

Alternatively, use the **no automatic-backout** command to enable automatic backout.

Related Commands

backout-queue, **backout-threshold**, **units-of-work**

---

auto-retry

Determines whether to enable the automatic retry procedure after a connection failure.

Syntax

auto-retry {**on** | **off**}

Parameters

on (Default) Enables the automatic retry procedure.

off Disables the automatic retry procedure.

Guidelines

The **auto-retry** command determines whether to enable the automatic retry procedure after a connection failure. This command controls whether the channel retries to establish connection. This command does not affect **MQGET** calls or **MQPUT** calls. These calls occur over an established connection.

If the connection to a queue manager fails for any reason other than MQ error code 2009 (connection broken), the DataPower appliance stops the transaction, starts error handling (such as run a configured Error Rule), and start the automatic retry procedure (if enabled). For broken connections, the DataPower appliance attempts to establish a connection at the interval that the **retry-interval** command defines.

When enabled:

- Use the **retry-interval** command to indicate how often to attempt to establish a connection.
- Use the **reporting-interval** command to indicate how often to log a failed connection attempt at the `error` priority. The log shows the other failed connection attempts at the `debug` priority.

Alternatively, use the **no auto-retry** command to disable the automatic retry procedure.

Related Commands

reporting-interval, **retry-interval**

backout-queue

Identifies the backout queue.

Syntax

backout-queue *name*

Parameters

name Identifies the name of the queue to contain messages that exceed the backout threshold.

Guidelines

Use the **backout-queue** command to specify the backout queue. The command is relevant when **units-of-work** is 1 and **automatic-backout** is **on**. The backout queue contains messages that cannot be processed or delivered. This queue is typically the dead-letter queue (`SYSTEM.DEAD.LETTER.QUEUE`).

Related Commands

automatic-backout, **units-of-work**

Examples

- Identifies `SYSTEM.DEAD.LETTER.QUEUE` as the backout queue.

```
# backout-queue SYSTEM.DEAD.LETTER.QUEUE
#
```

backout-threshold

Specifies the maximum number of reprocessing attempts per message.

Syntax

backout-threshold *attempts*

Parameters

attempts

Specifies the maximum number of reprocessing attempts per message. The default is 1.

Guidelines

Use the **backout-threshold** command to specify the backout count. This command is relevant when **units-of-work** is 1 and **automatic backout** is **on**, The backout count is the maximum number of reprocessing attempts that are allowed per message.

Logically, the backout count starts at 0, which accounts for the initial reception of a message. Consequently, the default value 1 specifies two attempts: the initial processing attempt and a single attempt to reprocess the message.

Related Commands

automatic-backout, **units-of-work**

---

cache-timeout

Specifies the maximum period to retain a dynamic connection in the connection cache.

Syntax

cache-timeout *seconds*

Parameters

seconds

Specifies the number of seconds to retain a dynamic connection in the connection cache. The minimal value is 1. The default is an empty string. This value indicates that connections in the cache do not time out.

Guidelines

The **cache-timeout** command specifies the number of seconds that the DataPower appliance retains (keeps alive) a dynamic connection in the connection cache. Specify a value that is greater than the negotiated heartbeat interval but less than the keep alive interval.

- The negotiated heartbeat interval is between the DataPower appliance and the backend MQ server. Use the **heartbeat** command to define the starting value for the negotiation.
- The keep alive (timeout) interval is on the backend MQ server. The `KAINT` attribute on the MQ server defines the timeout value for a channel.

  Not all channels have a defined, explicit keep alive interval on the MQ server. Some queue managers use an automatic timeout setting (the `KAINT` attribute set to `AUTO`). In these cases, the keep alive interval is the negotiated heartbeat interval plus 60 seconds.

When an inactive connection reaches this threshold, the DataPower appliance removes that dynamic connection from the cache. When the cache no longer contains dynamic connections, the DataPower appliance deletes the dynamic queue manager. Without a dynamic queue manager, there is no connection with the backend MQ server.

Note: The timeout value for the **cache-timeout** command is the only way to configure a timeout value from the DataPower appliance to the backend MQ server. No other configuration setting on the DataPower appliance can time out an MQ connection.

Related Commands

heartbeat, **initial-connections**, **total-connection-limit**

ccsid

Specifies the coded character set identifier to present to the queue manager.

Syntax

ccsid *identifier*

Parameters

identifier

Specifies the coded character set identifier. The default is 819.

Refer to the following Web site for the list of coded character set identifiers:

http://www.ibm.com/software/globalization/ccsid/ccsid_registered.jsp

Guidelines

The **ccsid** command specifies the coded character set identifier (CCSID) that the DataPower appliance presents to the MQ queue manager when attempting to make a connection. The CCSID that is used during the connection does not affect the application data in the message.

This property is equivalent to setting the `MQCCSID` variable for an MQ client. This property is essential when attempting to make a connection to the MQ queue manager server that uses a double-byte character set (DBCS).

Related Commands

ccsi (MQ Front Side Handler), **ccsi** (MQ Host), **convert**

channel-name

Identifies a multiplexed logical link to the remote MQ server.

Syntax

channel-name *name*

Parameters

name Specifies the name of the channel. The default is `SYSTEM.DEF.SVRCONN`.

Guidelines

The **channel-name** command identifies the channel (multiplexed logical link) to the remote MQ server. If you do not define a different channel, the configuration uses the default value.

Related Commands

hostname, **message-size**, **queue-manager**, **units-of-work**

convert

Determines whether to enable CCSI conversion.

Syntax

convert {**on** | **off**}

Parameters

on (Default) Enables CCSI conversion.

off Disables CCSI conversion.

Guidelines

The **convert** command determines whether the remote queue manager performs CCSI conversion. The queue manager, which the **queue-manager** command identifies, can convert input messages to a different CCSI than the one in the original message. The remote queue manager converts the message, not the DataPower appliance.

When enabled, use the **ccsid** command to specify which CCSI the remote queue manager uses for CCSI conversion of input messages. For output messages, use the **ccsi** command on the MQ Front Side Handler or on the MQ Host.

Alternatively, use the **no convert** command to disable CCSI conversion.

Related Commands

ccsi (MQ Front Side Handler), **ccsi** (MQ Host), **ccsid**, **queue-manager**

---

heartbeat

Specifies the value to use when sending heartbeat flows on the channel when waiting for a message on a queue.

Syntax

heartbeat *seconds*

Parameters

seconds

Specifies the approximate time, in seconds, between heartbeat flows on a channel when waiting for a message on a queue. Use an integer in the range of 0 through 999999. If 0, there will be no heartbeat flow exchange. The default is 300.

Guidelines

This value does not set the heartbeat on the channel, rather it is used to negotiate the heartbeat value with the channel. The greater of the two values is used.

Related Commands

channel-name, **message-size**, **queue-manager**, **units-of-work**

Examples

- Identifies an MQ channel heartbeat of 375.

```
# heartbeat 375
#
```

hostname

Identifies a remote MQ server.

Syntax

hostname {*address* | *hostname*}

Parameters

address Identifies the MQ server by dotted decimal IP address.

hostname
Identifies the MQ server by host name.

Guidelines

You must identify an MQ server when configuring a queue manager.

Related Commands

channel-name, **message-size**, **queue-manager**, **units-of-work**

Examples

- Identifies an MQ server at 10.10.13.124.

```
# hostname 10.10.14.124
#
```

- Identifies an MQ server at `WAS-2e.us.ibm.com`.

```
# hostname WAS-2e.us.ibm.com
#
```

initial-connections

Specifies the number of connections to allocate.

Syntax

initial-connections *count*

Parameters

count Specifies the number connections to allocate when the queue manager starts. Use an integer in the range of 0 through 5000. The default is 1.

Related Commands

cache-timeout, **total-connection-limit**

local-address

Specifies a local interface for outbound connections.

Syntax

local-address *interface*

Parameters

interface
Specifies an interface or interfaces for outbound connections.

Guidelines

Forces outbound connections to use a specific local interface or port.

Related Commands

total-connection-limit

Examples

- Uses any port on the specified interface to establish an outbound connection.

```
# local-address 192.168.33.33
#
```

- Uses port 56400 on the specified interface to establish an outbound connection.

```
# local-address 192.168.33.33(56400)
#
```

- Uses any port within the range 56400 through 58000 on the specified interface to establish an outbound connection. If a range of ports is specified, the range must be greater that the value specified by the **total-connection-limit** command.

```
# local-address 192.168.33.33(56400, 58000)
#
```

- Uses the specified port on any configured interface to establish an outbound connection. If a range of ports is specified, the range must be greater that the value specified by the **total-connection-limit** command.

```
# local-address (56400)
#
```

- Uses any port within the range 56400 through 58000 on any configured interface to establish an outbound connection. If a range of ports is specified, the range must be greater that the value specified by the **total-connection-limit** command.

```
# local-address (56400, 58000)
#
```

---

maximum-message-size

Specifies the queue-manager-specific maximum message size.

Syntax

message-size *bytes*

Parameters

bytes Specifies the maximum message size in bytes.

Guidelines

Optionally used when configuring a queue manager, **message-size** specifies the maximum message size in bytes for this queue manager. Use an integer in the range of 1024 through 1073741824. The default is 1048576.

Related Commands

channel-name, **hostname**, **queue-manager**, **units-of-work**

queue-manager

Identifies a queue manager.

Syntax

queue-manager *name*

Parameters

name Specifies the name of a queue manager on **hostname**.

Guidelines

Optionally used when configuring a queue manager, the **queue-manager** command identifies a specific queue manager resident on the MQ server identified by the **hostname** command.

Related Commands

channel-name, **hostname**, **message-size**, **units-of-work**

reporting-interval

Minimizes against duplicated log entries

Syntax

reporting-interval *seconds*

Parameters

seconds

Specifies the number of seconds to wait before writing identical log messages to an MQ logging target. The default is 1.

Guidelines

Optionally use this command to filter the generation of error messages to MQ logging target.

More frequent identical messages will be added to the log with a `debug` priority.

retry-interval

Specifies the number of seconds to wait before attempting to reestablish a failed connection.

Syntax

retry-interval *seconds*

Parameters

seconds

Specifies number of seconds to wait before attempting to reestablish a failed connection. The default is 1.

Related Commands

auto-retry

---

ssl

Assigns the SSL Proxy Profile to establish the SSL session.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ssl** command assigns an existing SSL Proxy Profile to control the behavior of SSL connections to the remotely located MQ Queue Manager.

You can specify the SSL behavior with either the **ssl** command or with a combination of the **ssl-key** and **ssl-cipher** commands. To use the **ssl-key** and **ssl-cipher** commands, the SSL artifacts where created with IBM Global Security (GSKit) and the required key database file and stash file were uploaded to the appliance.

Note: To integrate with MQ for z/OS, use an SSL Proxy Profile.

Table 10 maps the relationship between the **ssl-cipher** command and the setting that is required on the Crypto Profile of the SSL Proxy Profile that is assigned to the MQ Queue Manager. Use this information when configuring the SSL Proxy Profile to communicate with the MQ Queue Manager.

Table 10. Mapping of the SSL Cipher setting the Crypto Profile settings

Value for the MQ Queue Manager ssl-cipher command	Parameter values on the Crypto profile command
NULL_MD5	**ciphers**: NULL-MD5 **options**: OpenSSL-default+Disable-TLSv1
NULL_SHA	**ciphers**: NULL-SHA **options**: OpenSSL-default+Disable-TLSv1
RC4_MD5_EXPORT	**ciphers**: EXP-RC4-MD5 **options**: OpenSSL-default+Disable-TLSv1
RC4_MD5_US	**ciphers**: RC4-MD5 **options**: OpenSSL-default+Disable-TLSv1
RC4_SHA_US	**ciphers**: RC4-SHA **options**: OpenSSL-default+Disable-TLSv1
RC2_MD5_EXPORT	**ciphers**: EXP-RC2-CBC-MD5 **options**: OpenSSL-default+Disable-TLSv1
DES_SHA_EXPORT	**ciphers**: DES-CBC-SHA **options**: OpenSSL-default+Disable-TLSv1
RC4_56_SHA_EXPORT1024	**ciphers**: EX1024-RC4-SHA **options**: OpenSSL-default+Disable-TLSv1
DES_SHA_EXPORT1024	**ciphers**: EXP1024-DES-CBC-SHA **options**: OpenSSL-default+Disable-TLSv1
TRIPLE_DES_SHA_US	**ciphers**: DES-CBC3-SHA **options**: OpenSSL-default+Disable-TLSv1

Table 10. Mapping of the SSL Cipher setting the Crypto Profile settings (continued)

Value for the MQ Queue Manager ssl-cipher command	Parameter values on the Crypto profile command
TLS_RSA_WITH_AES_128_CBC_SHA	**ciphers**: AES128-SHA **options**: OpenSSL-default
TLS_RSA_WITH_AES_256_CBC_SHA	**ciphers**: AES256-SHA **options**: OpenSSL-default
AES_SHA_US	**ciphers**: AES128-SHA **options**: OpenSSL-default

Related Commands

ssl-cipher, ssl-key

Examples

- Assigns the `MQQM_AES_SHA_US` SSL Proxy Profile to the MQ Queue Manager

```
# ssl MQQM_AES_SHA_US
#
```

ssl-cipher

Specifies the cipher suite available to support SSL operations.

Syntax

ssl-cipher *ciphers*

Parameters

ciphers If SSL is enabled, specifies the cipher suite to make available on the MQ Queue Manager for SSL operations.

- `none`
- `NULL_MD5`
- `NULL_SHA`
- `RC4_MD5_EXPORT`
- `RC4_MD5_US`
- `RC4_SHA_US`
- `RC2_MD5_EXPORT`
- `DES_SHA_EXPORT`
- `RC4_56_SHA_EXPORT1024`
- `DES_SHA_EXPORT1024`
- `TRIPLE_DES_SHA_US`
- `TLS_RSA_WITH_AES_128_CBC_SHA`
- `TLS_RSA_WITH_AES_256_CBC_SHA`
- `AES_SHA_US`

Guidelines

Format of the cipher descriptor is defined by the MQ product. Refer to the latest MQ documents for supported values.

Required if SSL is enabled for communications with the current MQ Queue Manager (and otherwise not used), specifies the cipher suite available to support SSL operations.

Related Commands

ssl-key

Examples

- Identifies supported cryptographic algorithms.

```
# ssl-cipher TRIPLE_DES_SHA_US
#
```

ssl-key

Specifies the location of the MQ key database file.

Syntax

ssl-key *url*

Parameters

url Specifies the location of the MQ key database file on the DataPower appliance.

Guidelines

The **ssl-key** specifies the location of the key database file in which keys and certificates are stored. Each key database file has an associated stash file. The stash file holds encrypted passwords that are used to allow programmatic access to the key database. The stash file must reside in the same directory as the key database file, have the same file stem as the key database file, and must end with the suffix .sth.

For example, if the key database file is `cert:///MQkeys.pem` or `cert:///MQkeys.kdb`, the password stash file must be `cert:///MQkeys.sth`.

You can specify the SSL behavior with either the **ssl** command or with a combination of the **ssl-key** and **ssl-cipher** commands. To use the **ssl-key** and **ssl-cipher** commands, the SSL artifacts where created with IBM Global Security (GSKit) and the required key database file and stash file were uploaded to the appliance.

Related Commands

ssl, **ssl-cipher**

Examples

- Identifies `cert:///MQKeys.pem` as the MQ key database file.

```
# ssl-key cert:///MQKeys.pem
#
```

total-connection-limit

Specifies the maximum number of open connections to support.

Syntax

total-connection-limit *connections*

Parameters

connections
Sets the maximum open connections limit. Use a value in the range of 1 through 5000. The default is 250.

---

units-of-work

Specifies an MQ synchpoint.

Syntax

units-of-work *number*

Parameters

number
Specifies the number of MQ messages that comprise a unit-of-work.

Guidelines

Certain operations (for example, database update) may require that all transactions succeed or fail as a group. A group of contiguous, related messages is referred to as a transaction group or as a unit-of-work.

Optionally used when configuring a queue manager, units-of-work identifies a unit-of-work, and enables reliable processing for each such unit.

By default, the unit-of-work is set to 0, providing a low-overhead *unreliable* service in which undeliverable messages are silently discarded, leaving higher level protocols with the responsibility to detect and retransmit lost packets.

Setting units-of-work to 1 indicates the queue manager will commit or roll back (synchronize) on each message, even though this single message may not constitute the entire transaction.

A value of 1 is required to implement an automatic-backout procedure.

Related Commands

automatic-backout, **backout-queue**, **backout-threshold**, **channel-name**, **hostname**, **message-size**, **queue-manager**

Examples

- Specifies a unit-of-work of 1 message, essentially enabling reliable service for all MQ messages.

```
# units-of-work 1
#
```

- Specifies a unit-of-work of 5 messages; each unit-of-work is reliably processed until all five messages have been delivered.

```
# units-of-work 5
#
```

- Restores default processing.

```
# units-of-work 0
#
```

username

Specifies the plaintext string sent to the server to identify the client.

Syntax

username *string*

Parameters

string Identifies the client.

xml-manager

Assigns an XML Manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of an existing XML Manager. The default is `default`.

Guidelines

The **xml-manager** assigns an XML Manager to this Queue Manager. An XML Manager manages the compilation and caching of style sheets, the caching of documents, and provides configuration constraints on the size and parsing depth of documents. Multiple Queue Manager objects can use the same XML Manager.

Chapter 52. MQ Queue Manager Group configuration mode

This chapter provides an alphabetic listing of commands that are available in MQ Queue Manager Group configuration mode.

To enter this configuration mode, use the Global **mq-qm-group** command.

You can create a MQ Queue Manager Group to implement a failover configuration that provides connection redundancy in the event of a critical queue manager or bus error resulting in loss of connectivity between clients and backend servers.

The Queue Manager Group consists of one primary queue manager object and one or more backup queue managers. Status of the primary queue manager is periodically monitored, and, in the event of failure, one of the backup queue managers, each of which is created with a configurable number of pre-cached connections, is selected to assume the primary role. When, and if, the original primary queue manager returns to service, it reasserts its primary status and resumes its former role.

Most, but not all, of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in MQ Queue Manager Group configuration mode.

backup

Adds a backup queue manager to the current MQ Queue Manager Group.

Syntax

backup *name*

no backup *name*

Parameters

name Specifies the name of an existing MQ queue manager object to be added or removed as a backup member the current MQ Queue Manager Group

Guidelines

Use the **no backup** command to remove a backup MQ queue manager object from an MQ Queue Manager Group.

Related Commands

primary

Examples

- Creates the `redundantMQ` MQ Queue Manager Group. Assigns the `Qman-1` and `Qman-2` MQ queue manager to the group.

```
# mq-qm-group redundantMQ
New MQ Queue Manager Group configuration
# backup Qman-1
# backup Qman-2
#
```

- Accesses the existing redundantMQ MQ Queue Manager Group. Removes the Qman-1 MQ queue manager object from the group. Replaces it with the testQM MQ queue manager.

```
# mq-qm-group redundantMQ
Modify MQ Queue Manager Group configuration
# no backup Qman-1
# backup testQM
#
```

---

primary

Adds the primary queue manager.

Syntax

primary *name*

Parameters

name Specifies the name of an existing MQ queue manager object to add.

Guidelines

The **primary** command specifies the name of an existing MQ queue manager object to add as the primary member to the current MQ Queue Manager Group.

Related Commands

backup

Examples

- Creates the redundantMQ MQ Queue Manager Group. Designates the Alpha MQ queue manager as the primary. Assigns the Beta, Kappa, and Delta MQ queue managers to the group as backup members.

```
# mq-qm-group redundantMQ
New MQ Queue Manager Group configuration
# primary Alpha
# backup Beta
# backup Kappa
# backup Delta
#
```

- Accesses the existing redundantMQ MQ Queue Manager Group. Designates Beta as the new primary. Removes Delta from the group.

```
# mq-qm-group redundantMQ
Modify MQ Queue Manager Group configuration
# primary Beta
# no backup Delta
#
```

Chapter 53. MTOM Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in MTOM Policy configuration mode. MTOM is the abbreviation for SOAP Message Transmission Optimization Mechanism.

To enter this configuration mode, use the Global **mtom** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in MTOM Policy configuration mode.

include-content-type

Indicates whether to include the `contentType` attribute to output messages.

Syntax

include-content-type {**on** | **off**}

Parameters

on (Default) Adds the `contentType` attribute to the output message.

off Does not add the `contentType` attribute to the output message.

Guidelines

The **include-content-type** command determines whether the processing of the MTOM policy adds the `contentType` attribute to output messages when the input message does not contain this attribute. If the input message contains this attribute, the MTOM policy passes through the `contentType` attribute regardless of the setting for the **include-content-type** command.

Examples

- Enters MTOM policy configuration mode to create the `mtom1` MTOM policy, sets the optimization mode to `enable`, and does not add the `contentType` attribute to the output message.

```
# mtom mtom1
MTOM policy configuration mode
# mode enable
# include-content-type off
#
```

mode

Sets the optimization mode for the MTOM policy.

Syntax

mode {**encode** | **decode**}

Parameters

encode
Optimizes an input message.

decode
Extracts the attachment parts on an optimized message, which reconstitutes the original, non-optimized message.

Examples

- Enters MTOM policy configuration mode to create the `mtom1` MTOM policy and sets the optimization mode to `enable`.

```
# mtom mtom1
MTOM policy configuration mode
# mode enable
#
```

---

rule

Defines an MTOM policy rule.

Syntax

rule *XPath* [*type*] [*ID*]

Parameters

XPath Defines the XPath expression that describes which message element to optimize. A single XPath expression can be used to select one or more elements.

type Specifies the Content Type for the extracted data elements. This option overrides the value that is attained from the `xmlmime:contentType` attribute. If the provided XPath matches more than one element, each corresponding MIME attachment part will contain a content-type header with this value. If different content-type values are required, selective XPath expressions are required.

ID If not explicitly configured, content identifiers are automatically generated. Using this option allows for the explicit configuration of content-id headers and associated `href` values. Rules that match multiple data elements result in one attachment part for all matched elements. The resulting attachment part will contain data from the last match only.

Examples

- Enters MTOM policy configuration mode to create the `mtom1` MTOM policy and sets the XPath to `myelement`.

```
# mtom mtom1
MTOM policy configuration mode
# rule "myelement"
#
```

- Enters MTOM policy configuration mode to create the `mtom1` MTOM policy. Defines the rule that sets the XPath to `myelement`, the content type to `image/png`, and the content ID to `example.org/dp`.

```
# mtom mtom1
MTOM policy configuration mode
# rule "myelement" "image/png" "http://example.org/dp"
#
```

Chapter 54. Multi-Protocol Gateway configuration mode

This chapter provides an alphabetic listing of commands that are available in Multi-Protocol Gateway configuration mode.

To enter this configuration mode, use the Global **mpgw** command. This global command creates the specified Multi-Protocol Gateway, if it currently does not exist. While in this mode, you use the commands to define the configuration of a new or an existing Multi-Protocol Gateway.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are available in this configuration mode.

attachment-byte-count

Defines the maximum number of bytes to allow in attachments.

Syntax

attachment-byte-count *bytes*

Parameters

bytes Specifies the maximum number of bytes allowed in any single attachment. If this value is 0, no limit is enforced. The default is 2000000000.

Guidelines

The **attachment-byte-count** command defines the maximum number of bytes to allow in a single attachment. Attachments that exceed this size result in a failure of the entire transaction. Only available when parser limits are enabled.

Related Commands

gateway-parser-limits

Examples

- Sets the maximum attachment size for the current Multi-Protocol Gateway to 5000000 bytes. Any attachment that passes through the gateway can be no larger than 500000 bytes. If larger, the message will be rejected.

```
# attachment-byte-count 500000
#
```

attachment-package-byte-count

Defines the maximum number of bytes to allow for all parts of an attachment package.

Syntax

attachment-package-byte-count *bytes*

Parameters

bytes Specifies the maximum number of bytes allowed for all parts of an attachment package The default is 0.

Guidelines

The **attachment-package-byte-count** command defines the maximum number of bytes allowed for all parts of an attachment package, including the root part. Attachment packages that exceed this size will result in a failure of the whole transaction. If the value is 0, no limit is enforced. This default is 2. Only available when parser limits are enabled.

Related Commands

gateway-parser-limits

---

attribute-count

Defines the maximum number of attributes of a given element. If any of the parser limits are set in the Gateway, they will override those on the XML Manager.

Syntax

attribute-count *count*

Parameters

count Specifies the maximum number of attribute elements allowed by the current Multi-Protocol Gateway. The default is 128.

Related Commands

element-depth, **external-references**, **gateway-parser-limits**, **max-node-size**

Examples

- Sets the maximum number of attributes allowed to 512.

```
# attribute-count 512
#
```

---

back-attachment-format

Specifies the attachment format output to backend servers.

Syntax

back-attachment-format {**dime** | **dynamic** | **mime** | **detect**}

Parameters

dime Specifies that server attachments are DIME-encapsulated documents.

dynamic
Specifies that server attachments are deduced from front end content.

mime specifies that server attachments are MIME-encapsulated documents.

detect Specifies that server attachments are deduced from message data.

Related Commands

front-attachment-format

Examples

- Specifies that attachments output to servers are DIME-encapsulated.

```
# back-attachment-format dime
#
```

back-persistent-timeout

Sets the inter-transaction timeout between the completion of a TCP transaction and the initiation of a new TCP one on the gateway-to-server connection.

Syntax

back-persistent-timeout *timerValue*

Parameters

timerValue
Specifies the maximum inter-transaction idle time in seconds. The value must be in the range of 0 to 7200. The default is 180. A value of 0 disables persistent connections.

Guidelines

The **back-persistent-timeout** command sets the value of the inter-transaction timeout. This value is the maximum idle time to allow between the completion of a TCP transaction and the initiation of a new TCP transaction on the gateway-to-server connection. If the specified idle timeout is exceeded, the connection is torn down.

An idle TCP connection might remain in the idle state for as long as 20 seconds after the expiration of the inter-transaction timer.

Related Commands

back-timeout, **front-persistent-timeout**, **front-timeout**, **persistent-connections**

back-timeout

Sets the intra-transaction timeout for the gateway-to-server connection.

Syntax

back-timeout *timerValue*

Parameters

timerValue
Specifies the maximum intra-transaction idle time in seconds. The value must be in the range of 10 to 86400. The default is 120.

Guidelines

The **back-timeout** command sets the value for the intra-transaction timeout. This value is the maximum idle time to allow within a transaction on the gateway-to-server connection. This timer monitors the interval between sending

the client request and receiving the server response. In other words, this time monitors the idle time within the data transfer process. If the specified idle time is exceeded, the connection is torn down.

Related Commands

back-persistent-timeout, **front-timeout**, **front-persistent-timeout**, **persistent-connections**

---

backend-url

Specifies the URL to which all traffic to the static backend server is routed.

Syntax

backend-url *url*

Parameters

url Specifies a URL that fully identifies where all traffic is routed by default. This property can take one of the following general forms:

- `http://host:port/URI`
- `https://host:port/URI`
- `dpmq://queueManager/URI?RequestQueue=qName;ReplyQueue=qName...`
- `dptibems://server?RequestQueue=qName&ReplyQueue=qName...`
- `dpwasjms://server?RequestQueue=qName&ReplyQueue=qName...`

To use a load balancer, specify the name of an existing Load Balancer Group instead of the address-port pair in the URL.

Guidelines

HTTP and HTTPS

The service uses the HTTP or HTTPS protocol to connect the host at the specified port. The URL includes the URI. The URL or URI might be rewritten by other configuration options. With HTTPS, the configured SSL Proxy handles the SSL security negotiation.

MQ (dpmq)

The service uses the MQ protocol to connect to the requested Queue Manager. The URL includes the URI and query parameters for the request and reply queues. Optionally, the URL includes query parameters for transactionality and user identity.

The complete URL is as follows:

`dpmq://`*`queueManager`*`/`
`URI``?RequestQueue=`*`QName`*`;ReplyQueue=`*`QName`*`;Sync=`*`value`*`;PMO=`*`value`*

queueManager
Specifies the name of an existing MQ Queue Manager object.

URI Specifies the name of a service. Enter a string, such as `/SomeBank/services/checking` to be included in the URL.

`RequestQueue=`*QName*
Specifies the name of a queue that is managed by the specified Queue Manager. The DataPower service places requests on this queue.

ReplyQueue=*QName*;
: Specifies the name of a queue that is managed by the specified Queue Manager. The DataPower service polls this queue for responses.

Sync=*value*
: Optionally indicates whether the DataPower service implements transactionality. Use the binary `true` or `false`. If `true`, the service implements transactionality. When implemented, the service does not consider a request to be successfully delivered until it receives a corresponding response. If not specified, the default is `false`.

PMO=*value*
: Optionally sets the `MQPMO.Options` field on the **MQPUT** call. The specified value is a cumulative value in decimal format of all acceptable options. If not specified, the default is `MQPMO_NO_SYNCPOINT` (decimal `4`, hexadecimal `0x00000004`.

Table 11. MQPMO.Options available for MQPUT calls

Put-Message option	Hexadecimal value	Decimal value
MQPMO_SYNCPOINT	0x00000002	2
MQPMO_NO_SYNCPOINT	0x00000004	4
MQPMO_NEW_MSG_ID	0x00000040	64
MQPMO_NEW_CORREL_ID	0x00000080	128
MQPMO_LOGICAL_ORDER	0x00008000	32728
MQPMO_NO_CONTEXT	0x00004000	16384
MQPMO_DEFAULT_CONTEXT	0x00000020	32
MQPMO_PASS_IDENTITY_CONTEXT	0x00000100	256
MQPMO_PASS_ALL_CONTEXT	0x00000200	512
MQPMO_SET_IDENTITY_CONTEXT	0x00000400	1024
MQPMO_SET_ALL_CONTEXT	0x00000800	2048
MQPMO_ALTERNATE_USER_AUTHORITY	0x00001000	4096
MQPMO_FAIL_IF_QUIESCING	0x00002000	8192
MQPMO_NONE	0x00000000	0

TIBCO EMS (dptibems and dptibemss)
: The service uses the TIBCO EMS protocol to connect to the requested TIBCO server. The URL includes the server and query parameters for the request and reply queues. Optionally, the URL includes the `Selector` query parameter. With `dptibemss`, the configured SSL Proxy handles the SSL security negotiation.

The complete URL is as follows:

`dptibems:///`*`server`*`?RequestQueue=`*`qName`*`&ReplyQueue=`*`qName`*
`&Selector=`*`expression`*

server Specifies the name of an existing TIBCO Server object.

RequestQueue=*qName*
: Specifies the name of a queue or topic that is managed by the TIBCO server. Enter the queue or topic for the request.

`ReplyQueue=`*qName*
Specifies the name of a queue or topic that is managed by the TIBCO server. Enter the queue or topic for the reply.

`Selector=`*expression*
Optionally specifies an SQL92 conditional expression to identify *messages of interest*.

WebSphere JMS (dpwasjms and dpwasjmss)
The service uses the WebSphere JMS protocol to connect to the requested WebSphere JMS server. The URL includes the server and query parameters for the request and reply queues. Optionally, the URL includes the following query parameters:
- Request topic spaces
- Reply topic spaces
- Selector
- Timeout

With `dpwasjmss`, the configured SSL Proxy handles the SSL security negotiation.

The complete URL is as follows:

`dpwasjms:///`*server*`?RequestQueue=`*qName*`&ReplyQueue=`*qName*
`&RequestTopicSpace=`*topicSpace*`&ResponseTopicSpace=`*topicSpace*
`&Selector=`*expression*`&TimeOut=`*timeout*

server Specifies the name of an existing WebSphere JMS object.

`RequestQueue=`*qName*
Specifies the name of a queue or topic that is managed by the WebSphere JMS server. If using queues, specify the queue for the request. If using topic spaces, specify this query parameter without a value.

`ReplyQueue=`*qName*
Specifies the name of a queue or topic that is managed by the WebSphere JMS server. If using queues, specify the queue for the reply. If using topic spaces, specify this query parameter without a value.

`RequestTopicSpace=`*topicSpace*
Optionally specifies a non-default request topic namespace.

`ResponseTopicSpace=`*topicSpace*
Optionally specifies a non-default reply topic namespace.

`Selector=`*expression*
Optionally specifies an SQL92 conditional expression to identify *messages of interest*..

`Timeout=`*timeout*
Optionally specifies a timeout value.

Related Commands

type

Examples

- Sets the static backend URL to `http://10.10.10.2:3000/services`.

```
# backend-url http://10.10.10.2:3000/services
#
```

- Sets the static backend URL to `https://10.10.10.2:3000/services`. To support the SSL connection with the backend server, assigns the `clientssl` SSL Proxy Profile to provide the credentials for the secure connection.

```
# backend-url https://10.10.10.2:3000/services
# ssl clientssl
#
```

- Sets the static backend URL to `dpmq://BackEndQM/services?RequestQueue=Put_Q_Service;ReplyQueue=Get_Q_Service;UserName=aname;TimeOut=3000;Size=50000`.

```
# backend-url dpmq://BackEndQM/services?RequestQueue=
Put_Q_Service;ReplyQueue=Get_Q_Service;UserName=aname;TimeOut=3000;
Size=50000
#
```

chunked-uploads

Controls the ability to send Content-Type Chunked Encoded documents to the backend server.

Syntax

chunked-uploads {**on** | **off**}

Parameters

on Enables chunked encoding when sending HTTP 1.1 requests to the backend server.

off (default) Disables chunked encoding when sending HTTP 1.1 requests to the backend server.

Guidelines

The gateway might send an HTTP 1.1 request to the backend server. In this case, the body of the document can be delimited by either `Content-Length` or chunked encoding. All servers will understand how to interpret `Content-Length`, and many applications will fail to understand chunked, so `Content-Length` is generally used. However doing so interferes with the ability of the appliance to fully stream. If you must stream full documents towards the backend, this property should be turned on. Unlike all other HTTP 1.1 features that can be negotiated down at runtime, if necessary, you must know beforehand that the server you are communicating with is RFC 2616 compatible. You might also consider leaving this property turned off and turning it on a per-URL basis with the User Agent configuration.

Examples

- Enables HTTP 1.1 chunked encoding and subsequently disable chunked encoding support, thus restoring the default state.

```
# chunked-uploads on
⋮
# chunked-uploads off
#
```

compression

Controls GZIP compression negotiation between the Multi-Protocol Gateway and the backend server.

Syntax

compression {**on** | **off**}

Parameters

on Enables GZIP compression negotiation on backend.

off Disables GZIP compression negotiation on backend.

Examples

- Enables compression negotiation and subsequently disables such negotiation thus restoring the default state.

```
# compression on
⋮
# compression off
#
```

default-param-namespace

Specifies the namespace into which to assign the parameter.

Syntax

default-param-namespace *URL*

Parameters

URL Specifies a valid namespace URL. The default is `http://www.datapower.com/param/config`.

Guidelines

If a stylesheet parameter is defined without a namespace (or without explicitly specifying the null namespace), use the **default-param-namespace** command to specify the namespace into which the parameter is assigned.

Related Commands

default-query-namespace, **parameter**

Examples

- Identifies `http://www.space.com/wsnames` as the default parameter namespace.

```
# default-param-namespace http://www.space.com/wsnames
#
```

element-depth

Defines the maximum depth of element nesting in the XML parser.

Syntax

element-depth *depth*

Parameters

depth Defines the maximum depth of element nesting in XML parser. Defaults to 512.

Guidelines

The **element-depth** command defines the maximum depth of element nesting in the XML parser. If any of the parser limits, these explicit assignments override those on the XML Manager.

Related Commands

attribute-count, **external-references**, **gateway-parser-limits**, **max-node-size**

Examples

- Sets the maximum element depth to 128.

```
# element-depth 128
#
```

---

external-references

Defines the handling mode for input documents that contain external references.

Syntax

external-references {**allow** | **forbid** | **ignore**}

Parameters

allow Allows and resolves external references.

forbid Stops processing if the XML parser encounters an external reference.

ignore (Default) Ignores external references and replaces external entities with the empty string.

Guidelines

The **external-references** command defines the behavior for input documents that contain external references. External references can be external entities or external DTD definitions.

Examples

- Specifies that the XML parser processes documents that contain external references but ignores such references.

```
# external-references ignore
#
```

---

follow-redirects

Controls the resolution of redirects.

Syntax

follow-redirects {**on** | **off**}

Parameters

on (Default) Resolves redirects.

off Disables the resolution of redirects.

Guidelines

Some protocols generate redirects as part of the protocol; for example, HTTP response code 302. Use the **follow-redirects** command to specify if the Multi-Protocol Gateway attempts to resolve redirects.

Examples

- Disables redirect resolution and subsequently enables redirect resolution, which restores the default state.

```
# follow-redirects off
⋮
# follow-redirects on
#
```

forbid-external-references (deprecated)

Comments

This command has been deprecated. Use the **external-references** command in its place.

front-attachment-format

Specifies the attachment format received from front end clients.

Syntax

front-attachment-format {**dime** | **dynamic** | **mime** | **detect**}

Parameters

dime Specifies that client attachments are DIME-encapsulated documents.

dynamic
Specifies that client attachments are deduced from document content.

mime Specifies that client attachments are MIME-encapsulated documents.

detect Specifies that client attachments are deduced from message data.

Related Commands

back-attachment-format

Examples

- Specifies that front end attachments are DIME-encapsulated.

```
# front-attachment-format dime
#
```

front-persistent-timeout

Sets the inter-transaction timeout between the completion of a TCP transaction and the initiation of a new one on the gateway-to-client connection.

Syntax

front-persistent-timeout *timerValue*

Parameters

timerValue
Specifies the maximum inter-transaction idle time in seconds. Use an integer in the range of 0 to 7200. The default is 180. A value of 0 disables persistent connections.

Guidelines

The **front-persistent-timeout** command sets the value of the inter-transaction timeout. This value is the maximum idle time to allow between the completion of a TCP transaction and the initiation of a new TCP transaction on the gateway-to-client connection. If the specified idle timeout is exceeded, the connection is torn down.

An idle TCP connection might remain in the idle state for as long as 20 seconds after the expiration of the persistence timer.

Related Commands

back-persistent-timeout, **back-timeout**, **front-timeout**, **persistent-connections**

---

front-protocol

Assigns a front side protocol handlers.

Syntax

front-protocol *name*

Parameters

name Specifies the name of an existing front side protocol handler.

Related Commands

front-timeout, **request-type**, **request-attachments**

Guidelines

Issue the **front-protocol** command as many times as needed to add the desired front side protocol handlers. You must first create the handlers using the **source-http**, **source-https**, **source-mq**, or **source-raw** Global command.

Examples

- Creates the `httpHandler` Front Side Protocol Handler and assigns the local address of 10.10.12.10 and port of 16000. Assigns the handler to the `schemafilter` Multi-Protocol Gateway.

```
# source-http httpHandler
HTTP Front Side Handler configuration mode
# local-address 10.10.12.10
# port 16000
# exit
# mpgw schemafilter
Multi-Protocol Gateway configuration mode
# front-protocol http-server
# exit
#
```

front-timeout

Sets the maximum idle time to allow in a transaction on the gateway-to-client connection.

Syntax

front-timeout *timerValue*

Parameters

timerValue
Specifies the maximum intra-transaction idle time in seconds, Use and integer in the range of 10 through 86400. The default is 120.

Guidelines

The **front-timeout** command sets the value of the intra-transaction timeout. This value is the maximum idle time to allow in a transaction on the gateway-to-client connection. This timer monitors idle time in the data transfer process. If the specified idle time is exceeded, the connection is torn down.

Related Commands

back-persistent-timeout, **back-timeout**, **front-persistent-timeout**, **persistent-connections**

fwcred

Assigns a Firewall Credentials list.

Syntax

fwcred *name*

Parameters

name Specifies the name of an existing Firewall Credentials List. If not specified, all locally stored keys and certificates are available.

Guidelines

The **fwcred** command assigns a Firewall Credentials list to the current Multi-Protocol Gateway. A Firewall Credentials list specifies which keys and certificates are available to support gateway processing.

Related Commands

fwcred (Crypto)

Examples

- Assigns the `standardCreds` Firewall Credentials List to the current Multi-Protocol Gateway.

```
# fwcred standardCreds
#
```

gateway-parser-limits

Controls gateway-specific parser limitations.

Syntax

gateway-parser-limits

no gateway-parser-limits

Guidelines

Parser limitations guard against denial of service attacks that use malicious XML documents seeking to exhaust system resources.

With gateway-specific parser limitations enabled, the values specified by the **attachment-byte-count**, **attribute-count**, **element-depth**, **max-message-size**, and **max-node-size** commands (Multi-Protocol Gateway) are used to evaluate incoming XML documents.

With gateway-specific parser limitations disabled (the default condition), parser limitations, if any, are derived from the assigned XML Manager. Use the **no gateway-parser-limits** command to disable gateway-specific parser limitations.

Related Commands

attribute-count, **element-depth**, **external-references**, **max-node-size**

Examples

- Enables gateway-specific parser limits for the current Multi-Protocol Gateway.

```
# gateway-parser-limits
# attribute-count 512
# element-depth 128
# max-message-size 500000
# max-node-size 50000
#
```

- Disables gateway-specific parser limitations for the current Multi-Protocol Gateway. The Multi-Protocol Gateway inherits parser limits, if any, from the assigned XML Manager.

```
# no gateway-parser-limits
#
```

host-rewriting

Controls host rewriting.

Syntax

host-rewriting {**on** | **off**}

Parameters

on Enables host rewriting.

off Disables host rewriting.

Related Commands

urlrewrite-policy, **propagate-uri**

Guidelines

Some protocols have distinct name-based elements, separate from the URL, to demultiplex. HTTP uses the Host header for this purposes. If this feature is enabled the backside server will receive a request reflecting the final route. Otherwise it will receive a request reflecting the information as it arrived at the DataPower appliance. Web servers issuing redirects might want to disable this feature, as they often depend on the host header for the value of their redirect.

Examples

- Disables host rewriting and subsequently enables host rewriting, which restores the default state.

```
# host-rewriting off
:
# host-rewriting on
#
```

http-client-ip-label

Identifies the HTTP header that contains the IP address of the calling client.

Syntax

http-client-ip-label *header*

no http-client-ip-label

Parameters

header Identifies the HTTP header that contains the IP address. The default is `X-Client-IP`.

Guidelines

The **http-client-ip-label** command identifies the HTTP header that contains the IP address of the calling client. When defined, the IP address of the calling client is read from this HTTP header. This IP address will then be used for monitoring and logging.

Use the **no http-client-ip-label** command to disable the reading of the HTTP header to identify the IP address of the calling client.

Examples

- Disables the reading of the HTTP header to identify the IP address of the calling client. Subsequently, enables this function to read the IP address from the `X-Forwarded-For` HTTP header for monitoring and logging.

```
# no http-client-ip-label
:
# http-client-ip-label X-Forwarded-For
#
```

http-server-version

Selects the HTTP version to use on the server-side (backend) connection.

Syntax

http-server-version {**HTTP/1.0** | **HTTP/1.1**}

Parameters

HTTP/1.0
Specifies the server uses HTTP 1.0.

HTTP/1.1
(Default) Specifies the server uses HTTP 1.1

Examples

- Indicates that the connection to the backend server uses HTTP 1.0.

```
# http-server-version http/1.0
#
```

include-content-type-encoding

Controls the inclusion of character set encoding data in content-type headers.

Syntax

include-content-type-encoding {**on** | **off**}

Parameters

on Enables the inclusion of character set encoding date in content-type headers.

off Disables the inclusion of character set encoding date in content-type headers.

Guidelines

Assume a UTF-8 encoded XML document.

- When enabled, the content-type header contains:

```
text/xml; charset=UTF-8
```

- When disabled, the content-type the content-type header contains:

```
text/xml
```

Examples

- Enables the inclusion of content-type encoding and subsequently disables such inclusion, which restores the default state.

```
# include-content-type-encoding on
:
# include-content-type-encoding off
#
```

inject

Injects proprietary HTTP header fields into the packet stream between the current Multi-Protocol Gateway and an HTTP client or server.

Syntax

inject {**front** | **back**} *header value*

no inject {**front** | **back**} *header*

Parameters

front Designates the packet stream between a Multi-Protocol Gateway and the HTTP client.

back Designates the packet stream between a Multi-Protocol Gateway and the HTTP server.

header Identifies a proprietary HTTP header field. This property is case-sensitive.

value Specifies the value of the field and can contain a character string or an integer. This property is case-sensitive.

Guidelines

Use the **no inject** command to remove a previously-injected proprietary HTTP header field.

Related Commands

suppress

Examples

- Injects the `ProcInst` HTTP header field with a value of 0 into the packet stream directed to the HTTP client.

```
# inject front ProcInst 0
#
```

- Removes the `ProcInst` HTTP header field from the packet stream.

```
# no inject front ProcInst
#
```

load-balancer-hash-header

Syntax

Uses the value of an HTTP header

load-balancer-hash-header *header*

Uses the client IP address

no load-balancer-hash-header

Parameters

header Specifies the name of the HTTP header.

Guidelines

The **load-balancer-hash-header** command identifies the HTTP header to use for calculating the hash for load balancing traffic to the backend servers.

- When defined, the hash algorithm uses the value of the identified HTTP header.
- When not defined, the hash algorithm uses the IP address of the client.

This command is relevant only when the value defined by the **algorithm** command in Load Balancer configuration mode is **hash**.

Use the **no load-balancer-hash-header** command to disable the use of an HTTP header as the hash algorithm to use for load balancing.

Related Commands

algorithm

Examples

- Disables the use of an HTTP header for load balancing (uses the IP address to calculate the hash). Subsequently, enables load balancing traffic to the backend servers using a hash algorithm identified by the `X-Forwarded-For` HTTP header.

```
# no load-balancer-hash-header
⋮
# load-balancer-hash-header X-Forwarded-For
#
```

loop-detection

Controls loop detection behavior in the network.

Syntax

loop-detection {**on** | **off**}

Parameters

on Enables a loop detection mechanism.

off (Default) Disables a loop detection mechanism.

Guidelines

Some protocols provide a loop detection mechanism that can detect network loops. Loop detection is a good policy, but it runs the risk that the current Multi-Protocol Gateway might be publicly recorded in a transmitted message. Such visibility might be considered an information leak.

Use the **loop-detection** command to specify if the Multi-Protocol Gateway assists in loop detection within the network.

Related Commands

inject, **suppress**

Guidelines

When enabled, the Multi-Protocol Gateway inserts a `Via:` HTTP header that contains the gateway name in the HTTP transmission.

Examples

- Enables loop-detection and subsequently disables it, which restores the default state.

```
# loop-detection on
⋮
# loop-detection off
```

max-message-size

Specifies the maximum size of a SOAP or XML message.

Syntax

max-message-size *kilobytes*

Parameters

kilobytes
Specifies the maximum message size in kilobytes. Use an integer in the range of 0 through 2097151. The default is 0. A value of 0 specifies unlimited size.

Related Commands

attachment-byte-count, **attribute-count**, **element-depth**, **gateway-parser-limits**, **max-node-size**

Examples

- Sets the maximum message size to 500000 kilobytes.

```
# max-message-size 500000
#
```

max-node-size

Specifies the maximum size of a single XML node.

Syntax

max-node-size *bytes*

Parameters

bytes Specifies the maximum message node size in bytes. The default is 0. A value of 0 indicates that no size limit is applied to incoming message nodes.

Related Commands

attribute-count, **element-depth**, **external-references**, **gateway-parser-limits**

Examples

- Sets the maximum allowed node size to 50000 bytes.

```
# max-node-size 50000
#
```

mime-back-headers

Controls MIME multipart messages sent over HTTP in server responses.

Syntax

mime-back-headers {**on** | **off**}

Parameters

on (Default) Enables the ability to handle MIME package headers in the HTTP body of messages that are received from a server.

off Disables the ability to handle MIME package headers in the HTTP body of messages that are received from a server.

Guidelines

The body of a message (that is, the payload, independent of any protocol headers) can sometimes contain MIME headers before any preamble and before the first MIME boundary contained in the body of the message. These MIME headers can contain important information not available in the protocol headers, such as the string identifying the MIME boundary. If this property is set to on, then these MIME headers will be processed by the gateway.

Note that if this is on and there are no MIME headers contained in the message, the appliance will continue to try and parse the message, using the protocol header information, if available.

When this is off and MIME headers are present in the body of the message, these MIME headers will be considered part of the preamble, and not used to parse out the message. If the protocol headers (such as HTTP) indicate the MIME boundaries, the appliance can parse and process individual attachments. If such information is not available, no attachments can be parsed from the body of the message.

Related Commands

mime-front-headers, **request-attachments**, **response-attachments**

Examples

- Disables server-side support for MIME package headers and subsequently enables support, which restores the default state.

```
# mime-back-headers off
.
.
.
# mime-back-headers on
#
```

mime-front-headers

Controls MIME multipart messages sent over HTTP in client requests.

Syntax

mime-front-headers {**on** | **off**}

Parameters

on (Default) Enables the ability to handle MIME package headers in the HTTP body of messages that are received from a client.

off Disables the ability to handle MIME package headers in the HTTP body of messages that are received from a client.

Guidelines

The body of a message (that is, the payload, independent of any protocol headers) can sometimes contain MIME headers before any preamble and before the first MIME boundary contained in the body of the message. These MIME headers can contain important information not available in the protocol headers, such as the string identifying the MIME boundary. If this property is set to on, then these MIME headers will be processed by the gateway.

If this is on and there are no MIME headers contained in the message, the appliance will continue to try and parse the message, using the protocol header information, if available.

When this is off and MIME headers are present in the body of the message, these MIME headers will be considered part of the preamble, and not used to parse out the message. If the protocol headers (such as HTTP) indicate the MIME boundaries, the appliance can parse and process individual attachments. If such information is not available, no attachments can be parsed from the body of the message.

Related Commands

mime-back-headers, **request-attachments**, **response-attachments**

Examples

- Disables client-side support for MIME package headers and subsequently enables support, which restores the default state.

```
# mime-front-headers off
⋮
# mime-front-headers on
#
```

---

monitor-count

Assigns a Count Monitor.

Syntax

monitor-count *name*

no monitor-count *name*

Parameters

name Specifies the name of an existing Count Monitor.

Guidelines

Use the **monitor-count** command to assign one or more Count Monitors to the current Multi-Protocol Gateway.

Count Monitors watch for defined messaging events and increment counters each time the event occurs. When a certain threshold is reached, the monitor can either post a notification to a log or block service for a configured amount of time.

Use the **no monitor-count** command to remote the assignment of a Count monitor from a Multi-Protocol Gateway.

Related Commands

monitor-duration (Global), **monitor-service** (Global)

Examples

- Assigns the `mpgwCounter` Count Monitor to the current Multi-Protocol Gateway.

```
# monitor-count mpgwCounter
#
```

- Removes the `mpgwCounter` Count Monitor.

```
# no monitor-count mpgwCounter
#
```

monitor-duration

Assigns a Duration Monitor.

Syntax

monitor-duration *name*

no monitor-duration *name*

Parameters

name Specifies the name of a Duration Monitor.

Guidelines

Use the **monitor-duration** command to assign a Duration Monitor to the current Multi-Protocol Gateway.

Duration Monitors watch for events that meet or exceed a configured duration. When a duration is met or exceeded, the monitor can either post a notification to a log or block service for a configured amount of time.

Use the **no monitor-duration** command to remove the assignment of a Duration Monitor from a Multi-Protocol Gateway.

Related Commands

monitor-count (Global), **monitor-service** (Global)

Examples

- Assigns the `mpgwDuration` Duration Monitor to the current Multi-Protocol Gateway.

```
# monitor-duration mpgwDuration
#
```

- Removes the `mpgwDuration` Duration Monitor.

```
# no monitor-duration mpgwDuration
#
```

monitor-processing-policy

Sets the behavior when a service has multiple monitors.

Syntax

monitor-processing-policy {**terminate-at-first-throttle** | **terminate-at-first-match**}

Parameters

terminate-at-first-throttle
(Default) Monitors will execute in the order in which they are listed. After any monitor either shapes (buffers to delay) or rejects a message, none of the further monitors will execute.

terminate-at-first-match
Monitors will execute in the order in which they are listed. After any monitor matches a message and takes any action, none of the further monitors will execute.

Examples

- Allows only the first matching monitor to execute when a service has multiple monitors attached.

```
# monitor-processing-policy terminate-at-first-match
#
```

---

monitor-service

Assign a Service Level Monitor.

Syntax

monitor-service *name*

no monitor-service *name*

Parameters

name Specifies the name of the Service Level Monitor.

Guidelines

Use the **monitor-service** command to assign a Service Level Monitor to the current Multi-Protocol Gateway.

Service Level Monitors watch Web Services endpoints. A Service Level Monitor collects statistics, establishes count and duration monitors and can take action when thresholds are met or exceeded.

Use the **no monitor-service** command to remove the monitor from the Multi-Protocol Gateway assignment.

Related Commands

monitor-count (Global), **monitor-duration** (Global)

Examples

- Assigns the `mpgwSLM` Service Level Monitor to the current Multi-Protocol Gateway.

```
# monitor-service mpgwSLM
#
```

- Removes the mpgwSLM Service Level Monitor.

```
# no monitor-service mpgwSLM
#
```

parameter

Defines stylesheet parameters for processing policies.

Syntax

parameter *name value*

no parameter [*name*]

Parameters

name is the name of the parameter made available to the current Multi-Protocol Gateway.

value is the value of the parameter.

Guidelines

Style sheets that are used in processing policies can take stylesheet parameters. These parameters can be passed in. Use the **parameter** to define each required stylesheet parameter.

You must include the following namespace declaration in any style sheet to enable that style sheet to access parameter-value pairs defined by the **parameter** command:

```
xmlns:dpconfig="http://www.datapower.com/param/config"
```

Use the **no parameter** command to delete one or more stylesheet parameters.

Related Commands

default-param-namespace, **query-param-namespace**

Examples

- Makes two parameter-value pairs available to the current Multi-Protocol Gateway. The default parameter namespace is used.

```
# parameter recipient ALICE
# parameter type content
#
```

- Makes a single parameter-value pair available to the current Multi-Protocol Gateway. {http://www.example.com} designates the parameter namespace.

```
# parameter {}foobar value
#
```

- Makes a single parameter-value pair available to the current Multi-Protocol Gateway. {} designates no namespace.

```
# parameter {http://www.example.com}foobar value
#
```

- Deletes the recipient stylesheet parameter.

```
# no parameter recipient
#
```

- Deletes all stylesheet parameters.

```
# no parameter
#
```

persistent-connections

Controls persistent connections.

Syntax

persistent-connections {**on** | **off**}

Parameters

on (Default) Enables the establishment of persistent connections.

off Disables the establishment of persistent connections.

Guidelines

With persistent connections enabled, the default state for both HTTP 1.0 and HTTP 1.1, the appliance negotiates with the remote HTTP peer and establishes a persistent connection if agreeable to the peer.

With persistent connections disabled, the appliance refuses to negotiate the establishment of persistent connections.

Related Commands

back-persistent-timeout, **back-timeout**, **front-persistent-timeout**, **front-timeout**

Examples

- Disables persistent connections and subsequently enables such connections, which restores the default state.

```
# persistent-connections off
:
# persistent connections on
#
```

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal (Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

process-http-errors

Indicates whether to processing errors from the backend server.

Syntax

process-http-errors {**on** | **off**}

Parameters

on (default) Ignores the error condition, and processes the response rule.

off Notices the error condition, and processes the error rule.

Guidelines

The **process-http-errors** command indicates whether to process errors from the backend server.

Depending on the protocol, the backend service might return a response code that indicates an error condition. For HTTP messages, the response from the backend server might include a response body that contains XML that provides more details about the error. For MQ messages, the response from the backend MQ server does not provide a response message.

---

propagate-uri

Enables or disables the propagation of the local portion of URL from the URL given by the client to the URL used to contact the backend server.

Syntax

propagate-uri {**on** | **off**}

Parameters

on (default) Enables the propagation of the client URI to the backend.

off Disables the propagation of the client URI to the backend.

Guidelines

The **propagate-uri** command enables or disables the propagation of the client URL to the backend server.

Enabling URI propagation is meaningful in the following situations only:

- When the service is configured to use a static backend.
- When the service is configured to use a dynamic backend and dynamic routing is set with a route with style sheet (`route-action`) action in the processing policy. In this case, use the `dp:set-target()` extension element to define that target backend server.

  For the other dynamic routing options that are available with the `route-action` and `route-set` actions, the URI is absolute.

When enabled, the service rewrites the URI of the backend URL to the URI in the client request. If URI propagation is enabled and the client submits `http://host/service` and the backend URL is `http://server/listener`, the URL is rewritten to `http://server/service`.

Notes:

1. When enabled, any Matching Rule must match the rewritten URL.

2. Any action in the Processing Policy can change the URI that is sent to the backend server. The rewritten URI could override the intended effect of this setting.

Related Commands

urlrewrite-policy

Examples

- Disables client URI propagation and subsequently enables such propagation thus restoring the default state.

```
# propagate-uri off
:
# propagate-uri on
#
```

query-param-namespace

Identifies the namespace in which to put all parameters that are specified in the URL query string.

Syntax

query-param-namespace *namespace*

Parameters

namespace
Enter a valid namespace URL. Defaults to:

```
http://www.datapower.com/param/query
```

Related Commands

default-param-namespace, **parameter**

Examples

- Assigns the namespace `http://www.example.com/queries` to all query parameters in the client URL.

```
# query-param-namespace http://www.example.com/queries
#
```

request-attachments

Specifies the processing mode for SOAP attachments in client requests.

Syntax

request-attachment *mode*

Parameters

mode Specifies one of the following keywords to indicate the processing mode for SOAP attachments:

allow Allows messages that contain attachments, and processes *needed* attachments. Needed attachments are buffered, but attachments that are not needed might be streamed directly to output.

Attachments are buffered when an action in the processing rule requests any of the following:

- Needed attachments
- All attachments in the package before the needed attachment
- All attachments in the package for a needed manifest
- All attachments in the package if the package does not contain the needed attachment

reject Rejects messages that contain attachments.

strip (Default) Removes attachments from the message before processing.

streaming
Allows messages that contain attachments in streaming mode, but provides limited processing. Messages in the form of a *SOAP message package*, which is a SOAP with Attachments message, are supported. Processing can be applied individually to each attachment. The appliance does not create a manifest of all attachments. Attachments must be accessed and processed in the order that they appear in the package.

unprocessed
Allows messages that contain attachments, but does not process attachments. This option permits the forwarding of messages that contain large attachments. The root part of the message, which typically contains a SOAP message, is subject to filter and transform actions. No processing of parts other than the root part is possible. Accompanying documents can be passed intact.

Guidelines

The **request-attachment** command specifies the processing mode for attachments in client requests (as defined in RFC 2387). This type of request is a compound object that consists of several interrelated body parts and is the mechanism that is used to support the bundling of attachments in a *SOAP message package*, which is commonly referred to as a SOAP with Attachments message.

Meaningful only, if the value of the **request-type** command is **soap**.

Related Commands

request-type

Examples

- Provides full SOAP with Attachments support.

```
# request-attachments allow
#
```

- Provides partial SOAP with Attachments support.

```
# request-attachments streaming
#
```

request-type

Characterizes the client-originated traffic stream.

Syntax

request-type {**preprocessed** | **xml** | **soap** | **unprocessed**}

Parameters

preprocessed
Characterizes the client-originated traffic stream as non-XML traffic that is not transformed by the Multi-Protocol Gateway. The Multi-Protocol Gateway might operate on other aspects of the message, such as determining the route, or performing authentication and authorization.

xml Characterizes the client-originated traffic stream as *raw* (unencapsulated) XML.

soap Characterizes the client-originated traffic stream as SOAP.

unprocessed
(Default) Characterizes the client-originated traffic stream as non-XML traffic that is not transformed by the Multi-Protocol Gateway.

Related Commands

response-type, **soap-schema-url**

Examples

- Characterizes client-originated traffic as XML.

```
# request-type xml
#
```

- Characterizes client-originated traffic as SOAP.

```
# request-type soap
#
```

---

response-attachments

Specifies the processing mode for SOAP attachments in server responses.

Syntax

response-attachments *mode*

Parameters

mode Specifies one of the following keywords to indicate the processing mode for SOAP attachments:

allow Allows messages that contain attachments, and processes *needed* attachments. Needed attachments are buffered, but attachments that are not needed might be streamed directly to output.

Attachments are buffered when an action in the processing rule requests any of the following:

- Needed attachments
- All attachments in the package before the needed attachment
- All attachments in the package for a needed manifest
- All attachments in the package if the package does not contain the needed attachment

reject Rejects messages that contain attachments.

strip (Default) Removes attachments from the message before processing.

streaming
Allows messages that contain attachments in streaming mode, but provides limited processing. Messages in the form of a *SOAP message package*, which is a SOAP with Attachments message, are supported. Processing can be applied individually to each attachment. The appliance does not create a manifest of all attachments. Attachments must be accessed and processed in the order that they appear in the package.

unprocessed
Allows messages that contain attachments, but does not process attachments. This option permits the forwarding of messages that contain large attachments. The root part of the message, which typically contains a SOAP message, is subject to filter and transform actions. No processing of parts other than the root part is possible. Accompanying documents can be passed intact.

Guidelines

The **response-attachment** command specifies the processing mode for attachments in server responses (as defined in RFC 2387). This type of request is a compound object that consists of several interrelated body parts and is the mechanism that is used to support the bundling of attachments in a *SOAP message package*, which is commonly referred to as a SOAP with Attachments message.

Meaningful only when the value of the **response-type** command is **soap**.

Related Commands

response-type

Examples

- Provides full SOAP with Attachments support.

```
# request-attachments allow
#
```

- Provides partial SOAP with Attachments support.

```
# request-attachments streaming
#
```

response-type

Characterizes the server-originated traffic stream.

Syntax

response-type {**preprocessed** | **xml** | **soap** | **unprocessed**}

Parameters

preprocessed
Characterizes the server-originated traffic stream as non-XML traffic that is not transformed by the Multi-Protocol Gateway. The Multi-Protocol Gateway might operate on other aspects of the message, such as determining the route, or performing authentication and authorization.

xml Characterizes the server-originated traffic stream as raw (unencapsulated) XML.

soap Characterizes the server-originated traffic stream as SOAP.

unprocessed
Characterizes the server-originated traffic stream as non-XML traffic that is not transformed by the Multi-Protocol Gateway.

Related Commands

request-type, **soap-schema-url**

Examples

- Characterizes server-originated traffic as XML.

```
# response-type xml
#
```

- Characterizes server-originated traffic as SOAP.

```
# response-type soap
#
```

root-part-not-first-action

Defines the action to take when the MIME message root part is not first.

Syntax

root-part-not-first-action {**abort** | **buffer** | **process-in-order**}

Parameters

abort Stops the transaction and return an error.

buffer Buffers attachments before the root part into memory. Then processes the root part, buffered attachments, and subsequent attachments.

process-in-order
(Default) Processes the attachments and root part in the order that they appear in the original message. All parts are still processed in streaming mode even though only attachments after the root will be streamed from the network.

Guidelines

When streaming MIME messages, specifies the action to take when the root part is not the first part of the message. If the root part must be first (for example to do conformance checking) and the action is set to **process-in-order**, the attachments up to the root will be buffered.

This command is meaningful only when the value of either the **request-attachments** or **response-attachments** command is **streaming.**

Related Commands

request-attachments, **response-attachments**

soap-schema-url

Specifies the schema to validate SOAP messages.

Syntax

soap-schema-url *schema-url*

Parameters

schema-url
Specifies the URL of the schema file to validate that SOAP messages conform to the SOAP schema. Defaults to `store:///schemas/soap-envelope.xsd`.

Related Commands

request-type, **response-type**

Guidelines

When a Multi-Protocol Gateway is in SOAP mode, either on the request or response side, it validates incoming messages against a W3C Schema that defines the format of a SOAP message.

It is possible to customize which schema is used on a per-gateway basis by changing this property to accommodate nonstandard configurations or other special cases.

Examples

- Sets the URL of the SOAP schema to validate SOAP message formats to a file contained in the local file space denoted as `local://custom-soap-schema.xsd`.

```
# soap-schema-url local://custom-soap-schema.xsd
#
```

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

no ssl

Parameters

name Specifies the name of an existing SSL Proxy Profile. If not specified, the Multi-Protocol Gateway and server exchanges are accomplished over a nonsecure connection.

Guidelines

An SSL Proxy Profile specifies the SSL operational mode (client, server, or two-way) and identifies the cryptographic resources (key, certificates, and cipher lists) available to the SSL proxy.

The SSL Proxy Profile must have previously created with the **sslproxy** command.

Assignment of an SSL Proxy Profile to a Multi-Protocol Gateway is optional, unless the Backend URL begins with HTTPS.

Use the **no ssl** command to remove the SSL Proxy Profile assignment.

Examples

- Assigns the SSL1 SSL Proxy Profile to the current Multi-Protocol Gateway.

```
# ssl SSL1
#
```

- Removes the assignment of an SSL Proxy Profile.

```
# no ssl
#
```

stream-output-to-back

Determines whether or not the Multi-Protocol Gateway will begin sending output to the backend server before all processing of the message completes.

Syntax

stream-output-to-back {**buffer-until-verification** | **stream-until-infraction**}

Parameters

buffer-until-verification
(Default) Buffers submitted messages until all processing has been verified complete, and then the message is forwarded to the appropriate backend URL.

stream-until-infraction
Begins sending the message to the backend URL before all processing is complete, potentially increasing the speed. If an infraction is encountered, the gateway reverts to buffered behavior. Use this option when the XML Manager for this Gateway has streaming enabled to be certain that the appliance will stream messages end-to-end.

Related Commands

stream-output-to-front

Examples

- Changes the default to stream output to the backend server until an infraction is encountered.

```
# stream-until-infraction
#
```

stream-output-to-front

Determines whether or not the Multi-Protocol Gateway will begin sending output to the client before all processing of the message completes.

Syntax

stream-output-to-front {**buffer-until-verification** | **stream-until-infraction**}

Parameters

buffer-until-verification
(Default) Buffers submitted messages until all processing has been verified complete, and then the message is returned to the client.

stream-until-infraction
Begins sending the message to the client before all processing is complete, potentially increasing the speed. If an infraction is encountered, the gateway reverts to buffered behavior. Use this option when the XML Manager for this Multi-Protocol Gateway has streaming enabled to be certain that the appliance will stream messages end-to-end.

Related Commands

stream-output-to-back

Examples

- Changes the default to stream output to the client until an infraction is encountered.

```
# stream-until-infraction
#
```

stylepolicy

Assigns a Processing Policy.

Syntax

stylepolicy *name*

Parameters

name Specifies the name of an existing Processing Policy. If not specified, the Multi-Protocol Gateway uses the processing instructions, if any, in the XML document.

Guidelines

The Processing Policy is used in processing performed by the current Multi-Protocol Gateway. You need not specify a Processing Policy when configuring a Multi-Protocol Gateway.

Related Commands

xml-manager

Examples

- Assigns the `highRoad` Stylesheet Policy to the current Multi-Protocol Gateway.

```
# stylesheet-policy highRoad
#
```

suppress

Deletes standard HTTP header fields from the packet stream.

Syntax

suppress {**front** | **back**} *header*

no suppress {**front** | **back**} *header*

Parameters

front Indicates the packet stream between a Multi-Protocol Gateway and the HTTP client.

back Indicates the packet stream between a Multi-Protocol Gateway and the HTTP server.

header Identifies an HTTP header field as defined in sections 4.5, 5.3, 6.2, and 7.1 of RFC 2616.

Guidelines

Use the **no suppress** command to restore the standard HTTP header field to the packet stream.

Related Commands

host-rewriting, **inject**

Examples

- Deletes the `Authorization` HTTP header field from the packet stream directed to the HTTP server.

```
# suppress back Authorization
#
```

- Restores the `Authorization` HTTP header field to the packet stream directed to the HTTP server.

```
# no suppress back Authorization
#
```

---

type

Specifies the Multi-Protocol Gateway type.

Syntax

type {**dynamic-backend** | **static-backend**}

Parameters

dynamic-backend
Sets the gateway type to dynamic. The address of the target server is dynamically extracted from the client request using the `dp:set-target` or `dp:xset-target` extension elements.

static-backend
(Default) Sets the gateway type to static. Use the **backend-url** command to set a static backend destination.

Related Commands

backend-url

Examples

- Changes the Multi-Protocol Gateway type to dynamic backend.

```
# type dynamic-backend
#
```

urlrewrite-policy

Assigns a URL Rewrite Policy.

Syntax

urlrewrite-policy *name*

no urlrewrite-policy *name*

Parameters

name Specifies the name of a URL Rewrite Policy.

Guidelines

You need not specify a URL Rewrite Policy when configuring a Multi-Protocol Gateway.

Use the **no urlrewrite-policy** command to remove the URL Rewrite Policy assignment.

Related Commands

propagate-uri

Examples

- Assigns the `Rw1` URL Rewrite Policy to the current Multi-Protocol Gateway.

```
# urlrewrite-policy Rw1
#
```

- Removes the `Rw1` URL Rewrite Policy.

```
# no urlrewrite-policy Rw1
#
```

wsa-back-protocol

Specifies the Front Side Protocol Handler to receive asynchronous server responses and forward them to the original client.

Syntax

wsa-back-protocol *frontSideProtocolHandler*

Parameters

frontSideProtocolHandler
Specifies the name of an existing Front Side Protocol Handler.

Guidelines

The **wsa-back-protocol** command is relevant when the DataPower service provides asynchronous service (the **wsa-genstyle** command is **async**). In these topologies, this command specifies the Front Side Protocol Handler to receive the asynchronous response and forward that response to the original client.

This Front Side Protocol Handler can be overridden by the `var://context/__WSA_REQUEST/replyto` variable.

Related Commands

wsa-genstyle

wsa-default-faultto

Force the inclusion of the `FaultTo` element in Web Services Addressing (WS-Addressing) messages.

Syntax

wsa-default-faultto *faultURL*

Parameters

faultURL
Specifies the value of the `FaultTo` element.

Guidelines

The **wsa-default-faultto** command is relevant when the DataPower service provides service for WS-Addressing clients (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command ensures that all messages contain the WS-Addressing `FaultTo` element. This element identifies the recipient endpoint of fault messages.

Because the WS-Addressing specifications do not require the inclusion of the `FaultTo` element, the DataPower service might receive messages that do not contain a `FaultTo` element or that contain the element with no value.

When this happens, the DataPower service modifies the message to include a `FaultTo` element. This element contains the value specified by the *faultURL* argument.

If a default recipient endpoint of fault messages is not explicitly identified by this command, the DataPower service provides the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

Related Commands

wsa-mode

Examples

- Specifies `http://www.datapower.com/cs/internal/hdesk/` as the default recipient of `FaultTo` messages.

```
# wsa-default-faultto http://www.datapower.com/cs/internal/hdesk/
#
```

wsa-default-replyto

Force the inclusion of the `ReplyTo` element in Web Services Addressing (WS-Addressing) messages.

Syntax

wsa-default-replyto *replyURL*

Parameters

replyURL
Specifies the value of the `ReplyTo` element.

Guidelines

The **wsa-default-replyto** command is relevant when the DataPower service provides service for WS-Addressing clients (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command ensures that all messages contain the WS-Addressing `ReplyTo` element. This element identifies the recipient endpoint of a response message.

Because the WS-Addressing specifications do not require the inclusion of the `ReplyTo` element, the DataPower service might receive messages that do not contain a `ReplyTo` element or that contain the element without a value.

When this happens, the DataPower service modifies the message to include a `ReplyTo` element that contains the value specified by the *replyURL* argument.

If a default recipient endpoint of response messages is not explicitly identified by this command, the DataPower service provides the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

Related Commands

wsa-mode

Examples

- Specifies `http://www.customer.com/PO/inventoryReq/` as the default message recipient.

```
# wsa-default-replyto http://www.customer.com/PO/inventoryReq/
#
```

---

wsa-faultto-rewrite

Assigns or removes a URL Rewrite Policy that rewrites the contents of the Web Services Addressing (WS-Addressing) `FaultTo` element.

Syntax

Assign a URL Rewrite Policy
wsa-faultto-rewrite *urlRewritePolicy*

Removes a URL Rewrite Policy
no wsa-faultto-write

Parameters

urlRewritePolicy
Specifies the name of the URL Rewrite Policy.

Guidelines

The **wsa-faultto-write** command is relevant when the DataPower service provides service for WS-Addressing clients (the **wsa-mode** command is **wsa2sync** or

wsa2wsa). In these topologies, this command modifies the contents of an incoming `FaultTo` element. This element identifies the recipient endpoint of fault messages.

Related Commands

absolute-rewrite (URL Rewrite Policy), **urlrewrite** (Global), **wsa-mode**, **wsa-replyto-rewrite**, **wsa-to-rewrite**

Examples

- Assigns the `wsaErrorHandler` URL Rewrite Policy to modify the contents of the `FaultTo` element.

```
# wsa-faultto-rewrite wsaErrorHandler
#
```

- Removes the assigned URL Rewrite Policy.

```
# no wsa-faultto-rewrite
#
```

---

wsa-force

Forces the inclusion of Web Services Addressing (WS-Addressing) headers into incoming, traditionally-addressed messages.

Syntax

wsa-force {**on** | **off**}

Parameters

on (Default) Forces the inclusion of WS-Addressing headers.

off Retains the traditional addressing headers.

Guidelines

The **was-force** command is relevant when the DataPower service provides service to users of WS-Addressing and users or traditionally-addressed messages (the **wsa-mode** command is **wsa2wsa**, **wsa2sync**, or **sync2wsa**). In these topologies, the DataPower service generally handles a mix of messages that use the WS-Addressing format and the traditional format.

Use this command to ensure that all messages use WS-Addressing. By default, **wsa-force** is disabled. When disabled, the DataPower service supports of mix of addressing styles.

When enabled, the DataPower service converts traditionally-addressed messages to the WS-Addressing format by adding the `reply-to` and `fault-to` headers to the traditionally-addressed message.

The `reply-to` header will contain the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

The `fault-to` header will contain the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

These default values can be overridden with the **wsa-default-replyto** and **wsa-default-faultto** commands.

Related Commands

wsa-default-faultto, **wsa-default-replyto**, **wsa-strip-headers**

Examples

- Adds WS-Addressing headers to traditionally-addressed messages.

```
# wsa-force on
#
```

- Leaves traditionally-addressed message headers untouched.

```
# wsa-force off
#
```

or

```
# no wsa-force
#
```

---

wsa-genstyle

Specifies the request-response transmission model between the DataPower service and the target server.

Syntax

wsa-genstyle { **async** | **oob** | **sync** }

Parameters

async Identifies an asynchronous exchange pattern in which the server response is received over a different channel than the one used by the DataPower service to convey the client request.

oob Identifies an out-of-band exchange pattern in which the routing of the response to the original client is handled by the target server and does not pass through the DataPower service.

sync (Default) Identifies a synchronous exchange pattern in which the server response is received over the same channel used by the DataPower service to convey the client request.

Guidelines

If the request-response transmission model is **async**, use the **wsa-back-protocol** command to identify the Front Side Protocol Handler to convey asynchronous server responses to the original requesting clients. For the asynchronous model, use the **wsa-timeout** command to specify the maximum time allowed for a server response.

If the request-response transmission model is **oob**, ensure that the Web Server Proxy preserves explicit (non-anonymous), client-originated values for the `ReplyTo` and `FaultTo` elements and passes these values intact to the server.

Related Commands

wsa-back-protocol, **wsa-http-async-response-code**, **wsa-mode**, **wsa-timeout**

wsa-http-async-response-code

Specifies the HTTP response code to send to a client appliance before transmitting the actual asynchronous server response.

Syntax

wsa-http-async-response-code *responseCodeValue*

Parameters

responseCodeValue
Specifies the HTTP response code to close the original client channel. Use a value in the range of 200 through 599. The default is 204.

Guidelines

If the server response to an HTTP client request is asynchronous, the DataPower service must close the original HTTP channel with a valid response code. After the channel is closed, the DataPower service forwards the server-generated response or fault message to the client over a new channel.

Related Commands

wsa-genstyle

Examples

- Specifies an HTTP Response Code of 210 to close an open HTTP client channel.

```
# wsa-http-response-code 210
#
```

wsa-mode

Specifies the Web Services Addressing (WS-Addressing) support.

Syntax

wsa-mode {**sync2sync** | **sync2wsa** | **wsa2sync** | **wsa2wsa**}

Parameters

sync2sync
(Default) Disables WS-Addressing support. Both hosts (clients and servers) that access the DataPower service will use traditional addressing.

sync2wsa
Specifies that the DataPower service is mediating between hosts that employ traditional addressing and servers that support WS-Addressing.

wsa2sync
Specifies that the DataPower service is mediating between hosts that support WS-Addressing and servers that employ traditional addressing.

wsa2wsa
Specifies that the DataPower service is mediating between hosts and servers that support WS-Addressing.

Guidelines

The **wsa-mode** command specifies the WS-Addressing support that the DataPower service provides. The level of support is determined by the WS-Addressing capabilities of the associated clients and servers. Support for any particular level of WS-Addressing does not preclude simultaneous support for traditional addressing formats.

- When operating in **sync2wsa** mode, the DataPower service, under user control, can:
  - Insert WS-Addressing headers into the traditionally addressed original client request. The default behavior is to retain the original addressing format.
  - Strip the WS-Addressing headers from any server-generated response before forwarding the response to the original client. The default behavior is to strip the WS-Addressing headers.
  - Process synchronous or asynchronous server responses of either the `ReplyTo` (a standard response to a client request) or `FaultTo` (reporting an error condition) variety.

  A synchronous response is received over the same connection that carried the client request to the server.

  An asynchronous response is received over a different connection that carried the client request to the server, and requires the DataPower service to maintain state information associating the received response with an outstanding request.
- When operating in **wsa2sync** mode, the DataPower service, under user control, can:
  - Insert WS-Addressing headers into the traditionally addressed server response. The default behavior is to retain the original addressing format.
  - Strip the WS-Addressing headers from any client-generated request before forwarding the request to the target server. The default behavior is to strip the WS-Addressing headers.
  - Rewrite the contents of, or supply default values, for client-generated `ReplyTo` and `FaultTo` elements to specify the destinations of these response types.
  - Rewrite the contents of the client-generated `To` element to specify where the client request is routed.
  - Process synchronous or asynchronous server responses of either the `ReplyTo` (a standard response to a client request) or `FaultToA` (reporting an error condition) variety.

  A synchronous response is received over the same connection that carried the client request to the server.

  An asynchronous response is received over a different connection that carried the client request to the server, and requires the DataPower service to maintain state information associating the received response with an outstanding request.
- When operating in **wsa2wsa** mode, the DataPower service under user control, can:
  - Insert WS-Addressing headers into the traditionally addressed server response. The default behavior is to retain the original addressing format.
  - Strip the WS-Addressing headers from any client-generated request before forwarding the request to the target server. The default behavior is to strip the WS-Addressing headers.
  - Rewrite the contents of, or supply default values, for client-generated `ReplyTo` and `FaultTo` elements to specify the destinations of these response types.
  - Rewrite the contents of the client-generated `To` element to specify where the client request is routed.

– Support three response modes for either the ReplyTo (a standard response to a client request) or FaultTo (reporting an error condition) variety.

A synchronous response is received over the same connection that carried the client request to the server.

An asynchronous response is received over a different connection that carried the client request to the server, and requires the DataPower service to maintain state information associating the received response with an outstanding request.

An out-of-band response is handled by the target server and does not pass through the DataPower service. An out-of-band response requires explicit (non-anonymous) client-originated ReplyTo and FaultTo element values that are preserved by the DataPower service and passed to the server.

Related Commands

wsa-back-protocol, **wsa-force**, **wsa-genstyle**, **wsa-timeout**, **wsa-strip-headers**

Examples

- Specifies sync2wsa mode, indicating that the DataPower service is mediating between hosts that employ traditional addressing and servers that support WS-Addressing.

```
# wsa-mode sync2wsa
#
```

wsa-replyto-rewrite

Identifies the URL Rewrite Policy to rewrite the contents of the Web Services Addressing (WS-Addressing) ReplyTo element.

Syntax

wsa-replyto-rewrite *urlRewritePolicy*

no wsa-replyto-rewrite

Parameters

urlRewritePolicy
Specifies the name of the URL Rewrite Policy.

Guidelines

The **wsa-replyto-rewrite** command is relevant when the DataPower service provides service for WS-Addressing client users (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command modifies the contents of an incoming ReplyTo element. This element identifies the recipient endpoint of response messages.

Related Commands

absolute-rewrite, **urlrewrite**, **wsa-mode**, **wsa-faultto-rewrite**, **wsa-to-rewrite**

Examples

- Identifies wsaResponseHandler as the URL Rewrite Policy used to modify the contents of the ReplyTo element.

```
# wsa-replyto-rewrite wsaResponseHandler
#
```

- Removes the assignment of `wsaResponseHandler` as the URL Rewrite Policy used to modify the contents of the `ReplyTo` element.

```
# no wsa-replyto-rewrite
#
```

wsa-strip-headers

Removes all Web Services Addressing (WS-Addressing) headers from an incoming message before forwarding the message to the recipient.

Syntax

wsa-strip-headers {**on** | **off**}

Parameters

on (Default) Enables the deletion of WS-Addressing headers from an incoming message.

off Disables the deletion of WS-Addressing headers from an incoming message.

Guidelines

This command is relevant when the DataPower service is positioned between users of WS-Addressing and a nonusers; that is when the WS-Addressing mode, as specified by the **wsa-mode** command, is **wsa2sync** or **sync2wsa**.

Note: WS-Reliable Messaging requires the termination of WS-Addressing sequences. Changing the default value can break interoperability.

Related Commands

wsa-force, **wsa-mode**

Examples

- Changes the default state. Retains all WS-Addressing headers contained in incoming messages.

```
# wsa-strip-headers off
#
```

or

```
# no wsa-strip-headers
#
```

- Restores the default state. Deletes all WS-Addressing headers contained in incoming messages.

```
# wsa-strip-headers on
#
```

or

```
# wsa-strip-headers
#
```

wsa-timeout

Specifies the asynchronous timeout value.

Syntax

wsa-timeout *timerValue*

Parameters

timerValue
Specifies the maximum wait period in seconds. Use an integer in the range of 1 through 4000000. The default is 120.

Guidelines

The **wsa-timeout** command specifies the maximum period of time to wait for an asynchronous response, before abandoning the transaction.

This timeout value can be overridden by the `var://service/wsa/timeout` variable.

Related Commands

wsa-mode

Examples

- Specifies a maximum pause of 1 minute while waiting for an asynchronous response.

```
# wsa-timeout 60
#
```

wsa-to-rewrite

Assigns or removes a URL Rewrite Policy that rewrites the contents of the Web Services Addressing (WS-Addressing) `To` element.

Syntax

wsa-to-rewrite *urlRewritePolicy*

no wsa-to-rewrite

Parameters

urlRewritePolicy
Specifies the name of an existing URL Rewrite Policy.

Guidelines

The **wsa-to-rewrite** command modifies the contents of an incoming `To` element that identifies the message destination. This command is relevant when the DataPower service provides service for clients that support WS-Addressing formats. In these cases, the WS-Addressing mode, as specified by the **wsa-mode** command, is **wsa2sync** or **wsa2wsa**.

Related Commands

wsa-mode

wsrm

Enable or disables Web Services Reliable Messaging.

Syntax

wsrm {**on** | **off**}

Parameters

on Enables Reliable Messaging.

off (Default) Disables Reliable Messaging.

Related Commands

wsrm-aaapolicy, **wsrm-destination-accept-create-sequence**, **wsrm-destination-accept-offers**, **wsrm-destination-inorder**, **wsrm-destination-maximum-inorder-queue-length**, **wsrm-destination-maximum-sequences**, **wsrm-request-force**, **wsrm-response-force**, **wsrm-sequence-expiration**, **wsrm-source-back-acks-to**, **wsrm-source-exponential-backoff**, **wsrm-source-front-acks-to**, **wsrm-source-inactivity-close-interval**, **wsrm-source-make-offer**, **wsrm-source-maximum-queue-length**, **wsrm-source-maximum-sequences**, **wsrm-source-request-ack-count**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**, **wsrm-source-sequence-ssl**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

---

wsrm-aaapolicy

Assigns an AAA Policy.

Syntax

wsrm-aaapolicy *name*

Parameters

name Specifies the name of an existing AAA Policy.

Guidelines

Use the **wsrm-aaapolicy** command to assign an AAA Policy to perform authentication of incoming Reliable Messaging messages. This AAA Policy can be the same one that is used in later processing by the request or response rule. The results are cached, so it is not evaluated again.

While this is focused on protecting the Reliable Messaging control messages, such as `CreateSequence` and `TerminateSequence`, it is also run on incoming Reliable Messaging data messages, with a `Sequence` header. This prevents unauthorized clients from using system resources by issuing `CreateSequence` requests, or from disrupting existing Reliable Messaging sequences with `CloseSequence` or `TerminateSequence` messages, or from falsely acknowledging messages with `SequenceAcknowledgement` messages.

To create an AAA Policy, use the Global **aaapolicy** command.

Related Commands

aaapolicy (global), **wsrm**

wsrm-destination-accept-create-sequence

Indicates whether to accept incoming `CreateSequence` SOAP requests and create a Reliable Messaging destination when one is received.

Syntax

wsrm-destination-accept-create-sequence {**on** | **off**}

Parameters

on (Default) Enables this feature. If enabled, both the client and the server can use Reliable Messaging to send messages to this DataPower service.

off Disables this feature. If disabled, the client cannot use Reliable Messaging to communicate with this DataPower service. If disabled, the only way that a Reliable Messaging destination can be created on this DataPower service is when the Reliable Messaging source is configured to make offers. In this case an `Offer` and `Accept` can create a Reliable Messaging destination for the server to send Reliable Messaging messages to the client.

Related Commands

wsrm

wsrm-destination-accept-offers

Indicates whether to accept offers for two-way Reliable Messaging in `CreateSequence` SOAP requests.

Syntax

wsrm-destination-accept-offers { **on** | **off**}

Parameters

on Accepts two-way requests.

off (Default) Does not accept two-way requests.

Guidelines

The **wsrm-destination-accept-offers** command indicates whether to accept offers for two-way Reliable Messaging in `CreateSequence` SOAP requests. If the request includes an offer, the creation of a Reliable Messaging destination creates a Reliable Messaging source to send responses to the client.

Related Commands

wsrm, **wsrm-source-exponential-backoff**, **wsrm-source-inactivity-close-interval**, **wsrm-source-maximum-queue-length**, **wsrm-source-request-ack-count**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

wsrm-destination-inorder

Indicates whether to enable `InOrder` delivery assurance for Reliable Messaging destinations

Syntax

wsrm-destination-inorder {**on** | **off**}

Parameters

on Enables `InOrder` and `ExactlyOnce` delivery assurance.

off (Default) Enables `ExactlyOnce` delivery assurance only.

Guidelines

The **wsrm-destination-inorder** command indicates whether to enable `InOrder` delivery assurance for Reliable Messaging destinations in addition to the standard `ExactlyOnce` delivery assurance. No messages will be passed from the receive queue for further processing unless their sequence number as assigned by the client is one greater than the last one that was processed. `InOrder` delivery assurance increases memory and resource utilization by the Reliable Messaging destination.

Related Commands

wsrm, **wsrm-destination-maximum-inorder-queue-length**

---

wsrm-destination-maximum-inorder-queue-length

Specifies the maximum number of messages held in the queue.

Syntax

wsrm-destination-maximum-inorder-queue-length *numberOfMessages*

Parameters

numberOfMessages
Specifies the maximum number of messages beyond the gap. Use an integer in the range of 1 through 256. The default is 10.

Guidelines

The **wsrm-destination-maximum-inorder-queue-length** command specifies the maximum number of messages held in the Reliable Messaging queue beyond a gap in the received sequence numbers.

This property controls memory utilization.

Related Commands

wsrm, **wsrm-destination-inorder**

---

wsrm-destination-maximum-sequences

Sets a limit on the maximum number of simultaneously active sequences to Reliable Messaging destinations.

Syntax

wsrm-destination-maximum-sequences *maximumSequences*

Parameters

maximumSequences
Specifies the maximum number of simultaneous active sequences. The default is 400.

Guidelines

The **wsrm-destination-maximum-sequences** command sets a limit on the maximum number of simultaneously active sequences to Reliable Messaging destinations of this DataPower service. Attempts by clients to create sequences in excess of this limit result in a SOAP Faults. This property controls memory resource utilization.

Related Commands

wsrm

---

wsrm-request-force

Indicates whether to require Reliable Messaging for all SOAP messages that request rules process.

Syntax

wsrm-request-force {**on** | **off**}

Parameters

on Requires Reliable Messaging for all requests.

off (Default) Does not require Reliable Messaging for all requests.

Guidelines

The xxx command indicates whether to require the use of Reliable Messaging for all SOAP messages that request rules process. The client must establish a sequence with a `CreateSequence` SOAP call and must include a `Sequence` in each SOAP header. Any SOAP message without a `Sequence` results in a SOAP fault.

Related Commands

wsrm

---

wsrm-response-force

Indicates whether to require Reliable Messaging for all SOAP messages that response rules process.

Syntax

wsrm-response-force {**on** | **off**}

Parameters

on Requires Reliable Messaging for all responses.

off (Default) Does not require Reliable Messaging for all responses.

Guidelines

The **wsrm-response-force** command indicates whether to require the use of Reliable Messaging for all SOAP messages that response rules process. Any SOAP message without a `Sequence` results in a SOAP fault.

Note: When WS-Addressing is in use, SOAP messages without a WS-Addressing `RelatesTo` SOAP Header are processed by the request rule, not the response rule, even if the message come from the backend server.

Related Commands

wsrm

---

wsrm-sequence-expiration

Sets the target expiration interval in seconds for all Reliable Messaging sequences.

Syntax

wsrm-sequence-expiration *lifetime*

Parameters

lifetime Specifies the lifetime in seconds. The default is 3600.

Guidelines

If an incoming `CreateSequence` SOAP message has an `Expires`lifetime that is longer than this value, the value in the `SequenceResponse` SOAP message is reduced to this value. The same process applies to the `Expires`lifetime in any accepted `Offer` in an incoming `CreateSequence` and for the requested `Expires` value in any `CreateSequence` SOAP call that is made to the client or server from a Reliable Messaging source. This implementation never requests or accepts a non-expiring sequence (a value of `PT0S` that represents zero seconds).

Related Commands

wsrm

---

wsrm-source-back-acks-to

Specifies the name of the Front Side Protocol Handler to receive responses from the server.

Syntax

wsrm-source-back-acks-to *handler*

Parameters

handler

Specifies the name of an existing Front Side Protocol Handler.

Guidelines

The **wsrm-source-back-acks-to** command identifies the Front Side Protocol Handler to receive the asynchronous Reliable Messaging `SequenceAcknowledgement`

SOAP responses from the server. The Front Side Protocol Handler must be associated with the same DataPower service where the corresponding Reliable Messaging sequence is occurring.

This property controls whether the backside Reliable Messaging source uses a unique URL to receive asynchronous `Acks` from the server Reliable Messaging destination, or whether `Acks` are sent synchronously in future responses to the backside server.

- With a specified Front Side Protocol Handler and the response process causes a `CreateSequence` SOAP message to be sent, the `AcksTo` element of the `CreateSequence` SOAP message will be set to the URL that is specified in back `AcksTo`.
- Without a Front Side Protocol Handler, the `AcksTo` element has the value `http://www.w3.org/2005/08/addressing/anonymous`, which indicates synchronous `Acks`.

Related Commands

wsrm

---

wsrm-source-exponential-backoff

Indicates whether to use the exponential back off.

Syntax

wsrm-source-exponential-backoff {**on** | **off**}

Parameters

on (Default) Uses the exponential back off to increase the interval between retransmissions. The value of the **wsrm-source-retransmission-interval** command sets with the initial timeout.

off Does not use the exponential back off to increase the interval between retransmissions.

Guidelines

The **wsrm-source-exponential-backoff** command indicates whether to use the exponential back off to increase the interval between retransmissions on unacknowledged messages by a Reliable Messaging source.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**, **wsrm-source-retransmission-interval**

---

wsrm-source-front-acks-to

Identifies the Front Side Protocol Handler to receive response for the client.

Syntax

wsrm-source-front-acks-to *handler*

Parameters

handler
Specifies the name of an existing Front Side Protocol Handler.

Guidelines

The **wsrm-source-front-acks-to** command identifies the Front Side Protocol Handler to receive the asynchronous Reliable Messaging `SequenceAcknowledgement` SOAP responses from the client. The Front Side Protocol Handler must be associated with the same DataPower service where the corresponding Reliable Messaging sequence is occurring.

This property controls whether a front-side Reliable Messaging source uses a unique URL to receive asynchronous `Acks` from the client Reliable Messaging destination or whether `Acks` are sent synchronously in future requests to the front-side client.

- With a specified Front Side Protocol Handler and the client includes an `Offer` in a `CreateSequence` SOAP message sent due to response processing, there will be a non-anonymous URL specified in the `AcksTo` element of the `Accept` element of the `CreateSequenceResponse` SOAP reply.
- With a specified Front Side Protocol Handler and the front-side sends a `CreateSequence` SOAP message to establish a reliable back channel, there will be a non-anonymous URL specified in the `AcksTo` element of the `CreateSequence` SOAP request.
- Without a Front Side Protocol Handler, the `AcksTo` elements has the value `http://www.w3.org/2005/08/addressing/anonymous`, which indicates synchronous `Acks`.

Related Commands

wsrm

---

wsrm-source-inactivity-close-interval

Specifies the duration to wait before closing the sequence.

Syntax

wsrm-source-inactivity-close-interval *duration*

Parameters

duration
Specifies the duration to wait in seconds. Use an integer in the range of 1 through 3600. The default is 3.

Guidelines

The **wsrm-source-inactivity-close-interval** command specifies the duration in second that a Reliable Messaging source waits for an another message to be sent before closing the sequence by sending a `CloseSequence` SOAP message.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

wsrm-source-make-offer

Indicates whether to include an offer for two-way.

Syntax

wsrm-source-make-offer {**on** | **off**}

Parameters

on Include an offer.

off (Default) Does not include an offer.

Guidelines

The **wsrm-source-make-offer** command indicates whether to include an offer for two-way Reliable Messaging in `CreateSequence` SOAP requests that are made as the result of request processing. Including an offer can result in the creation of a Reliable Messaging destination for the server to send responses on when the DataPower service creates a Reliable Messaging source to send requests to the server. If the server does not accept the offer, DataPower server does not create a Reliable Messaging destination.

Related Commands

wsrm, **wsrm-source-request-create-sequence**

wsrm-source-maximum-queue-length

Specifies the maximum number of messages held in the queue.

Syntax

wsrm-source-maximum-queue-length *numberOfMessages*

Parameters

numberOfMessages
Specifies the size of the queue in number of messages. Use an integer in the range of 1 through 256. The default is 30.

Guidelines

The **wsrm-source-maximum-queue-length** command specifies the maximum number of messages held in the Reliable Messaging queue while waiting for `Ack` messages. This property controls memory utilization.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

wsrm-source-maximum-sequences

Sets the limit of simultaneous active sequences.

Syntax

wsrm-source-maximum-sequences *limit*

Parameters

limit Specifies the number of simultaneous active sequence. Use an integer in the range of 1 through 2048. The default is 400.

Guidelines

The **wsrm-source-maximum-sequences** command sets a limit on the maximum number of simultaneously active sequences from Reliable Messaging sources of this DataPower server. Each remote Reliable Messaging destination endpoint reference (URL) requires one sequence. Transactions that request the creation of sequences in excess of this limit result in a SOAP Fault. This property controls memory resource utilization.

Related Commands

wsrm

---

wsrm-source-request-ack-count

Specifies the number of messages to send before requesting acknowledgement.

Syntax

wsrm-source-request-ack-count *numberOfMessages*

Parameters

numberOfMessages
Use an integer in the range of 1 through 256. The default is 1.

Guidelines

The **wsrm-source-request-ack-count** command specifies the number of messages that the a Reliable Messaging source sends before including the `AckRequested` SOAP header to request an acknowledgement.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

---

wsrm-source-request-create-sequence

Indicates whether to create a source from the backend to the server.

Syntax

wsrm-source-request-create-sequence {**on** | **off**}

Parameters

on Creates a Reliable Messaging source.

off (Default) Does not create a Reliable Messaging source.

Guidelines

The **wsrm-source-request-create-sequence** command indicates whether to create a Reliable Messaging source from the backend to the server when there is SOAP data

to sent to the server and when there is no Reliable Messaging source that was created by a `MakeOffer` from the server. The Reliable Messaging source is created by sending a `CreateSequence` SOAP request to the server address.

Related Commands

wsrm, **wsrm-source-exponential-backoff**, **wsrm-source-inactivity-close-interval**, **wsrm-source-make-offer**, **wsrm-source-maximum-queue-length**, **wsrm-source-request-ack-count**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

wsrm-source-response-create-sequence

Indicates whether to create a source from the front side to the client.

Syntax

wsrm-source-response-create-sequence {**on** | **off**}

Parameters

on Creates a Reliable Messaging source.

off (Default) Does not create a Reliable Messaging source.

Guidelines

When the WS-Addressing mode as defined by the **wsa-mode** command is **wsa2sync** or **wsa2wsa**, the **wsrm-source-response-create-sequence** command indicates whether to create a Reliable Messaging source from the front side to the client when there is SOAP data to send to the client and there is no Reliable Messaging source that was created by a `MakeOffer` from the client by sending a `CreateSequence` SOAP request to the WS-Addressing `ReplyTo` address.

Related Commands

wsa-mode, **wsrm**, **wsrm-source-exponential-backoff**, **wsrm-source-inactivity-close-interval**, **wsrm-source-maximum-queue-length**, **wsrm-source-request-ack-count**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

wsrm-source-retransmission-interval

Specifies the duration that a source waits.

Syntax

wsrm-source-retransmission-interval *interval*

Parameters

interval

Specifies the duration in milliseconds. Use an integer in the range of 10 through 60000. The default is 2000.

Guidelines

The **wsrm-source-retransmission-interval** command specifies the duration in milliseconds that a Reliable Messaging source waits for an `Ack` before retransmitting the message. This property also applies to the retransmission of the `CreateSequence` SOAP message.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-exponential-backoff**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

---

wsrm-source-retransmit-count

Specifies the number of times to retransmit a message.

Syntax

wsrm-source-retransmit-count *count*

Parameters

count Specifies the number of retransmissions. Use and integer in the range of 1 through 256. The default is 4.

Guidelines

The **wsrm-source-retransmit-count** command specifies the number of times a Reliable Messaging source retransmits a message before declaring a failure.

This command also controls the retransmission of `CreateSequence` requests.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

---

wsrm-source-sequence-ssl

Indicates whether to use an SSL session binding to protect sequence lifecycle messages.

Syntax

wsrm-source-sequence-ssl {**on** | **off**}

Parameters

on Uses an SSL session binding.

off (Default) Does not use an SSL session binding.

Guidelines

All Reliable Messaging control messages and sequence messages are bound to the original SSL/TLS session that is created by the Reliable Messaging source to transmit the `CreateSequence` control message. Sequence messages that are received by the Reliable Messaging destination with the correct identifier but on a different SSL/TLS session are rejected.

The lifetime of a SSL/TLS protected sequence is bound by the lifetime of the SSL/TLS session this is used to protect that sequence.

Related Commands

wsrm

xml-manager

Assigns an XML Manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of the XML manager to be assigned to the Multi-Protocol Gateway.

Guidelines

An XML manager obtains and controls resources required by the Multi-Protocol Gateway. You do not need to change it. If you want to use an XML Manager with user-specific characteristics, use the Global **xml-manager** command to create a new Manager. Then use this command to associate it with the current Multi-Protocol Gateway.

Related Commands

stylesheet-policy xml-manager (Global)

Examples

- Assigns the `mgr1` XML Manager to the current Multi-Protocol Gateway.

```
# xml-manager mgr1
#
```

Chapter 55. Network Settings configuration mode

This chapter provides an alphabetic listing of commands that are available in Network Settings configuration mode.

To enter this configuration mode, use the Global **network** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Network Settings configuration mode.

arp-interval

Sets the time interval between ARP retries.

Syntax

arp-interval *interval*

Parameters

interval

Specifies the amount of time in milliseconds to wait before retrying a failed ARP request. Use an integer in the range of 10 through 5000. The default is 50.

Related Commands

arp-retries

Examples

- Sets the arp interval to 100 milliseconds.

```
# arp-interval 100
#
```

arp-retries

Sets the number of times the networking system retries a failed ARP request.

Syntax

arp-retries *retries*

Parameters

retries Specifies the number of times to retry a failed ARP request. Use an integer in the range from 1 through 64. The default is 8.

Related Commands

arp-interval

Examples

- Sets the ARP retry limit to 5.

```
# arp-retries 5
#
```

destination-routing

Controls the behavior of destination-based routing.

Syntax

destination-routing {**on** | **off**}

Parameters

on Selects the interface based on the best path to the client, irrespective of the service or receiving interface. Best path is determined by static routes bound to the available interfaces.

Note: Destination-based routing is for backward compatibility only. Enable destination-based routing only if an upgrade disables existing connectivity.

off (Default) Selects the interface based on the interface that is bound to the address of the service that generated the response. If the service is bound to a single address, responses are routed using the interface that is assigned to that address. If the service is bound to more than one address (a configuration of 0.0.0.0), responses are routed using the interface that received the original client request (not the interface that is bound to the service that generated the response).

Related Commands

relax-interface-isolation

Examples

- Ensures that outbound packets originate from an interface bound to an address of the service that created the packet.

```
# destination-routing off
#
```

disable-interface-isolation

Controls interface isolation.

Syntax

disable-interface-isolation {**on** | **off**}

Parameters

on Enables interface-isolation.

off (Default) Disables interface-isolation.

Guidelines

By default the appliance will refuse to accept a packet on an interface other than the one bound to the destination address of the packet. Use the **disable-interface-isolation** command to disable that behavior and allow any interface on the same subnet to accept the packet.

As a security policy, the interface receiving a network packet must also be configured with the IP address that is the destination address of the packet. Enabling this option relaxes that restriction.

Related Commands

relax-interface-isolation

Examples

- Allows any interface on the same subnet to accept a packet.

```
# disable-interface-isolation on
#
```

ecn-disable

Turn on or off ECN-capable TCP sessions.

Syntax

ecn-disable {**on** | **off**}

Parameters

on Stops the generation of ECN-capable TCP sessions.

off (Default) Generates ECN-capable TCP sessions.

Examples

- Stops the networking system from generating ECN-enabled TCP sessions.

```
# ecn-disable on
#
```

icmp-disable

Disables the generation of a specific Internet Control Message Protocol (ICMP) reply message.

Syntax

icmp-disable {**addressmask-reply** | **echo-reply** | **info-reply** | **timestamp-reply**}

no icmp-disable

Parameters

addressmask-reply, **echo-reply**, **info-reply**, and **timestamp-reply**
Specifies the target ICMP reply type to disable.

Guidelines

By default, the appliance replies to the corresponding ICMP requests.

Use the **no icmp-disable** command to enable the generation of a specific ICMP reply.

Related Commands

network

Examples

- Disables ICMP echo message (`ping`) replies.

```
# icmp-disable echo-reply
#
```

- Enables `ping` replies, which restores the default state.

```
# no icmp-disable echo-reply
#
```

relax-interface-isolation

Relaxes the restriction on interface isolation.

Syntax

relax-interface-isolation {**on** | **off**}

Parameters

on (Default) Accepts a packet on an interface other than the one bound to the destination address of the packet.

off Allows only the interface bound to the destination address to accept the packet.

Guidelines

As a security policy, the interface that receives a network packet must also be configured with the IP address that is the destination address of the packet. Enabling this option relaxes that restriction. Relax interface isolation, if destination-routing is enabled.

Related Commands

destination-routing, **disable-interface-isolation**

Examples

- Allows only the interface bound to the destination address to accept a packet.

```
# relax-interface-isolation off
#
```

tcp-retries

Sets the number of times the local system sends a failed TCP SYN request.

Syntax

tcp-retries *retries*

Parameters

retries Specifies the number of times the local system attempt send a TCP SYN that receives no response. Use an integer in the range of 1 through 32. The default is 5.

Examples

- Sets the retry limit to 10.

```
# tcp-retries 10
#
```

Chapter 56. NFS Client Settings configuration mode

This chapter provides an alphabetic listing of all commands that are available in NFS Client Settings configuration mode.

To enter this configuration mode, use the Global **nfs-client** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in NFS Client Settings configuration mode.

kerberos-keytab

Assigns a keytab file for Kerberos authentication.

Syntax

kerberos-keytab *name*

Parameters

name Specifies the name of an existing Kerberos keytab object.

Guidelines

The **kerberos-keytab** command is meaningful only when the authentication method for NFS mount uses Kerberos. A keytab (or key table) is an unencrypted file that contains a list of Kerberos principals and their passwords.

Use the Crypto **kerberos-keytab** command to create a Kerberos keytab object.

Related Commands

authenticate (NFS Dynamic Mounts), **authenticate** (NFS Static Mounts), **kerberos-keytab** (Crypto)

mount-refresh-time

Specifies the interval between NFS mount maintenance rounds.

Syntax

mount-refresh-time *interval*

Parameters

interval

Specifies the interval in seconds between mount maintenance rounds. Use an integer in the range of 1 through 1000. The default is 10.

Guidelines

Each NFS mount maintenance round checks all existing NFS mounts (both dynamic and static), and retries any NFS mount that is not currently up.

Decreasing the interval lessens the chance that a transaction will time out while waiting for an NFS file open operation to fail because the NFS server is down or unreachable. Increasing the interval reduced local and NFS server overhead from mount checking.

Related Commands

mount-timeout (NFS Dynamic Mounts), **nfs-dynamic-mounts** (NFS Dynamic Mounts), **nfs-static-mount** (NFS Static Mounts), **show NFS Client Settings**

Examples

- Enters NFS Client Settings configuration mode to specify a 15-second interval between NFS mount maintenance rounds, and return to Global configuration mode after saving configuration changes.

```
# NFS Client Settings
Modify NFS Client Settings configuration
# mount-refresh-time 15
# exit
#
```

Chapter 57. NFS Dynamic Mounts configuration mode

This chapter provides an alphabetic listing of all commands that are available in NFS Dynamic Mounts configuration mode.

To enter this configuration mode, use the global **nfs-dynamic-mounts** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in NFS Dynamic Mounts configuration mode.

authenticate

Specifies an authentication method.

Syntax

authenticate {**AUTH_SYS** | **krb** | **krb5i** | **krb5p**}

Parameters

AUTH_SYS
(Default) Specifies the original NFS schema.

krb Specifies the authentication version to use on the Kerberos credentials that are stored on the appliance.

krb5i Specifies the authentication version to use on the Kerberos credentials that are stored on the appliance. This authentication method includes a secure hash function to protect the data from being changed by the network.

krb5p Specifies the authentication version to use on the Kerberos credentials that are stored on the appliance. This authentication method includes a secure hash function to protect the data from being changed by the network, and to protect the data encryption to prevent data from being read or changed by the network.

Guidelines

Use the **AUTH_SYS** authentication method when using all versions of NFS.

Use the **krb**, **krb5i**, or **krb5p** Kerberos authentication method when using NFS version 4.

If authenticating with Kerberos, ensure that a keytab is defined in the NFS client.

Related Commands

version, **kerberos-keytab** (Crypto)

inactivity-timeout

Specifies the time period before an inactive mount is unmounted.

Syntax

inactivity-timeout *seconds*

Parameters

seconds

Specifies the number of seconds an idle NFS mount, that is a mount with no file read-write activity, is maintained before the file system is unmounted. The default is 900. A value of 0 indicates that the NFS mount is never unmounted.

Related Commands

mount-timeout

---

mount-timeout

Specifies the maximum time the appliance attempts to complete an NFS mount before abandoning the effort.

Syntax

mount-timeout *seconds*

Parameters

seconds

Specifies the maximum period of time, expressed in seconds, that the appliance allows for the completion of an NFS mount. Use an integer in the range of 10 through 240. The default is 30.

Guidelines

Failure to complete the NFS mount process within the period specified by this timer results in a file open error and the cancellation of the NFS mount process.

This timer should be set to a value greater than the interval between NFS maintenance rounds, as defined by the mount-refresh-time command.

Related Commands

inactivity-timeout, **mount-refresh-time**

---

read-only

Specifies the NFS mount type.

Syntax

read-only {**on** | **off**}

Parameters

on Indicates that the mount allows only read transactions.

off Indicates that the mount allows read and write transactions.

Guidelines

Use the **read-only** command to specify the mount type as read-only. This setting allows only file read operations on NFS mounts. By default, NFS mounts can read transactions and write transactions.

retrans

Specifies the maximum number of RPC minor time outs to allow before the transaction fails.

Syntax

retrans *count*

Parameters

count Specifies the maximum number of RPC minor time outs to tolerate before an NFS transaction is abandoned and an NFS failure is declared. Use an integer in the range of 1 through 60. The default is 3.

Guidelines

Used in conjunction with the **timeo** command to specify behavior in response to RPC minor time outs and subsequent retransmission attempts.

The **retrans** command specifies the number of RPC minor time outs that are tolerated per NFS transaction before an NFS read or write error is declared. The **timeo** command provides a base value to determine the interval between the RPC time out and the subsequent retransmission attempt.

For example, assuming default values (3 for **retrans**, and 0.7 seconds for **timeo**), RPC time outs are dealt with as follows.

1. In response to the first RPC time out, the appliance waits 0.7 seconds and then retransmits.
2. In response to the second RPC time out, the appliance doubles the initial time out value to 1.4 seconds and then retransmits.
3. In response to the third RPC time out, the appliance doubles the previously used time out value to 2.8 seconds and then retransmits.
4. In response to the fourth time out, which is greater than the value specified by the **retrans** command, the appliance declares an NFS read or write error.

The interval between RPC time outs and a subsequent retransmission will never exceed 6 seconds.

Related Commands

timeo

rsize

Specifies the size of each read operation.

Syntax

rsize *size*

Parameters

size Specifies the number of bytes in each NFS read operation. Use an integer in the range of 1024 through 32769. The default is 4096.

Guidelines

Operations greater than 8192 bytes should only be used with TCP as the transport-layer protocol.

Related Commands

read-only, **transport**, **wsize**

---

timeo

Specifies the initial interval between an RPC minor time out and a subsequent retransmission attempt.

Syntax

timeo *seconds*

Parameters

seconds

Specifies the interval in tenths of a second between an initial RPC time out and the subsequent retransmission by the appliance. Use an integer in the range of 1 through 60. The default is 7.

Guidelines

Used in conjunction with the **retrans** command to specify behavior in response to RPC minor time outs and subsequent retransmission attempts.

The **retrans** command specifies the number of RPC minor time outs that are tolerated per NFS transaction before an NFS read or write error is declared. The **timeo** command provides a base value to determine the interval between the RPC time out and the subsequent retransmission attempt.

For example, assuming default values (3 for **retrans**, and 0.7 seconds for **timeo**), RPC time outs are dealt with as follows.

1. In response to the first RPC time out, the appliance waits 0.7 seconds and then retransmits.
2. In response to the second RPC time out, the appliance doubles the initial time out value to 1.4 seconds and then retransmits.
3. In response to the third RPC time out, the appliance doubles the previously used time out value to 2.8 seconds and then retransmits.
4. In response to the fourth time out, which is greater than the value specified by the **retrans** command, the appliance declares an NFS read or write error.

The interval between RPC time outs and a subsequent retransmission will never exceed 6 seconds.

Related Commands

retrans

transport

Identifies the preferred transport-layer protocol.

Syntax

transport {**tcp** | **udp**}

Parameters

tcp (Default) Identifies TCP as the protocol

udp identifies UDP as the protocol

Guidelines

The **transport** command specifies the preferred transport-layer protocol to use, if available. Use the TCP protocol to perform read or write transactions larger than 8192 bytes.

- For NFS version 2 or version 3, select the transport protocol to use when initiating the mount. If TCP is selected and it is not available on the NFS server, UDP will be used instead.
- For NFS version 4, this property is ignored. NFS version 4 only supports TCP.

Related Commands

rsize, **wsize**

version

Identifies the preferred protocol version.

Syntax

version {**2** | **3** | **4**}

Parameters

2 Specifies NFS version 2

3 (Default) Specifies NFS version 3

4 Specifies NFS version 4

Guidelines

The **version** command specifies the preferred NFS protocol version. If the version is 3, but the server only implements version 2, the client falls back to version 2. If the version is 4, there is no fallback.

wsize

Specifies the size of each write operation.

Syntax

wsize *size*

Parameters

size Specifies the number of bytes in each NFS write transaction. Use an integer in the range of 1024 through 32769. The default is 4096.

Guidelines

Operations greater than 8192 bytes should only be used with TCP as the transport-layer protocol.

Related Commands

rsize, **transport**

Examples

- Enters NFS Dynamic Mounts configuration mode. Sets the mount time out to 1 minute. Sets the mount inactivity timer to 3 minutes. Sets the read and write request sizes to 8192 bytes. Sets the maximum number of RPC time outs per NFS transaction to 4. Sets the initial retransmission interval to 0.5 seconds. Default values are retained for the transport-layer protocol (TCP), the NFS version (3), and the mount type (read-write).

```
# nfs-dynamic-mounts
Modify NFS Dynamic Mounts configuration
# mount-timeout 60
# inactivity-timeout 180
# rsize 8192
# wsize 8192
# retrans 4
# timeo 5
# exit
#
```

Chapter 58. NFS Poller Front Side Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in NFS Poller Front Side Handler configuration mode.

To enter this configuration mode, use the Global **source-nfs-poller** command. While in this mode, define the client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

delay-time

Specifies the time period between polling intervals.

Syntax

delay-time *interval*

Parameters

interval
Specifies the number of milliseconds after the completion of one poll interval and the start of the next interval. Use an integer in the range 25 through 100000. The default is 60000 (1 minute).

Guidelines

The **delay-time** command specifies the number of milliseconds to wait after the completion of one poll before starting the next interval. This interval is not the polling interval. It is the delay between polling intervals.

error-delete

Indicates whether to delete a file after a processing failure.

Syntax

error-delete {**on** | **off**}

Parameters

on Deletes the input or processing renamed file if it could not be processed.

off (Default) Does not delete the input or processing renamed file if it could not be processed.

Guidelines

The **error-delete** command indicates whether the input or processing renamed file should be deleted when it could not be processed.

error-rename-pattern

Specifies the rename pattern when a file could not be processed.

Syntax

error-rename-pattern *pattern*

Parameters

pattern Defines a PCRE that defines the rename pattern.

Guidelines

The **error-rename-pattern** command specifies the PCRE to rename a file when it could not be processed.

This command is relevant when **error-delete** is **off**. Otherwise, it is ignored.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

error-delete

match-pattern

Specifies the file name pattern for the search criteria.

Syntax

match-pattern *pattern*

Parameters

pattern Defines a PCRE to use as the match pattern to search the contents of the directory.

Guidelines

The **match-pattern** command specifies the PCRE used to match the contents of the directory being polled. If there is file-renaming or there is a response, this PCRE must create PCRE back references using `()` pairs.

For example, if input files are `NNNNNN.input` the **match-pattern** would be

`([0-9] {6})\.input$`.

PCRE documentation is available at the following web site:

http://www.pcre.org

processing-rename-pattern

Specifies the rename pattern when a file could be processed.

Syntax

processing-rename-pattern *pattern*

Parameters

pattern

Defines a PCRE that defines the rename pattern.

Guidelines

The **processing-rename-pattern** command specifies the PCRE to rename a file that is being processed. This functionality allows multiple pollers to poll the same directory with the same match pattern. There is no lack of atomicity if the rename operation on the server is atomic. The poller that succeeds in renaming the input file will proceed to process the file. Any other poller that tries to rename the file at the same time will fail to rename the file and will proceed to try the next file that matches the specified match pattern.

To ensure uniqueness, the resulting file name will be in the following format:

filename.serial.domain.poller.timestamp

filename
The file name for the renamed input file.

serial The serial number of the configured DataPower appliance.

domain The domain of the polling object.

poller The name of the polling object.

timestamp
The timestamp.

Note: File renaming cannot be used with an FTP server that supports only 8.3 file names.

For example, if the input files are `NNNNNN.input` and you want to rename them to `NNNNNN.processing`, the **match-pattern** would be `([0-9]{6})\.input$` and the rename pattern would be `$1.processing`. The resultant file name on the server would be:

`NNNNNN.processing.`*`serial.domain .poller.timestamp`*

Note: If no processing rename pattern is configured, the file will still be renamed. The only difference is that the *filename* portion of the resulting file is the name of the original input file, not the renamed input file.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

match-pattern

---

processing-seize-pattern

Specifies the rename pattern to search for files that are being processed.

Syntax

processing-seize-pattern *pattern*

Parameters

pattern Defines the PCRE to use as the match pattern to search for files that are being processed.

Guidelines

The **processing-seize-pattern** command specifies the PCRE to find files that were renamed to indicate that they are in the ″being processed″ state but the processing was never completed.

The processing seize pattern contains three phrases that must be in `\( \)` pairs. The first phrase is the base file name that includes the configured processing suffix. The second phrase is the host name. The third phrase is the timestamp.

PCRE documentation is available at the following web site:

http://www.pcre.org

---

processing-seize-timeout

Specifies the time to wait before processing a file that is already in the processing state.

Syntax

processing-seize-timeout *timeout*

Parameters

timeout
Specifies the number of seconds to wait before processing a file that is already in the processing state. Use an integer in the range of 0 through 1000. The default is 0.

Guidelines

The **processing-seize-timeout** command allows failure handling of a poller when multiple data routers are polling the same target. If another data router renames a file and does not process (and rename or delete) it within the specified number of seconds, this system will try to take over processing.

This system will try to take over processing when all of the following conditions are met when compared to the processing seize pattern:

The base file name (first match phrase) is the base file name of the processing seize pattern

The host name (second match phrase) is not the name of this system

The timestamp (third match phrase) is further in the past than the wait time specified by this command

When these conditions are met, this system renames the file (with its host name and a fresh timestamp) and locally processes the file. This processing assumes that the rename succeeded.

Related Commands

processing-seize-pattern

result

Indicates whether to create a response file after processing an input file.

Syntax

result { **on** | **off**}

Parameters

on (Default) Creates a result file.

off Does not create a result file.

Guidelines

The **result** command indicates whether the appliance should create a response file after successfully processing an input file.

result-name-pattern

Specifies the match pattern to build the name of the response file.

Syntax

result-name-pattern *pattern*

Parameters

pattern Defines the PCRE to use as the match pattern to build the name of the response file.

Guidelines

The **result-name-pattern** command specifies the PCRE to use as the match pattern to build the name of the result file. This PCRE will normally have a back reference to the base input file name. For instance, in input files are `NNNNNN.input` and the desired result file name is `NNNNNN.result`, then the match pattern would be `([0-9]{6})\.input$` and the result pattern would be `$1.result`.

Some servers might allow this pattern to indicate a path that puts the file in a different director, if it allows cross-directory renames. For instance, the match pattern would be `(.*)` and the result pattern would be `../result/$1`.

This command is relevant when **result** is `on`. Otherwise, it is ignored.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

result

---

success-delete

Indicates whether the input file is deleted after successful processing.

Syntax

success-delete {**on** | **off**}

Parameters

on Deletes the input file.

off (Default) Does not delete the input file.

Guidelines

The **success-delete** command indicates whether the input (or processing renamed) files should be deleted after successful processing.

---

success-rename-pattern

Specifies the rename pattern for input files on success.

Syntax

success-rename-pattern *pattern*

Parameters

pattern Defines the PCRE to use as the match pattern to rename the input file on success.

Guidelines

The **success-rename-pattern** command specifies the PCRE to rename the input file on success. This PCRE will normally have a back reference for the base input file name. For instance, in input files are `NNNNNN.input` and the desired result file name is `NNNNNN.processed`, then the match pattern would be `([0-9]{6})\.input$` and the result pattern would be `$1.processed`.

Some servers might allow this pattern to indicate a path that puts the file in a different director, if it allows cross-directory renames. For instance, the match pattern would be `(.*)` and the result pattern would be `../processed/$1`.

This command is relevant when **success-delete** is `off`. Otherwise, it is ignored.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

success-delete

target-dir

Specifies the directory to poll.

Syntax

target-dir *directory*

Parameters

directory
Specifies the directory to poll.

Guidelines

The **target-dir** command specifies a directory to poll. The path must end in a slash, which denoting a directory. For example:

```
dpnfs://static-mount-name/path/
```

Do not configure one NFS poller to point at a host name that is the virtual name of a load balancer group. This configuration is not the correct way to poll multiple hosts. To poll multiple hosts, use the same Multi-Protocol Gateway and configure one NFS poller object for each real host.

xml-manager

Assigns an XML Manager.

Syntax

xml-manager*name*

Parameters

name Specifies the name of the XML Manager.

Chapter 59. NFS Static Mounts configuration mode

This chapter provides an alphabetic listing of all commands that are available in NFS Static Mounts configuration mode.

To enter this configuration mode, use the global **nfs-static-mount** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in NFS Static Mounts configuration mode.

authenticate

Specifies an authentication method.

Syntax

authenticate {**AUTH_SYS** | **krb** | **krb5i** | **krb5p**}

Parameters

AUTH_SYS
(Default) Specifies the original NFS Schema.

krb Specifies the authentication version to use on the Kerberos credentials that are stored on the appliance.

krb5i Specifies the authentication version to use on the Kerberos credentials that are stored on the appliance. This authentication method includes a secure hash function to protect the data from being changed by the network.

krb5p Specifies the authentication version to use on the Kerberos credentials that are stored on the appliance. This authentication method includes a secure hash function to protect the data from being changed by the network, and to protect the data encryption to prevent data from being read or changed by the network.

Guidelines

Use the **AUTH_SYS** authentication method when using all versions of NFS.

Use the **krb**, **krb5i**, or **krb5p** Kerberos authentication method when using NFS version 4.

If authenticating with Kerberos, ensure that a keytab is defined in the NFS client.

Related Commands

version, **kerberos-keytab** (Crypto)

local-filesystem-access

Controls local access to the mounted file system.

Syntax

local-filesystem-access {**on** | **off**}

Parameters

on Enables local access.

off (Default) Disables local access.

Guidelines

By default, access to the mounted file system is not supported. This command enables access to the mounted file system through a folder with the name of the NFS Static Mount object.

Attempts to access an unavailable or downed file system can cause the DataPower appliance to become unstable.

read-only

Specifies the NFS mount type.

Syntax

read-only {**on** | **off**}

Parameters

on Indicates read-only file access.

off (Default) Indicates read-write file access.

Guidelines

he **read-only** command specifies the mount type as read-only. This setting allows only file read operations on NFS mounts. By default, NFS mounts are read-write file access.

When mounting the same NFS version 4 mount point in different domains, the first mount sets file access privileges. For example, if `domain-A` mounts `host:/foo` as read-only access and then `domain-B` mounts `host:/foo` as read-write access, both mounts are read-only.

Alternatively, use the **no read-only** command to restore the default state, which is to allow read transactions and write transactions on NFS mounts.

remote

Identifies the remote NFS file system to make available.

Syntax

remote *mount*

Parameters

mount Specifies the NFS mount point as an ASCII string in the form `<host>:/<path>` where `<host>` is the domain name or IP address of the target NFS server, and `<path>` is a hierarchical directory path.

<path> must match or be more specific than the NFS export that is provided by the target server. For example, the server provides an export of XML/stylesheets, the <path> portion can specify XML/stylesheets or XML/stylesheets/financialServices, (if there is a financialServices subdirectory).

Guidelines

Refer to RFC 2224, *NFS URL Schema*, for complete information about using an NFS URL.

retrans

Specifies the maximum number of RPC minor time outs to allow before the transaction fails.

Syntax

retrans *count*

Parameters

count Specifies the maximum number of RPC minor time outs to tolerate before an NFS transaction is abandoned and an NFS failure is declared. Use an integer in the range of 1 through 60. The default is 3.

Guidelines

Used in conjunction with the **timeo** command to specify behavior in response to RPC minor time outs and subsequent retransmission attempts.

The **retrans** command specifies the number of RPC minor time outs that are tolerated per NFS transaction before an NFS read or write error is declared. The **timeo** command provides a base value to determine the interval between the RPC time out and the subsequent retransmission attempt.

For example, assuming default values (3 for **retrans**, and 0.7 seconds for **timeo**), RPC time outs are dealt with as follows.

1. In response to the first RPC time out, the appliance waits 0.7 seconds and then retransmits.
2. In response to the second RPC time out, the appliance doubles the initial time out value to 1.4 seconds and then retransmits.
3. In response to the third RPC time out, the appliance doubles the previously used time out value to 2.8 seconds and then retransmits.
4. In response to the fourth time out, which is greater than the value specified by the **retrans** command, the appliance declares an NFS read or write error.

The interval between RPC time outs and a subsequent retransmission will never exceed 6 seconds.

Related Commands

timeo

rsize

Specifies the size of each read operation.

Syntax

rsize *size*

Parameters

size Specifies the number of bytes in each NFS read operation. Use an integer in the range of 1024 through 32769. The default is 4096.

Guidelines

Operations greater than 8192 bytes should only be used with TCP as the transport-layer protocol.

Related Commands

read-only ,transport, wsize

---

timeo

Specifies the initial interval between an RPC minor time out and a subsequent retransmission attempt.

Syntax

timeo *time*

Parameters

time Specifies the interval in tenths of a second between an initial RPC time out and the subsequent retransmission by the appliance. Use an integer in the range of 1 through 60. The default is 7.

Guidelines

Used in conjunction with the **retrans** command to specify behavior in response to RPC minor time outs and subsequent retransmission attempts.

The **retrans** command specifies the number of RPC minor time outs that are tolerated per NFS transaction before an NFS read or write error is declared. The **timeo** command provides a base value to determine the interval between the RPC time out and the subsequent retransmission attempt.

For example, assuming default values (3 for **retrans**, and 0.7 seconds for **timeo**), RPC time outs are dealt with as follows.

1. In response to the first RPC time out, the appliance waits 0.7 seconds and then retransmits.
2. In response to the second RPC time out, the appliance doubles the initial time out value to 1.4 seconds and then retransmits.
3. In response to the third RPC time out, the appliance doubles the previously used time out value to 2.8 seconds and then retransmits.
4. In response to the fourth time out, which is greater than the value specified by the **retrans** command, the appliance declares an NFS read or write error.

The interval between RPC time outs and a subsequent retransmission will never exceed 6 seconds.

Related Commands

retrans

transport

Identifies the preferred transport-layer protocol.

Syntax

transport {**tcp** | **udp**}

Parameters

tcp (Default) Identifies TCP as the protocol.

udp Identifies UDP as the protocol.

Guidelines

The **transport** command specifies the preferred transport-layer protocol to use, if available. Use the TCP protocol to perform read or write transactions larger than 8192 bytes.

- For NFS version 2 or version 3, select the transport protocol to use when initiating the mount. If TCP is selected and it is not available on the NFS server, UDP will be used instead.
- For NFS version 4, this property is ignored. NFS version 4 only supports TCP.

Related Commands

rsize, **wsize**

version

Identifies the preferred protocol version.

Syntax

version {**2** | **3** | **4**}

Parameters

2 Specifies NFS version 2.

3 (Default) Specifies NFS version 3.

4 Specifies NFS version 4.

Guidelines

The **version** command specifies the preferred NFS protocol version. If the version is 3, but the server only implements version 2, the client falls back to version 2. If the version is 4, there is no fallback.

wsize

Specifies the size of each write operation.

Syntax

wsize *bytes*

Parameters

bytes Specifies the number of bytes in each NFS write operation. Use an integer in the range of 1024 through 32769. The default is 4096.

Guidelines

Operations greater than 8192 bytes should only be used with TCP as the transport-layer protocol.

Related Commands

rsize, **transport**

Examples

- Enters NFS Static Mounts configuration mode to create the `danvilleServer` NFS Static Mount object. Specifies the `server2` NFS server and `export/XML/stylesheets` mount point. Sets the mount time out to 1 minute. Sets the read and write request sizes to 8192 bytes. Sets the maximum number of RPC time outs per NFS transaction to 4. Sets the initial retransmission interval to 0.5 seconds.

  Default values are retained for the transport-layer protocol (TCP), for the NFS version (3), for the mount type (read-write), and for local access to the mount (disabled).

```
# nfs-static-mount danvilleServer
New NFS Static Mounts configuration
# server2:/export/XML/stylesheets
# mount-timeout 60
# rsize 8192
# wsize 8192
# retrans 4
# timeo 5
# exit
#
```

Chapter 60. NTP Service configuration mode

This chapter provides an alphabetic listing of commands that are available in NTP Service configuration mode.

To enter this configuration mode, use the Global **ntp-service** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in NTP Service configuration mode.

refresh-interval

Designates the interval between clock synchronizations.

Syntax

refresh-interval *frequency*

Parameters

frequency
Specifies the number of seconds between clock synchronizations. Use an integer in the range of 60 through 86400. The default is 900.

Guidelines

In the absence of a specified interval the appliance synchronizes with the NTP server every 15 minutes.

Related Commands

remote-server

Examples

- Identifies the NTP server and specifies a clock synchronization interval of 5 minutes.

```
# ntp-service
NTP Service configuration mode
# remote-server Chronos-1
# refresh-interval 300
#
```

remote-server

Identifies an NTP server.

Syntax

remote-server *server*

no remote-server

Parameters

server Identifies the NTP server by host name or IP address.

Guidelines

From the command line, the appliance supports one NTP server at a time. To designate a new NTP server, use the **no ntp-service** command to delete the current server.

Note: The WebGUI supports the specification of multiple NTP servers. If you invoke the **no ntp-service** command, all defined NTP servers are deleted. To delete just one of the defined NTP servers, use the WebGUI.

Related Commands

refresh-interval

Examples

- Identifies the NTP server and specifies a clock synchronization interval of 5 minutes.

```
# ntp-service
NTP Service configuration mode
# remote-server Chronos-1
# refresh-interval 300
#
```

Chapter 61. Peer Group configuration mode

This chapter provides an alphabetic listing of commands that are available in Peer Group configuration mode.

To enter this configuration mode, use the Global **peer-group** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Peer Group configuration mode.

type

Identifies the peer group type.

Syntax

type slm

Parameters

slm Specifies a token for an SLM Monitoring Peer Group.

Guidelines

The firmware supports only the SLM type.

Examples

- Enters Peer Group configuration mode to create the `SLM-Group1` Peer Group. Identifies the peer group as of type SLM.

```
# peer-group SLM-Group1
Peer Group configuration mode
# type slm
#
```

url

Identifies a member of the current peer-group.

Syntax

url *member*

Parameters

member
Identifies a peer group member by IP address or domain name.

Guidelines

When configuring a peer group you must add this DataPower appliance to the peer group list; the peer group lists must be identical across all group members.

Examples

- Enters Peer Group configuration mode to create the SLM-Group1 Peer Group. Specifies the peer group type as SLM and designates group members.

```
# peer-group SLM-Group1
Peer Group configuration mode
# type slm
# url 192.168.12.100
# url 192.168.49.13
# url 192.168.80.126
#
```

Chapter 62. Policy Attachments configuration mode

This chapter provides an alphabetic listing of commands that are available in Policy Attachments configuration mode.

To enter this configuration mode, use the Global **policy-attachments** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

enforcement-mode

Filters or modifies messages.

Syntax

enforcement-mode {**enforce** | **filter**}

Parameters

enforce
Creates a configuration that might transform client requests or client responses to satisfy policy.

filter Creates a configuration that rejects client requests and server responses that do not satisfy policy. A rejection triggers error handling.

Guidelines

The **enforcement-mode** command defines how the configuration enforces WS-Policy. For example, if a policy requires a response to be encrypted, **filter** will reject the response and trigger error handling if the output is not encrypted, but **enforce** will encrypt the outgoing.

If the mode is **enforce** and the configuration does not provide the required policy parameters to encrypt the output, the mode switches to **filter** behavior and triggers error handling and the log contains a message that is similar to the following warning:

```
Wed Nov 07 2007 08:24:00 [ws-security-policy][ws-proxy][warn]
wsgw(wssp-policy-015h): tid(1425)[request]: WS-SecurityPolicy Mapping:
A message cannot be encrypted during enforcement
```

external-policy

Associate external policy with a service or port.

Syntax

external-policy {**service** | **port**} *wsdlComponentValue URL*

Parameters

service
Indicates to associate the policy with a WSDL service.

port Indicates to associate the policy with a WSDL port.

wsdlComponentValue
Specifies the QName of a WSDL component in the {*namespace*}*ncname* format.

URL Specify the location of the document that contain the policy to attach.

---

ignore-attachment-point

Disables external policy for a service or port.

Syntax

ignore-attachment-point {**service** | **port**} *wsdlComponentValue*

Parameters

service
Indicates to ignore the policy for a WSDL service.

port Indicates to ignore the policy for a WSDL port.

wsdlComponentValue
Specifies the QName of a WSDL component in the {*namespace*}*ncname* format.

Guidelines

The **ignore-attachment-point** command disable all policies that are attached by policy reference at a configured attachment point. All other policy references remain intact.

---

policy-references

Controls WSDL-defined policy references.

Syntax

policy-references {**on** | **off**}

Parameters

on Uses WSDL-defined policy references.

off Ignores WSDL-defined policy references.

Guidelines

WSDL-defined policy references are attached to a WSDL file with `PolicyURI` attributes and `<wsp:PolicyReference>` elements. These attachments are sometimes called *XML element attachments*. When **off**, all XML element attachments are ignored and only external policies are enforced.

Chapter 63. Policy Parameters configuration mode

This chapter provides an alphabetic listing of commands that are available in Policy Parameters configuration mode.

To enter this configuration mode, use the Global **policy-parameters** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

parameter

Defines a policy parameter.

Syntax

parameter *name value*

Parameters

name Specifies the name of the parameter in the {*domainNamespace*}*key* format.

value Specifies the value for the parameter.

Guidelines

The **parameter** command defines a policy parameter as a key-value pair. The key specifies the name of a policy parameter that is used in a policy mapping style sheet. Use this command to defined all required policy parameters that are needed by the policy mapping style sheet.

A policy parameters is the way that you must map the needed parameters that are defined in or referenced by the WSDL policy or that are defined in an attached source to the specific DataPower object. If you do not define all the needed parameters, processing a message with the defined WS-Policy generates errors.

For example, you might need an X.509 token to use the defined WS-Policy. If you need an X.509 token, you need to define which certificate that is stored on the DataPower appliance to use. If the certificate is `alice`, you would need to set the `{http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702}ws-secpol-Certificate` parameter to `alice`.

Note: If you defined a policy parameters at the **port** or **port-operation** level, these parameters are not applied to its parallel synthesize port or operation. The policy parameters for synthesized ports and operations must be inherited from the **service** level or redefined at the synthesized level.

Chapter 64. Processing Action configuration mode

This chapter provides an alphabetic listing of commands that are available in Processing Action configuration mode.

To enter this configuration mode, use the Global **action** command. While in this mode, create a named, reusable processing action.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

aaa-policy

Identifies an AAA Policy for the current `aaa` action.

Syntax

aaa-policy *name*

Parameters

name Specifies the name of the AAA Policy.

Examples

- Assigns the `AAA-Policy1` AAA Policy to the current `aaa` action.

```
# type aaa
# aaa-policy AAA-Policy1
#
```

async-action

Specifies the action for the current `event-sink` action.

Syntax

async-action *action*

Parameters

action Specifies the name of the action to wait for

Guidelines

The **async-action** command specifies the name of an asynchronous action that the current `event-sink` action should wait for. This command is meaningful only when the action type specified by the **type** command is `event-sink`.

Related Commands

type

Examples

- Causes the `event-sink` action to wait until the `async-fetch-1` and `async-fetch-2` actions complete.

```
# type event-sink
# async-action async-fetch-1
# async-action async-fetch-2
```

asynchronous

Indicates when to run the action asynchronously.

Syntax

asynchronous {**on** | **off**}

Parameters

on Runs the action asynchronously.

off (Default) Runs the action synchronously.

Guidelines

The **asynchronous** command indicates whether to run the current action asynchronously.

When asynchronous, the action runs in parallel with other actions. A later `event-sink` action can wait for named actions to complete. In this case, the output of asynchronous actions are visible after processing the `event-sink` action.

Without an `event-sink` action, the output of an asynchronous action is not reliably available to subsequent actions. In this case, there are no guarantees as to when the asynchronous action will finish, errors that are generated by the action are ignored by the remainder of the processing rule, and the network transaction could complete while this action is still in progress.

Examples

- Specifies that the `results` action sends the data in the `context-to-log` input context to `http://log-server/log-me`, and indicates that processing does not wait for an acknowledgement or response.

```
# type results
# input context-to-log
# destination http://log-server/log-me
# asynchronous on
```

attachment-uri

Specifies the URI of an attachment that is to be stripped from a MIME multipart package by the current `strip-attachments` action.

Syntax

attachment-uri *uri*

Parameters

uri Identifies a document attachment to be stripped from the MIME multipart package.

Guidelines

attachment-uri is used only if the action type (as specified by the **type** command) is `strip-attachments`.

Related Commands

type

Examples

- Strips attachments from the specified document.

```
# type strip-attachments
# attachment-uri https://sona/TestBase/simple.xsl
#
```

---

condition

Specifies a match condition and action for the current `conditional` action.

Syntax

condition *xpath action*

Parameters

xpath Specifies the XPath expression to match against the input.

action Specifies the name of the action to run when the XPath expression matches.

Guidelines

The **condition** command specifies an XPath expression as the match criteria and the name of the action to run when that XPath expression matches. Use this command to define multiple condition-based clauses. The first XPath expression that matches against the named input context invokes the corresponding action.

This command is meaningful only when the action type specified by the **type** command is `conditional`.

Related Commands

type

Examples

- Specifies that the `conditional` action runs the `process-request` action when the message contains the `Request` element. Otherwise, the action runs the `process-any` action.

```
# type conditional
# condition //*[local-name()='Request'] process-request
# condition . process-any
```

destination

Either identifies an external resource or identifies the target destination for a transmitted message.

Syntax

destination *uri*

Parameters

uri Identifies the resource or message destination.

Guidelines

destination is required when the action type is `fetch`, `log`, `results-async`, or `route-set`. This command is optional when the action type is `results`.

- When the action type is `fetch`, specifies the source location of the resource to be retrieved.
- When the action type is `log`, specifies the destination location of the transmitted message.
- When the action type is `results-async`, specifies the destination location of the data recipient. A destination URI can take the form of a DataPower variable, a context name, or a standard URI.
- When the action type is `route-set`, specifies the routing destination.
- When the action type is `results`, specifies the destination location of the data recipient.

Related Commands

multiple-outputs

Examples

- Stores the fetch resource in the destination `identity.xml`.

```
# type fetch
# destination http://datapower.com/identity.xml
#
```

- Stores the log resource in the destination `logexamples.xml`.

```
# type log
# destination http://datapower.com/logexamples.xml
#
```

- Stores the results resource in the destination `results.log`.

```
# type results
# destination http://datapower.com:9999/results.log
#
```

- Sends the input of the results action to the list of URLs specified in the variable given.

```
# type results
# destination var://context/mycontext/urllist
#
```

dynamic-schema

Specifies a dynamic schema for the current `validate` action.

Syntax

dynamic-schema *schema*

Parameters

schema Identifies the dynamic schema.

Guidelines

The **dynamic-schema** command is used only if the action type (as specified by the **type** command) is `validate` to identify a dynamic schema to validate incoming documents.

Examples

- Specifies the dynamic schema used for document validation.

```
# type validate
# dynamic-schema https://sona/TestBase/validate.xsd
#
```

dynamic-stylesheet

Specifies a dynamic style sheet to process documents.

Syntax

dynamic-stylesheet *url*

Parameters

url Identifies the URL of the dynamic style sheet.

Guidelines

The **dynamic-stylesheet** command is used when the action type (as specified by the **type** command) is `route-action`, `xform`, `xformbin`, or `xformpi` to either route or transform documents.

Examples

- Specifies the `validate.xsl` dynamic style sheet to use to transform the document.

```
# type xform
# dynamic-stylesheet https://sona/TestBase/validate.xsl
#
```

error-input

Specifies the input context for an `on-error` action.

Syntax

error-input *context*

Parameters

context Identifies the input context.

Guidelines

The **error-input** command is used only if the action type (as specified by the **type** command) is `on-error`.

If no context is explicitly identified, the input context of the failed action is used.

Examples

- Specifies `temp1` as the input context for the on-error action.

```
# type on-error
# error-input temp1
#
```

---

error-mode

Specifies the error response mechanism for the current `on-error` action.

Syntax

error-mode {**abort** | **alternative** | **continue**}

Parameters

abort (Default) Indicates that processing ceases.

alternative
Indicates that processing invokes the alternative processing rule.

continue
Indicates that processing continues with the next action.

Guidelines

The **error-mode** command is used only if the action type (as specified by the **type** command) is `on-error` to determine whether to continue processing or to cease processing of the current processing rule.

Examples

- Specifies to continue processing the current processing rule in the event of an error.

```
# type on-error
# error-mode continue
#
```

---

error-output

Specifies the output context for an `on-error` action.

Syntax

error-output *context*

Parameters

context Identifies the output context.

Guidelines

The **error-output** command is used only if the action type (as specified by the **type** command) is `on-error`.

If no context is explicitly identified, the output context of the failed action is used.

Examples

- Specifies `trashCan` as the output context for the action.

```
# type on-error
# error-output trashCan
#
```

---

event

Specifies the event that triggers a `checkpoint` action.

Syntax

event *trigger*

Parameters

trigger Identifies one of the following values as the event that triggers the checkpoint:

AuthComplete
(Default) Signifies the completion of an authentication process.

Fault Signifies a fault condition.

Request
Signifies the input of a client-originated document.

Response
Signifies the input of a server-originated document.

Guidelines

The **event** command is used only if the action type (as specified by the **type** command) is `checkpoint`.

Examples

- Specifies `fault` as the checkpoint trigger.

```
# type checkpoint
# event fault
#
```

---

input

Specifies the input context for the current action.

Syntax

input *context*

Parameters

context Identifies the input context for the current action.

Guidelines

The **input** command is required when the action type (as specified by the **type** command) is `aaa`, `call`, `checkpoint`, `convert-http`, `extract`, `filter`, `log`, `results`, `results-async`, `route-action`, `setvar`, `slm`, `strip-attachments`, `validate`, `xform`, `xformbin`, or `xformpi`.

The **input** command is not used when the action type is `fetch`, `on-error`, `rewrite`, or `route-set`.

Examples

- Identifies INPUT as the input context for an `extract` action.

```
# type extract
# input INPUT
#
```

input-conversion

Specifies the conversion map for a `convert-http` action.

Syntax

input-conversion *name*

Parameters

name Specifies the name of a conversion map.

Guidelines

The **input-conversion** command is used only if the action type (as specified by the **type** command) is `convert-http` to specify the conversion map to translate non-XML content.

Examples

- Identifies the `httpToXML` conversion map for the current `convert-http` action.

```
# type convert-http
# input-conversion httpToXML
#
```

iterator-count

Specifies the number of times to run the specified action for the current `for-each` action.

Syntax

iterator-count *count*

Parameters

count Specifies the number of times to run the loop action. Use an integer in the range of 1 through 32768.

Guidelines

The **iterator-count** command specifies the number of times to run the specified action for the current `for-each` action. During the loop, the `var://service/multistep/loop-count service` variable is set to the current iteration of the loop. The first iteration starts the count at 1.

This command is meaningful only when both of the following conditions are met:

- The action type that is specified by the **type** command is `for-each`.
- The iteration type that is specified by the **iterator-type** command is `count`.

Related Commands

iterator-type, **loop-action**, **type**

Examples

- Specifies that the `transformer` action runs 4 times.

```
# type for-each
# iterator-type count
# iterator-count 4
# loop-action transformer
```

iterator-expression

Specifies an XPath expression for the current `for-each` action.

Syntax

iterator-expression *expression*

Parameters

expression
Specifies the XPath expression that the loop should use.

Guidelines

The **iterator-expression** command specifies the XPath expression to apply to the input context of a loop. The loop action runs one time for each item in the node set that the XPath expression produces. The `var://service/multistep/loop-iterator` service variable is set to the matching node.

This command is meaningful only when both of the following conditions are met:

- The action type that is specified by the **type** command is `for-each`.
- The iteration type that is specified by the **iterator-type** command is `xpath`.

Related Commands

input, **iterator-type**, **loop-action**, **type**

Examples

- Specifies that the `transformer` action runs one time for each `item` element in the `INPUT` input context.

```
# type for-each
# input INPUT
# iterator-type xpath
# iterator-expression //*[local-name()='item']
# loop-action transformer
```

iterator-type

Indicates the iteration type for the current `for-each` action.

Syntax

iterator-type {**count** | **xpath**}

Parameters

count Indicates that iterations are based on a fixed count.

xpath Indicates that iterations are based on each XPath expression match.

Guidelines

The **iteration-type** command indicates whether a `for-each` action either run a fixed number of iterations or runs one time for each XPath expression match in the input context.

- To run a fixed number of iterations, use the **iterator-count** command.
- To run one time for each XPath expression match, use the **iterator-expression** command.

This command is meaningful only when the action type (as specified by the **type** command) is `for-each`.

Related Commands

iteration-count, **loop-action**, **iteration-expression**, **type**

Examples

- Specifies that the `transformer` action runs one time for each `item` element in the `INPUT` input context.

```
# type for-each
# input INPUT
# iterator-type xpath
# iterator-expression //*[local-name()='item']
# loop-action transformer
```

- Specifies that the `transformer` action runs 4 times.

```
# type for-each
# iterator-type count
# iterator-count 4
# loop-action transformer
```

log-level

Specifies the priority of the message that is generated by the current `log` action.

Syntax

log-level *priority*

Parameters

priority

Specifies one of the following message priority:

emergency
alert
critical
error
warning
notice (Default)
info
debug

Guidelines

The **log-level** command is used only if the action type (as specified by the **type** command) is `log`.

Examples

- Identifies the message priority as `warning`.

```
# type log
# log-level warning
#
```

log-type

Specifies the category of the message that is generated by the current `log` action.

Syntax

log-type *messageType*

Parameters

messageType

Specifies the category of event. Use the **show logging event** command to view available categories.

Guidelines

The **log-type** command is used only if the action type (as specified by the **type** command) is `log`.

Related Commands

show logging event

Examples

- Identifies that the `log` action sends messages as belonging to the `aaa` message category.

```
# type log
# log-type aaa
#
```

loop-action

Specifies the action to run for the current `for-each` action.

Syntax

loop-action *action*

Parameters

action Specifies the name of an existing action to run.

Guidelines

The **loop-action** command specifies the name of the existing action within the current `for-each` action. The output context of the `for-each` action replaces the output context of the named action. If the output context of the `for-each` action and the named action are the same and the value of the **multiple-outputs** command is **on**, the final output context uses the same name and appends a number.

This command is meaningful only when the action type (as specified by the **type** command) is `for-each`.

Related Commands

input, **iterator-count**, **iterator-expression**, **iterator-type**, **multiple-outputs**, **type**

Examples

- Specifies that the `transformer` action runs one time for each `item` element in the `INPUT` context.

```
# type for-each
# input INPUT
# iterator-type xpath
# iterator-expression //*[local-name()='item']
# loop-action transformer
```

---

multiple-outputs

Indicates whether to generate an output context for each iteration or result.

Syntax

multiple-outputs {**on** | **off**}

Parameters

on Generates multiple output contexts.

off (Default) Generates a single output context.

Guidelines

The **multiple-outputs** command specifies that `for-each` and `results` actions should generate multiple output contexts. When enabled and the output context is `ctx`, the iterations or individual results generate separate contexts (`ctx_1`, `ctx_2`, and so forth).

This command is meaningful only when the action type (as specified by the **type** command) is `for-each` or `results`.

Related Commands

iterator-count, **iterator-expression**, **iterator-type**, **loop-action**, **output**, **type**

Examples

- Specifies that the `transformer` action runs one time for each `item` element in the input context. The processing generates output contexts `out_1`, `out_2`, and so forth.

```
# type for-each
# output out
# multiple-outputs
# iterator-type xpath
# iterator-expression //*[local-name()='item']
# loop-action transformer
```

named-inouts

Sets the method to determine the location of named inputs and outputs for an `xformbin` action.

Syntax

named-inouts {**default** | **explicit** | **dynamic**}

Parameters

default
Specifies that the contexts defined for the action contain the input data and will contain the output data.

explicit
Specifies that each input context is explicitly defined with the **name** command and that each output context is explicitly defined with the **name** command.

dynamic
Specifies that the contexts are defined dynamically during processing.

Guidelines

The **named-inouts** command is used only when the action (as defined by the **type** command) is `xformbin` and the **tx-map** command specifies a map file generated by WebSphere Transformation Extender (WTX). Use the **named-input** and **named-output** command to define each input or output `cardname` and its respective context.

Related Commands

named-input, **named-output**, **type**, **tx-map**

Examples

- Sets the mode to `explicit`.

```
# tx-map local:///invoices.dpa
# named-inouts explicit
# named-input "ContactFileCard" "INPUT"
# named-input "SecondCard" "custom1"
# named-output "LabelCard" "OUTPUT"
#
```

named-input

Identifies the name of the input card and processing context.

Syntax

named-input *card context*

Parameters

card Specifies the name of an input that is expected by the map. This name must be the same as a `cardname` in the `<Inputs>` section of the map file.

context Specifies the name of the DataPower processing context that contains the input data. You must arrange to fill this context with the expected input data.

Guidelines

The **named-input** command is used only when the **named-inouts** command is **explicit**.

Related Commands

named-inouts, **name**

Examples

- Sets the mode to `explicit`. The DataPower processing input contexts are `INPUT` and `custom1` and are defined by `cardname` entries in the `<Inputs>` section of the map file as `ContactFileCard` and `SecondCard`, respectively.

```
# tx-map locl:///invoices.dpa
# named-inouts explicit
# named-input "ContactFileCard" "INPUT"
# named-input "SecondCard" "custom1"
#
```

---

named-output

Identifies the name of the output card and processing context.

Syntax

name *card context*

Parameters

card Specifies the name of an output that is expected by the map. The name must be the same as a `cardname` in the `<Outputs>` section of the map file.

context Specifies the name of the DataPower processing context to contain the output data.

Guidelines

The **named-output** command is used only when the **named-inouts** command is **explicit**.

Related Commands

named-inouts

Examples

- Sets the mode to `explicit`. The DataPower processing output context is `OUTPUT` and is defined by the `cardname` entry in the `<Outputs>` section of the map file as `LabelCard`.

```
# tx-map locl:///invoices.dpa
# named-inouts explicit
# name-output "LabelCard" "OUTPUT"
#
```

---

output

Specifies the output context for the current action.

Syntax

output *context*

Parameters

context Identifies the output context.

Guidelines

The **output** command is required when the action type (as specified by the **type** command) is `call`, `convert-http`, `extract`, `fetch`, `xform`, `xformbin`, or `xformpi`.

output is optional when the action type is `aaa`, `filter`, `log`, `results`, `route-action`, `slm`, or `validate`.

The **output** command is not used when the action type is `checkpoint`, `on-error`, `results-async`, `rewrite`, `route-set`, `setvar`, or `strip-attachments`.

Examples

- Specifies `temp2` as the output context for the current `fetch` action. For a `fetch` action, the output context is the context that contains the retrieved resource.

```
# type fetch
# output temp2
#
```

---

output-type

Characterizes the outgoing data.

Syntax

output-type *datatype*

Parameters

datatype

Identifies one of the following values as the data type of the outgoing data:

default

Indicates that the data type is determined by reading the content type field of the document; if the content type is XML or undeclared, the data is treated as XML. Otherwise, the data is treated as binary.

xml Indicates that the data is treated and parsed as XML.

binary Indicates that the data is treated as binary and unprocessed.

Guidelines

The **output-type** command is optionally used only when the action type (as specified by the **type** command) is `fetch`, `log`, `results`, `xform`, `xformbin`, or `xformpi`.

Examples

- Characterizes the retrieved data as `binary`.

```
# type fetch
# output-type binary
#
```

parameter

Defines a stylesheet parameter for the current action.

Syntax

parameter *name value*

Parameters

name Specifies the name of the parameter.

value Specifies the value of the parameter.

The **parameter** command is optionally used to specify a parameter for the current style sheet only when the action type (as specified by the **type** command) is `filter`, `route-action`, `xform`, `xformbin`, or `xformpi`.

Examples

- Defines the `staticRoute` stylesheet parameter and sets its value to `192.168.12.12`.

```
# type route-action
# parameter staticRoute 192.168.12.12
#
```

results

Specifies the behavior of multiple targets for the current `results` action.

Syntax

results {**first-available** | **require-all** | **attempt-all**}

Parameters

first-available
(Default) Indicates that targets are dispatched one at a time. The action succeeds when the input reaches any backend target.

require-all
Indicates that targets are dispatched in parallel. The action succeeds only after the input reaches all of the backend targets.

attempt-all
Indicates that targets are dispatched in parallel. The action succeeds if the input reaches any backend target. In other words, the action is successful even when the input does not reach each of the backend targets.

Guidelines

The **results** command indicates when to consider a `results` action successful when sending to multiple backend targets.

This command is meaningful only when the action type (as specified by the **type** command) is `results`.

Related Commands

destination, **input**, **type**

Examples

- Uses the `var://context/ctx/urls` context variable to determine the location to which to send a copy of the `ctx` input context one at a time until successful.

```
# type results
# input ctx
# destination var://context/ctx/urls
# results first-available
```

retry-count

Specifies the number of retry attempts for the current `results` action.

Syntax

retry-count *count*

Parameters

count Specifies the number of times to retry a target after a failure. A value of 0 indicates no retry attempts and the action fails immediately. The default is 0.

Guidelines

The **retry-count** command specifies how many times a to retry a `results` action when an attempt to reach a backend target fails before indicating that the action fails. The next attempt is tried after the interval set with the **retry-interval** command elapses.

This command is meaningful only when the action type (as specified by the **type** command) is `results`.

Related Commands

destination, **input**, **retry-interval**, **type**

Examples

- Specifies that if the action fails to write the input to `http://log-server/log`, the request is tried 10 times at 5 seconds intervals.

```
# type results
# input ctx
# destination http://log-server/log
# retry-count 10
# retry-interval 5000
```

retry-interval

Specifies the retry interval for the current `results` action.

Syntax

retry-interval *interval*

Parameters

interval

Specifies the time between retries in milliseconds. A value of 0 indicates that failed connections will be retried immediately. The default is 1000.

Guidelines

The **retry-interval** command specifies how long a results action waits before attempting to retry a connection to a target. This command is meaningful only when the action type (as specified by the **type** command) is `results`.

Related Commands

destination, **input**, **retry-count**, **type**

Examples

- Specifies that if the action fails to write the input to `http://log-server/log`, the request is tried 10 times at 5 seconds intervals.

```
# type results
# input ctx
# destination http://log-server/log
# retry-count 10
# retry-interval 5000
```

rule

Specifies the rule invoked by a `call` action.

Syntax

rule *name*

Parameters

name Specifies the name of an existing processing rule.

Guidelines

This command is used only if the action type (as specified by the **type** command) is `call`. This command identifies the rule invoked by the `call` action.

Examples

- Indicates that the `call` action invokes the `validateSOAP` processing rule.

```
# type call
# rule validateSOAP
#
```

schema-url

Specifies a schema to be used in validation operations by the current `validate` action.

Syntax

schema-url *url*

Parameters

url Identifies the schema used for document validation.

Guidelines

The **schema-url** command is used only if the action type (as specified by the **type** command) is `validate`. This command identifies the schema to validate incoming documents.

Examples

- Indicates that the `validate` action uses the `soapSchema2.xsd` schema for validation.

```
# type validate
# schema-url local:///soapSchema2.xsd
#
```

slm

Identifies an SLM Policy for the current `slm` action.

Syntax

slm *name*

Parameters

name Specifies the name of an existing SLM Policy.

Guidelines

The **slm** command is used only if the action type (as specified by the **type** command) is `slm`. This command identifies the SLM policy.

Examples

- Indicates that the `slm` action uses the `wsMonitor1` SLM Policy.

```
# type slm
# slm wsMonitor1
#
```

soap-validation

Specify the SOAP validation type

Syntax

soap-validation {**body** | **body-or-detail** | **envelope** | **ignore-faults**}

Parameters

body Validates the contents of the SOAP body element.

body-or-detail
Validates the content of the detail element for SOAP faults and the contents of the SOAP body element.

envelope
(default) Validates the entire message to include the contents of the SOAP envelope.

ignore-faults
Performs no validation on SOAP faults. If the message is not a fault message, validates the contents of the SOAP body element.

Guidelines

The **soap-validation** command is used only if the action type (as specified by the **type** command) is `validate`. This command identifies the type of validation to perform.

Validation of SOAP messages does not affect the validation of input context to ensure that it is a valid document. If you are validating an intermediate context, such as the result of a transform, the intermediate context is not implicitly validated as SOAP. To validate the entire document, retain the default value (**envelope**).

Examples

- Indicates that only the SOAP body is subject to validation.

```
# type validate
# soap-validation body
#
```

---

sql-source

Identifies the data source object for an `sql` action.

Syntax

sql-source *name*

Parameters

name Identifies the name of an existing data source object.

Guidelines

The **sql-source** command is used only when the action type is `sql` to identify which data source to access. The data source object must already be created with the **sql-source** (Global) command.

Related Commands

sql-source (Global)

Examples

- Identifies `db2source` as the data source object to access.

```
# type sql
# sql-source db2source
#
```

sql-source-type

Identifies the source of the SQL statement for an `sql` action.

Syntax

sql-source-type {**static** | **stylesheet** | **variable**}

Parameters

static (Default) Indicates that the SQL statement is the literal text supplied by the **sql-text** command.

stylesheet
Indicates that the SQL statement is derived by executing the style sheet specified by the **transform** command against the contents on the context specified by the **input** command.

variable
Indicates that the SQL statement is contained within the variable specified by the **variable** command.

Guidelines

The **sql-source-type** command is required when the action type is `sql`. Otherwise, it is not used.

Related Commands

input, **sql-text**, **transform**, **variable**

Examples

- Indicates that the SQL statement used by the current `sql` action is the result of applying the `ODBC.xsl` style sheet to the contents of the `initFilter` context.

```
# type sql
# input initFilter
# sql-source-type stylesheet
# transform ODBC.xsl
#
```

sql-text

Provides the SQL statement for an `sql` action as a literal string.

Syntax

sql-text *statement*

Parameters

statement
Specifies the SQL statement.

Guidelines

The **sql-text** command is required when the action type is `sql` and the **sql-source-type** is `static`. Otherwise, it is not used.

Related Commands

sql-source-type

Examples

- Defines the `SELECT SSN, BaseSalary FROM Senior Management` SQL statement to be executed by the `sql` action against the `db2Source` data source.

```
# type sql
# sql-source db2Source
# sql-source-type static
# sql-text "SELECT SSN, BaseSalary FROM SeniorManagement"
#
```

---

sslcred

Identifies the SSL Proxy Profile to use with a `route-set` action.

Syntax

sslcred *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **sslcred** command is optional when the action type (as specified by the **type** command) is `route-set` to establish a secure (SSL-enabled) connection with the remote destination. Otherwise, it is not used.

Related Commands

type

Examples

- Indicates that the current `route-set` action uses the `SSLProfile-2` SSL Proxy Profile.

```
# type route-set
# sslcred SSLProfile-2
#
```

---

timeout

Specifies the wait duration for the current `event-sink` action.

Syntax

timeout *duration*

Parameters

duration
Specifies the time to wait for the action to complete in milliseconds.

Guidelines

The **timeout** command specifies the duration that an `event-sink` action waits for its named actions to complete. A value of 0 indicates that the action waits indefinitely.

This command is meaningful only when the action type that is specified by the **type** command is `event-sink`.

Related Commands

type

Examples

- Indicates that the async-fetch action has only 1 second to complete. If it does not complete during this interval, it fails.

```
# type event-sink
# async-action async-fetch
# timeout 1000
```

transform

Specifies a style sheet for the current action.

Syntax

transform *url*

Parameters

url Identifies the name and location of the style sheet.

Guidelines

The **transform** command is required when the action type (as specified by the **type** command) is `filter`, `route-action`, `xform`, `xformbin`, or `xformpi`. The **transform** action is also required when the action type is `sql` and the **sql-source-type** is `stylesheet`. In this case, the **transform** command identifies the style sheet to run against the contents of the input context to create the SQL statement.

Related Commands

sql-source-type, **type**

Examples

- Identifies the `processHeader.xsl` style sheet in the `local:` directory for the current `xform` action.

```
# type xform
# transform local:///processHeaders.xsl
#
```

tx-map

Specifies the map file used by a Binary Transform action.

Syntax

tx-map *url*

Parameters

url Identifies the location of the map file generated by the WebSphere Transformation Extender (WTX).

Guidelines

The **tx-map** command specifies the WTX-generated map file that the Binary Transform (`xformbin`) action uses to determine the input contexts and the output contexts of the transform.

The map file must reside on the appliance. A single map file can contain definitions for more than one map.

With the **tx-map** command, ensure the definition of the following commands:

- The **named-inouts** command specifies the method to determine the input contexts and the output contexts.
- The **dynamic-stylesheet** command should not be used.
- If a single map file contains definitions for more than one map, the **tx-tlm** command determines which map the transform uses as the top-level map.
- If the **tx-mode** command uses**dpa**, the tx-map file must be a file that was compiled in WTX in DataPower mode and this file must have a `.dpa` extension.

Related Commands

dynamic-stylesheet, **named-inouts**, **tx-tlm**

Examples

- Specifies that the Binary Transform action uses the `invoices.dpa` map file in the `local:` directory.

```
# tx-map local:///invoices.dpa
# named-inouts explicit
# name "ContactFileCard" "INPUT"
# name "SecondCard" "custom1"
# name "LabelCard" "OUTPUT"
```

---

tx-mode

Indicates whether to use mapping logic in XML-to-binary or binary-to-XML WTX maps.

Syntax

tx-mode {**dpa** | **no-map** | **default**}

Parameters

dpa This option is the recommended mode. Specify to process the map in normal mode. In this mode, the map is processed without a special option. The **tx-map** file must be a file that was compiled in WTX in DataPower mode and this file must have a `.dpa` extension.

no-map
This option is deprecated. Select to disable mapping logic.

default
This option is deprecated. Specify when using an XML file that you exported from WTX Design Studio to the appliance.

Guidelines

If the **tx-mode** is not defined from the CLI, the default value is programmatically determined based on the value provided for the **tx-map** command:

- If the map ends with `.dpa`, the default is **dpa**.
- If the map ends with `.xml`, the default is **default**.

The **tx-mode** command indicates whether to disable mapping logic in XML-to-binary or binary-to-XML WTX maps.

The **no-map** option is valid only with maps that have exactly one input card and one output card. Further, the type of at least one of the cards must have XML intent. While legal if both cards have XML intent, this option will disable all transformation, which effectively makes the map an identity transform.

- If the input is non-XML, the input `TypeTree` alone will be used to transform the input to an XML representation of the underlying binary format. That XML representation will then be sent directly to the output context, without transformation. (Note the format of the XML representation corresponds to the *input* `TypeTree`, and might not match the XML format that is described by the output `TypeTree`).
- If the output is non-XML, the output `TypeTree` alone will be used to transform the XML input to the output binary format. (Note the format of the XML input must correspond to the *output* `TypeTree`, which might not match the XML format that is described by the input `TypeTree`).

The XML format that is produced or consumed by the binary-to-XML transformation corresponds directly to the relevant input or output `TypeTree`. Each `ITEM` or `GROUP` is represented by an element whose name is the absolute name of the type. In the representation, spaces are replaced by two dashes and other non-NCName characters are escaped as "`-nn-`" (where `nn` is the decimal value of the character code-point without leading zeros).

After any escaping, if the first character in the name is not an XML letter or underscore, the name is preceded by an underscore. For example:

- The `Foo Bar Root` type will correspond to the `Foo--Bar--Root` element
- The `Foo_Bar Baz? Root` type will correspond to the `Foo_Bar--Baz-63---Root` element
- The `#93 Root` type will correspond to the `_-35-93--Root` element

Only concrete (nonpartitioned) `ITEM` and `GROUP` structures are represented in the XML format. When a binary input `ITEM` or `GROUP` is partitioned, the XML representation that is produced will have an element that corresponds to the actual partition that is seen in the input, and not the partitioned base type. When a binary output `ITEM` or `GROUP` is partitioned, the input XML must contain an element that corresponds to the actual partition to be produced, not the partitioned base type.

tx-tlm

Determines which map in the map file to use.

Syntax

tx-tlm *name*

Parameters

name Specifies the name of a map in a map file.

Guidelines

The **tx-tlm** command specifies which map in the WTX-generated map file that the Binary Transform (`xformbin`) action uses this property to determine the input contexts and the output contexts of the transform.

Before using this command, use the **tx-map** command to specify the location of the map file that contains the map.

Not applicable if you are using **dpa** as the **tx-mode**.

Related Commands

tx-map

Examples

- Specifies that the Binary Transform action uses the `invoices.mms.xml` map file in the `local:` directory. Because the map file contains more than one map, specifies that `InvoiceItems` is the name of the top-level map in the map file.

```
# tx-map local:///invoices.mms.xml
# tx-tlm InvoiceItems
#
```

---

type

Specifies the action type.

Syntax

type *action*

Parameters

action Identifies one of the following values as the action:

aaa Indicates an `aaa` action. This action implements a specified AAA Policy. This action is relevant for all services except XSL Coprocessor and XSL Proxy services.

call Indicates a `call` action. This action invokes a processing rule. This action is relevant for all services.

checkpoint
Indicates a `checkpoint` action. This action records information about each transaction for reporting through Web Services Management. This action is relevant for Web Service Proxy services only.

conditional
Indicates a `conditional` action. This action enables if-then-else processing. This action is relevant for all services except XSL Coprocessor and XSL Proxy services.

convert-http
Indicates a `convert-http` action. This action converts non-XML

input to XML. Examples of non-XML input are an HTTP POST or an HTML form. This action is relevant for all services except XSL Coprocessor services.

crypto-binary
Indicates a `crypto-binary` action. This action signs, verifies, encrypts or decrypts binary data. This action is relevant for all services provided that the appliance has the PKCS7-MIME license.

event-sink
Indicates an `event-sink` action. This action waits for named asynchronous actions to complete. This action is relevant for all services except XSL Coprocessor and XSL Proxy services.

extract Indicates an `extract` action. This action applies an XPath expression to a context and stores the result in another context. This action is relevant for all services.

fetch Indicates a `fetch` action. This action retrieves a remote resource and stores it in a specified context. This action is relevant for all services.

filter Indicates a `filter` action. This action filters a document set with a specified style sheet. This action is relevant for all services except XSL Coprocessor services.

for-each
Indicates a `for-each` action. This action enables for-each processing loops. This action is relevant for all services except XSL Coprocessor and XSL Proxy services.

log Indicates a `log` action. This action generates a log message that contains the contents of a specified context and sends it to a specified destination. This action is relevant for all services.

on-error
Indicates an `on-error` action. This action specifies customized error-processing. This action is relevant for all services.

results
Indicates a `results` action. This action sends the contents of a context to a remote destination. This action is relevant for all services.

results-async
Indicates a `results-async` action. This action asynchronously transmits the contents of a context to a remote destination. This action is relevant for all services.

rewrite
Indicates a `rewrite` action. This action implements a specified URL Rewrite Policy. This action is relevant for all services except XSL Coprocessor services.

route-action
Indicates a `route-action` action. This action implements dynamic routing with an XPath or style sheet in the routing action. This action is relevant for all services except XSL Coprocessor services.

route-set
Indicates a `route-set` action. This action implements static routing

with a variable in the routing action. This action is relevant for all services except XSL Coprocessor services.

setvar Indicates a `setvar` action. This action creates a variable. This action is relevant for all services.

slm Indicates an `slm` action. This action implements a specified SLM Policy. This action is relevant for Multiprotocol Gateway and Web Service Proxy services only.

sql Indicates an `sql` action. This action runs an SQL statement against a specified data source. This action is relevant for all services except XSL Coprocessor and XSL Proxy services provided that the appliance has the SQL-ODBC license.

strip-attachments
Indicates a `strip-attachments` action. This action strips attachments from a MIME multipart package. This action is relevant for all services except XSL Coprocessor and XSL Proxy services.

validate
Indicates a `validate` action. This action performs schema validation. This action is relevant for all services.

xform Indicates an `xform` action. This action performs a style sheet-based document transform. This action is relevant for all services.

xformbin
Indicates an `xformbin` action. This action performs a binary to XML transform. This action is relevant for all services provided that the appliance has the DataGlue license.

xformpi
Indicates an `xformpi` action. This action performs a transform based on processing instructions in the candidate documents. This action is relevant for all services.

---

urlrewrite-policy

Identifies the URL Rewrite Policy implemented by the current action.

Syntax

urlrewrite-policy *name*

Parameters

name Specifies the name of an existing URL Rewrite Policy.

Guidelines

The **urlrewrite-policy** command is required when the action type (as specified by the **type** command) is `rewrite`. The command is optional when the action type is `validate`, `xform`, `xformbin`, or `xformpi`.

Examples

- Identifies `rewritePolicy-1` as the URL Rewrite Policy to implement for the current `rewrite` action.

```
# type rewrite
# urlrewrite-policy rewritePolicy-1
#
```

value

Sets the value of the variable declared in the current `setvar` action.

Syntax

value *value*

Parameters

value Specifies the value of the variable.

Guidelines

The **value** command is required when the action type (as specified by the **type** command) is `setvar`. Otherwise, it is not used.

Examples

- Assigns the value `preferredAccount` to the `customer` variable as declared by the current `setvar` action.

```
# type setvar
# variable customer
# value preferredAccount
#
```

variable

Identifies the variable declared by the current `setvar` action.

Syntax

variable *name*

Parameters

name Specifies the name of the variable name.

Guidelines

The **variable** command is required when the action type (as specified by the **type** command) is `setvar`.

The **variable** command is required when the action type is `sql` and the **sql-source-type** is `variable`. In this case, the **variable** command defines the SQL statement that the `sql` action runs.

Examples

- Assigns the value `preferredAccount` to the `customer` variable as declared by the current `setvar` action.

```
# type setvar
# variable customer
# value preferredAccount
#
```

wsdl-attachment-part

Specifies the name of the WSDL message part of the attachment for the current `validate` action.

Syntax

wsdl-attachment-part *name*

Parameters

name Specifies the name of the WSDL message part that contains the MIME attachment.

Guidelines

The **wsdl-attachment-part** command specifies the name of the WSDL message part that defines the content of a MIME attachment. The value should be the unqualified name of the message part. This name is the same as the part attribute on the corresponding `mime:content` component in the WSDL file.

When this property is not defined or has the special value `"*"`, the root MIME part is validated. The root MIME part is the part that is bound to a `soap:body`.

The command is meaningful only when the action type that is specified by the **type** command is `validate`.

Related Commands

type

wsdl-message-direction-or-name

Specifies the WSDL-defined service traffic to validate with the current `validate` action.

Syntax

wsdl-message-direction-or-name *name*

Parameters

name Specifies the name or direction of the service traffic.

Guidelines

The **wsdl-message-direction-or-name** command specifies the name or direction of the WSDL input, output, or fault that defines the service traffic to validate. Use one of the following values:

- The name of one or more WSDL input, output, or fault components.
- `"#input"` for the request direction.
- `"#output"` for the response direction.
- `"*"` for all inputs, outputs, and faults in the WSDL file.

If specified and not `"*"`, only messages that are defined for inputs, outputs, and faults that match the specified name or direction are considered valid. Faults are considered valid for the response direction.

The command is meaningful only when the action type that is specified by the **type** command is `validate`.

Related Commands

type

wsdl-operation

Specifies the name of the WSDL operation for the current `validate` action.

Syntax

wsdl-operation *name*

Parameters

name Specifies the name of the WSDL operation.

Guidelines

The **wsdl-operation** command specifies the name of the WSDL operation that defines the service traffic to validate. The value should be the unqualified name of the WSDL operation or `"*"` for all operations that are defined in the WSDL file.

If specified and not `"*"`, only messages that are defined for WSDL operations that match the specified name are considered valid.

The command is meaningful only when the action type that is specified by the **type** command is `validate`.

Related Commands

type

wsdl-port

Specifies the QName of the WSDL port for the current `validate` action.

Syntax

wsdl-port *qname*

Parameters

qname Specifies the QName of a WSDL port.

Guidelines

The **wsdl-port** command specifies the QName of the WSDL port. The WSDL port defines the service traffic to validate. The value should be a QName of the form `"{namespace-uri}local-part"` or `"*"` for all ports defined in the WSDL file.

If specified and not `"*"`, only messages that are defined for the named port are considered valid.

The command is meaningful only when the action type that is specified by the **type** command is `validate`.

Related Commands

type

---

wsdl-url

Identifies the WSDL URL for the current `validate` action.

Syntax

wsdl-url *url*

Parameters

url Specifies the URL of the WSDL file.

Guidelines

The **wsdl-url** command specifies the URL of the WSDL file that defines the operations to use during the `validate` action. The WSDL file can reside on the local system or on the network.

The command is meaningful only when the action type that is specified by the **type** command is `validate`.

Related Commands

type

---

xpath

Identifies the XPath expression for the current `extract` action.

Syntax

xpath *expression*

Parameters

expression
Identifies the XPath expression.

Guidelines

The **xpath** command is required when the action type (as specified by the **type** command) is `extract`. Otherwise, it in not used.

Examples

- Indicates that the current `extract` action should use `.//Order_Number` as the XPath expression.

```
# type extract
# xpath .//Order_Number
#
```

Chapter 65. Processing Metadata configuration mode

This chapter provides an alphabetic listing of commands that are available in Processing Metadata configuration mode.

To enter this configuration mode, use the Global **metadata** command. A Processing Metadata object provides access to the value of protocol headers, DataPower objects and other data related to, but not contained in, a message passing through the appliance.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Processing Metadata configuration mode.

meta-item

Adds a metadata item to the list of items retrieved by this object.

Syntax

meta-item *category name*

meta-item {**header** | **variable**} [*input*]

no meta-item

Parameters

category
Specifies the predefined metadata category that contains the desired item. The following categories are available:
- **http**
- **mq**
- **tibco**
- **wasjms**

name Specifies the name of the predefined metadata item. Refer to Table 12 on page 598 for the list of predefined items. See the list of predefined inputs below for valid values.

The value can be any alphanumeric string to define a custom protocol header or variable item

name Specifies the name of the predefined metadata item.
- Refer to Table 12 on page 598 for the list of predefined items for the **http** category.
- Refer to Table 13 on page 599 for the list of predefined items for the **mq** category.
- Refer to Table 14 on page 599 for the list of predefined items for the **tibco** category.
- Refer to Table 15 on page 600 for the list of predefined items for the **wasjms** category.

See the list of predefined inputs below for valid values.

The value can be any alphanumeric string to define a custom protocol header or variable item

header
Specifies a metadata category that allows the definition of a custom protocol header item.

variable
Specifies a metadata category that allows the definition of a custom system variable item.

input Optionally specifies the actual protocol header or system variable to examine. Only used when the metadata category is **header** or **variable**.

Guidelines

Issue this command for each item included in the complete object configuration. If the metadata desired consists of three different items of data, you would issue this command three times, once for each item. To remove all items, use the **no meta-item**.

Table 12. Predefined metadata items in the **http** *category*

Accept	If-Range	URL-in
Accept-Charset	If-Unmodified-Since	URL-out
Accept-Encoding	Last-Modified	URI-request
Accept-Language	Location	local-service-address
Accept-Ranges	Max-Forwards	Client-Address
Age	Pragma	Client-Protocol
Allow	Proxy-Authenticate	transaction-key
Authorization	Proxy-Authorization	transaction-timeout
Cache-Control	Range	transaction-name
Connection	Referer	transaction-error-code
Content-Encoding	Retry-After	transaction-error-message
Content-Language	Server	error-protocol-response
Content-Length	TE	error-protocol-reason-phrase
Content-Location	Trailer	error-subcode
Content-MD5	Transfer-Encoding	xmlmgr-name
Content-Range	Upgrade	domain-name
Content-Type	User-Agent	service-type
Date	Vary	service-name
ETag	Via	transaction-id
Expect	Warning	transaction-client
Expires	WWW-Authenticate	transaction-rule-name
From	SOAPAction	transaction-rule-type
Host	Attachment	transaction-policy-name
If-Match	Backside-Transport	input-message-size
If-Modified-Since	External-Request	response-mode
If-None-Match	Forwarded-for	rule-direction

Table 13. Predefined metadata items in the **mq** *category*

MQMD	PutApplName	Client-Protocol
StrucId	PutDate	transaction-key
Version	PutTime	transaction-timeout
Report	ApplOriginData	transaction-name
MsgType	GroupId	transaction-error-code
Expiry	MsgSeqNumber	transaction-error-message
Feedback	Offset	error-protocol-response
Encoding	MsgFlags	error-protocol-reason-phrase
CodedCharSetId	OriginalLength	error-subcode
Format	X-MQ-Receive-Q	xmlmgr-name
Priority	X-MQ-Send-Q	domain-name
Persistence	SOAPAction	service-type
MsgId	Attachment	service-name
CorrelId	Backside-Transport	transaction-id
BackoutCount	External-Request	transaction-client
ReplyToQ	Forwarded-for	transaction-rule-name
ReplyToQMgr	URL-in	transaction-rule-type
UserIdentifier	URL-out	transaction-policy-name
AccountingToken	URI-request	input-message-size
ApplIdentityData	local-service-address	response-mode
PutApplType	Client-Address	rule-direction

Table 14. Predefined metadata items in the **tibco** *category*

JMSCorrelationID	JMS_TIBCO_DISABLE_SENDER	local-service-address
JMSDeliveryMode	JMS_TIBCO_PRESERVE_UNDELIVERED	Client-Address
JMSDestination	JMS_TIBCO_MSG_EXT	Client-Protocol
JMSExpiration	JMS_TIBCO_MSG_TRACE	transaction-key
JMSMessageID	JMS_TIBCO_CM_PUBLISHER	transaction-timeout
JMSPriority	JMS_TIBCO_CM_SEQUENCE	transaction-name
JMSRedelivered	JMS_TIBCO_SS_SENDER	transaction-error-code
JMSReplyTo	JMS_TIBCO_SS_DELIVERY_MODE	transaction-error-message
JMSTimestamp	JMS_TIBCO_SS_EXPIRATION	error-protocol-response
JMSType	JMS_TIBCO_SS_LB_MODE	error-protocol-reason-phrase
JMSXUserID	JMS_TIBCO_SS_MESSAGE_ID	error-subcode
JMSXAppID	JMS_TIBCO_SS_PRIORITY	xmlmgr-name
JMSXDeliveryCount	JMS_TIBCO_SS_SENDER_TIMESTAMP	domain-name
JMSXGroupID	JMS_TIBCO_SS_TYPE_NUM	service-type
JMSXGroupSeq	JMS_TIBCO_SS_CORRELATION_ID	service-name
JMSXProducerTXID	JMS_TIBCO_SS_SEQ_NUM	transaction-id
JMSXConsumerTXID	JMS_TIBCO_SS_USER_PROP	transaction-client
JMSXRcvTimestamp	SOAPAction	transaction-rule-name
JMSXState	Attachment	transaction-rule-type
DP_JMSMessageType	Backside-Transport	transaction-policy-name
DP_JMSReplyToServer	External-Request	input-message-size
DP_JMSReplyToTopicSpace	Forwarded-for	response-mode
JMS_TIBCO_COMPRESS	URL-in	rule-direction
JMS_TIBCO_IMPORTED	URL-out	
JMS_TIBCO_SENDER	URI-request	

Table 15. Predefined metadata items in the **wasjms** *category*

JMSCorrelationID	JMS_IBM_ExceptionTimestamp	URI-request
JMSDeliveryMode	JMS_IBM_ExceptionProblemDestination	local-service-address
JMSDestination	JMS_IBM_Feedback	Client-Address
JMSExpiration	JMS_IBM_Format	Client-Protocol
JMSMessageID	JMS_IBM_Last_Msg_In_Group	transaction-key
JMSPriority	JMS_IBM_MsgType	transaction-timeout
JMSRedelivered	JMS_IBM_PutApplType	transaction-name
JMSReplyTo	JMS_IBM_PutDate	transaction-error-code
JMSTimestamp	JMS_IBM_PutTime	transaction-error-message
JMSType	JMS_IBM_Report_COA	error-protocol-response
JMSXUserID	JMS_IBM_Report_COD	error-protocol-reason-phrase
JMSXAppID	JMS_IBM_Report_Discard_Msg	error-subcode
JMSXDeliveryCount	JMS_IBM_Report_Exception	xmlmgr-name
JMSXGroupID	JMS_IBM_Report_Expiration	domain-name
JMSXGroupSeq	JMS_IBM_Report_NAN	service-type
JMSXProducerTXID	JMS_IBM_Report_PAN	service-name
JMSXConsumerTXID	JMS_IBM_Report_Pass_Correl_ID	transaction-id
JMSXRcvTimestamp	JMS_IBM_Report_Pass_Msg_ID	transaction-client
JMSXState	JMS_IBM_System_MessageID	transaction-rule-name
DP_JMSMessageType	SOAPAction	transaction-rule-type
DP_JMSReplyToServer	Attachment	transaction-policy-name
DP_JMSReplyToTopicSpace	Backside-Transport	input-message-size
JMS_IBM_Character_Set	External-Request	response-mode
JMS_IBM_Encoding	Forwarded-for	rule-direction
JMS_IBM_ExceptionMessage	URL-in	
JMS_IBM_ExceptionReason	URL-out	

See the file `store:///ProcessingMetadata.html` for complete information.

Examples

- Enters Processing Metadata configuration mode for the `MQtrans` object. Adds the `MsgId`, `ReplyToQ`, and `UserIdentifier` items to the list of items retrieved by this object.

```
# metadata MQtrans
Processing Metadata configuration mode
# meta-item mq MsgId
# meta-item mq ReplyToQ
# meta-item mq UserIdentifier
#
```

- Enters Processing Metadata configuration mode for the `HTTPtrans` object. Adds the `Host` and `Referer` items. Adds the custom `CustomID` HTTP Header to the list of items retrieved by this object.

```
# metadata HTTPtrans
Processing Metadata configuration mode
# meta-item http Host
# meta-item http Referer
# meta-item header CustID CustomID
#
```

- Enters Processing Metadata configuration mode for the `HTTPtrans` object. Removes all metadata items from the configuration.

```
# metadata HTTPtrans
Processing Metadata configuration mode
# no meta-item
#
```

Chapter 66. Processing Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Processing Policy configuration mode.

To enter this configuration mode, use the Global **stylepolicy** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Processing Policy configuration mode.

error-rule

Assigns an error rule.

Syntax

error-rule *rule*

Parameters

rule Specifies the name of an existing Matching Rule.

Guidelines

The **error-rule** command defines an error rule. An error rule is invoked in response to a processing error. An error rules requires a matching rule.

Create the matching rule with the **matching** command and populated it with the **httpmatch** or **urlmatch** commands. The matching rule serves as a source of URL or HTTP templates. Candidate documents that match any of the templates in the matching rule can be processed.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of Processing Policies.

Related Commands

exit

Examples

- Enters Processing Policy configuring mode to create the `faultProcess` error rule that uses the `faultMatch` Matching Rule.

```
# stylepolicy faultProcess
Processing Policy configuration
# error-rule faultMatch
```

filter

Identifies a default style sheet to filter documents.

Syntax

filter *URL*

Parameters

URL Specifies the location of the default style sheet.

Guidelines

This default style sheet performs XML filtering only if a candidate XML document fails to match any of the filter rules in the processing policy.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of Processing Policies.

Examples

- Identifies `validate.xsl` in the `store:` directory as the default style sheet to filter documents.

```
# filter store:///validate.xsl
#
```

---

match

Associates global rule with a matching rule and adds the rule-template pair.

Syntax

match *rule global-rule*

no match

Parameters

rule Specifies the name of an existing Matching Rule.

global-rule
Specifies the name of a transform or filtering rule.

Guidelines

Use the Global **matching** command to create a Matching Rule and populated it with the **httpmatch** or **urlmatch** commands. The matching rule serves as a source of URL or HTTP templates. Candidate documents that match any of the templates in the matching rule can be processed.

Use the Global **rule** command to create a global rule. The global rule defines processing procedures for documents that:

- Matches any of the expressions in the associated matching rule
- Matches the rule direction. A request-rule applies only to client-originated documents. A response-rule applies only to server-originated documents. A bidirectional rule applies to all documents regardless of source.

Use the **no match** command to reset a Processing Policy, which clear all rules from the Processing Policy.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of Processing Policies.

Examples

- Adds the associated matching rule and global rule to the current Processing Policy.

```
# match star valClientServer
#
```

- Remove all rules from the current Processing Policy.

```
# no match
#
```

request-rule

Assigns a request rule.

Syntax

request-rule *rule*

Parameters

rule Specifies the name of an existing Matching Rule.

Guidelines

The **request-rule** command defines a request rule. A request rule requires a matching rule. A request rule is applied to client-originated traffic only.

Create the matching rule with the **matching** command and populated it with the **httpmatch** or **urlmatch** commands. The matching rule serves as a source of URL or HTTP templates. Candidate documents that match any of the templates in the matching rule can be processed.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of Processing Policies.

Related Commands

exit

Examples

- Adds the `requestMatch` request rule.

```
# request-rule requestMatch
#
```

response-rule

Assigns a response rule.

Syntax

response-rule *rule*

Parameters

rule Specifies the name of an existing Matching Rule.

Guidelines

The **response-rule** command defines a request rule. A response rule requires a matching rule. A response rule is applied to server-originated traffic only.

Create the matching rule with the **matching** command and populated it with the **httpmatch** or **urlmatch** commands. The matching rule serves as a source of URL or HTTP templates. Candidate documents that match any of the templates in the matching rule can be processed.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of Processing Policies.

Related Commands

exit

Examples

- Adds the `responseMatch` response rule.

```
# response-rule responseMatch
#
```

---

rule

Assigns a bidirectional rule.

Syntax

rule *rule*

Parameters

rule Specifies the name of an existing Matching Rule.

Guidelines

The **rule** command defines a bidirectional rule. A bidirectional rule requires a matching rule. A bidirectional rule is applied to both client-originated and server-originated traffic.

Create the matching rule with the **matching** command and populated it with the **httpmatch** or **urlmatch** commands. The matching rule serves as a source of URL or HTTP templates. Candidate documents that match any of the templates in the matching rule can be processed.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of Processing Policies.

Related Commands

exit

Examples

- Adds the `bidiMatch` bidirectional rule.

```
# rule bidiMatch
#
```

xsldefault

Identifies a default style sheet to transform documents.

Syntax

xsldefault *URL*

Parameters

URL Specifies the location of the default style sheet.

Guidelines

This default style sheet performs XML transformation only if a candidate XML document fails to match any of the transformation rules that are defined in the processing policy.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for details about the creation and implementation of processing policies.

Examples

- Identifies `identity.xsl` in the `store:` directory as the default style sheet to transform documents.

```
# xsldefault store:///identity.xsl
#
```

Chapter 67. Processing Rule configuration mode

This chapter provides an alphabetic listing of commands that are available in Processing Rule configuration mode.

To enter this configuration mode, use the Global **rule** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Processing Rule configuration mode.

aaa

Adds an `aaa` action.

Syntax

aaa *input-context name* [*output-context*]

Parameters

input-context
Identifies the context that contains the document authenticated or authorized by the AAA policy that is implemented in this Processing Rule.

name Specifies the name of an existing AAA Policy.

output-context
Optionally identifies the context where any post processing output is stored. Use `OUTPUT` to specify the final policy output, that is the transformed client request or transformed server response.

Examples

- Applies the `AAA-Policy1` AAA Policy to the original input to the Processing Rule.

```
# aaa INPUT AAA-Policy-1
#
```

call

Adds a `call` action.

Syntax

call *input-context rule output-context*

Parameters

input-context
Identifies the context in which the specified rule is called. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

rule Specifies the name of the existing rule to invoke.

output-context
Identifies the context where results are stores. Use `OUTPUT` to specify the final policy output, that is the transformed client request or transformed server response

Examples

- Applies the `processRequest` rule to the document in the `temp1` context and moves the results to the `temp2` context.

```
# call temp1 processRequest temp2
#
```

checkpoint

Applies a `checkpoint` action.

Syntax

checkpoint *event* [*input-context*]

Parameters

checkpointEvent
Identifies the event that triggers the checkpoint and takes one of the following values:

`AuthComplete`
Indicates the completion of an authentication process.

`Fault` Indicates a fault condition.

`Request`
Indicates the input of a client-originated document.

`Response`
Indicates the input of a server-originated document.

input-context
Optionally identifies the context in which the checkpoint is triggered. In the absence of an explicit context assignment, the `INPUT` context is used by default.

Examples

- Adds a checkpoint that is triggered by a fault occurrence in the `temp1` context.

```
# checkpoint fault temp1
#
```

convert-http

Adds a `convert-http` action.

Syntax

convert-http *input-context output-context* [*map*]

Parameters

input-context
Identifies the context that contains the non-XML source. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

output-context
Identifies an output context where the converted document is stored. Use `OUTPUT` to specify the final policy output, that is the transformed client request or transformed server response

map Optionally identifies an input conversion map that specifies document encoding. In the absence a map designation, all input is treated as URL-escaped.

Examples

- Converts the original input to the rule to XML and puts the XML content in the `tempParams` context. The `ICM-normal` Input Conversion Map specifies the encoding of the non-XML input.

```
# convert-http INPUT tempParams ICM-normal
#
```

---

extract

Adds an `extract` action.

Syntax

extract *input-context output-context expression* [*variable*]

Parameters

input-context
Identifies the context to which the XPath expression is applied. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

output-context
Identifies the context that stores the result of the XPath expression. Use `OUTPUT` to specify the final policy output, that is the transformed client request or transformed server response

expression
Specifies the XPath expression that is applied and can be expressed in standard XPath format or as a variable that expands to an XPath expression.

variable
Optionally specifies a variable, within the output context, in which to store the result of the XPath expression. In the absence of this parameter, the results of the XPath expression are stored as the default contents (tree) of the destination context.

Examples

- Applies the XPath expression `//games/url` to the `INPUT` context and stores the result in the `three` context.

```
# extract INPUT three //games/url
#
```

- Applies the XPath expression `//games/url` to the `INPUT` context and stores the result in the variable `url` within the `three` context.

```
# extract INPUT three //games/url var://local/url
#
```

- Applies the XPath expression referenced by the local variable `xpath` and stores the result in the variable `url` in the `three` context.

```
# extract INPUT three var://local/xpath var://local/url
#
```

- Applies the XPath expression referenced by the local variable `xpath` and stores the result in the variable `url` in the `three` context. Note the explicit addressing of the optional variable.

```
# extract INPUT three var://local/xpath var://three/url
#
```

---

fetch

Adds a `fetch` action.

Syntax

fetch *url output-context*

Parameters

url Identifies the resource to be fetched and can be expressed as a URL or as a variable that expands to a URL.

output-context
Identifies the context in which to store the fetched resource.

Guidelines

You can use any protocol-specific URL when addressing the target resource.

Examples

- Retrieves the resource referenced by the variable `doc` from the default context and stores it in the `TestIt` context.

```
# fetch var://local/doc TestIt
#
```

- Retrieves the style sheet from the `store:` directory and stores it in the `count` context.

```
# fetch store:///count.xsl count
#
```

- Retrieves the specified style sheet and stores it in the `tmp1` context.

```
# fetch https://sona/TestBase/simple.xsl tmp1
#
```

---

filter

Adds a `filter` action.

Syntax

filter *input-context URL*

Parameters

input-context
Identifies the context that contains the document to be filtered. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

URL Identifies the XSL style sheet to filter the source document. Takes the form of a URL or a variable that expands to a URL.

Guidelines

Filters differ from transforms in that filters produce no output. Filters are generally used to propose conditions against which a candidate document is evaluated. They result in an accept or reject decision.

Filters are implemented by processing policies. A processing policy enables DataPower services to select an appropriate style sheet with which to filter or transform an input document. The selected style sheet can be used in conjunction with, or instead of, processing instructions contained within the input document.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for procedural details regarding the creation and implementation of Style Policies.

Related Commands

validate

Examples

- Uses the specified style sheet to filter the original input to the Processing Policy.

```
# filter INPUT store:///filter-1.xsl
#
```

- Uses the style sheet that is referenced by the `filter` variable in the `tools` context to filter the original input to the processing policy.

```
# filter INPUT var://context/tools/filter
#
```

input-filter

Specifies a decompression algorithm to apply to all incoming traffic before any other processing occurs.

Syntax

input-filter {**zip** | **pkzip** | **none**}

Parameters

zip Decompresses all incoming traffic with the ZIP algorithm.

pkzip Decompresses all incoming traffic with the PKZIP algorithm.

none (default) Does not decompress incoming traffic.

Guidelines

Use this command only if you are certain that all incoming traffic is compressed with the selected algorithm. Attempts to decompress non-compressed data will result in data corruption.

Examples

- Decompresses all incoming traffic with PKZIP.

```
# input-filter pkzip
#
```

- Restores the default state, where incoming traffic is not decompressed.

```
# input-filter none
#
```

log

Adds a `log` action.

Syntax

log *input-context destination* [*output-context*]

Parameters

input-context
Identifies the context whose contents are to be sent to a target location. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

destination
Specifies a URL for the log message recipient.

output-context
Optionally identifies an output context.

Examples

- Sends the contents of the INPUT context to the specified target URL.

```
# log INPUT http://www.us.ibm/ragnarok/log
#
```

non-xml-processing

Enables processing of non-XML input or output.

Syntax

non-xml-processing

no non-xml-processing

Guidelines

Use the **no non-xml-processing** command to disable non-XML processing.

Examples

- Enables non-XML processing in the Processing Rule.

```
# non-xml-processing
#
```

- Disables non-XML processing.

```
# no non-xml-processing
#
```

on-error

Adds an `on-error` action.

Syntax

on-error *mode* [*rule*] [*input-context*] [*output-context*]

Parameters

mode Specifies the operational response to an error and takes one of the following forms:

`abort` Indicates that processing ceases in the event of an error.

`continue`
Indicates that processing continues with the next action in the event of an error.

rule Optionally specifies an error rule that is executed in the event of an error condition.

input-context
Optionally identifies the input context for the error rule. If no context is explicitly identified, the input context of the failed action is used.

output-context
Optionally identifies the output context for the error rule. If no context is explicitly identified, the output context of the failed action is used.

Examples

- Specifies that rule processing ceases in the event of an error; calls the rule, `faultProcessing`, as an error handler.

```
# on-error abort faultProcessing
#
```

output-filter

Specifies a compression algorithm to apply to all outgoing traffic after all other processing occurs.

Syntax

output-filter {**zip** | **pkzip** | **none**}

Parameters

zip Compresses all incoming traffic with the ZIP algorithm.

pkzip Compresses all incoming traffic with the PKZIP algorithm.

none (Default) Does not compress outgoing traffic.

Examples

- Compresses all outgoing traffic with PKZIP.

```
# output-filter pkzip
#
```

- Restores the default state, outgoing traffic is not compressed.

```
# output-filter none
#
```

results

Adds a `results` action.

Syntax

results *context* [*destination*] [*response*]

Parameters

context Identifies the target context, that is the target whose contents are transmitted.

destination
Optionally specifies the destination. In the absence of this argument, the contents of the target context are transmitted to the OUTPUT of the Processing Rule.

response
Identifies the target context that stores the parsed reply. Required only when a response is expected from the destination, otherwise it is not used.

Examples

- Sends the contents of the `INPUT` context to the destination of the rule.

```
# results INPUT
#
```

- Sends the contents of the `INPUT` context to the destination referenced by the local `var://local/dest` variable.

```
# results INPUT var://local/dest
#
```

- Sends the contents of the `INPUT` context to the loopback server for processing. Processing results are stored in the `apple` context.

```
# results INPUT http://127.0.0.1:9000/ apple
#
```

results-async

Adds a `results-async` action.

Syntax

results *context destination*

Parameters

context Identifies the target context, that is the target whose contents are transmitted.

destination
Specifies the destination.

Guidelines

A `results-async` action differs from a `results` action in that `results-async` transmits the contents message asynchronously. That is, a `results-async` action never expects a response from the target destination.

Examples

- Sends the contents of the `INPUT` context to the destination of the rule.

```
# results INPUT
#
```

- Sends the contents of the `INPUT` context to the destination referenced by the local `var://local/dest` variable.

```
# results INPUT var://local/dest
#
```

- Sends the contents of the `INPUT` context to the loopback server for processing. Processing results are stored in the `apple` context.

```
# results INPUT http://127.0.0.1:9000/ apple
#
```

---

rewrite

Adds a `rewrite` action.

Syntax

rewrite *name*

Parameters

name Specifies the name of the URL Rewrite Policy.

Examples

- Rewrites the input URL with the `URLRewrite-1` policy.

```
# rewrite URLRewrite-1
#
```

---

route-action

Adds a `route-action` action.

Syntax

route-action *input-context URL*

route-action *input-context* **dynamic-stylesheet** *name*

Parameters

input-context
Identifies the context whose contents are to be routed by the specified style sheet. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

URL Identifies the local style sheet to route the contents of input context.

dynamic-stylesheet *name*
Identifies the dynamic style sheet for routing.

Examples

- Specifies style sheet-based routing of the contents of the `temp1` context with the `route.xsl` style sheet.

```
# route-action temp1 local:///route.xsl
#
```

route-set

Adds a `route-set` action.

Syntax

route-set *destination* [*proxy*]

Parameters

destination
: Identifies the document destination and can be expressed as a protocol-specific URL or as a variable that expands to a transport URL.

proxy
: Optionally specifies the name of an existing SSL Proxy Profile to secure the connection with the destination.

Examples

- Specifies a dynamic route expanded from the `dest-1` variable in the `destinations` context. The `DySSL-1` SSL Proxy Profile provides the credentials to establish a secure connection.

```
# route-set var://context/destinations/dest-1 DySSL-1
#
```

setvar

Adds a `setvar` action.

Syntax

setvar *context variable value*

Parameters

context
: Identifies the context in which the variable is set.

variable
: Specifies the name of the variable and takes the form of a `var://` URL.

value
: Specifies the value for the variable

Guidelines

If the `var://` URL is not of the local type, this value overrides the context specified by the context argument.

Examples

- Sets a variable in the `INPUT` context with the name of `dest` and a value of `http://ragnarok:9010/`.

```
# setvar INPUT var://local/dest http://ragnarok:9010/
#
```

- Sets a variable in the routing context with the name of dest and a value of http://ragnarok:9010/.

```
# setvar INPUT
 var://context/routing/dest http://ragnarok:9010/
#
```

slm

Adds an slm action.

Syntax

slm *input-context name*

Parameters

input-context
: Identifies the context monitored by the specified SLM Policy. Use INPUT to specify the initial policy input, that is the original client request or server response.

name
: Specifies the name of an existing SLM Policy.

Examples

- Assigns the SLM-1 SLM Policy to the INPUT context.

```
# slm INPUT SLM-1
#
```

strip-attachments

Adds a strip-attachments action.

Syntax

strip-attachments *context* [*uri*]

Parameters

context
: Identifies the context from which attachments are stripped.

uri
: Identifies a document attachment to strip. In the absence of a specified attachment, all attachments are stripped from the target context.

Examples

- Strips all attachments from the temp1 context.

```
# strip-attachments temp1
#
```

type

Enables the dynamic retyping.

Syntax

type {**error-rule** | **request-rule** | **response-rule** | **rule**}

Parameters

error-rule
Indicates an error rule, a rule invoked in response to a fault condition.

request-rule
Indicates a request rule, a rule applied to client requests only.

response-rule
Indicates a response rule, a rule applied to server responses only.

rule Indicates a bidirectional rule, a rule applied to both client requests and server responses.

Examples

- Classifies the rule as a request-rule.

```
# type request-rule
#
```

---

unprocessed

Enables data to passthrough subsequent actions in an unprocessed state.

Syntax

unprocessed

no unprocessed

Examples

- Enables unprocessed mode.

```
# unprocessed
#
```

- Disables unprocessed mode.

```
# no unprocessed
#
```

---

validate

Adds a `validate` action.

Syntax

validate *input-context* [*output-context*]

validate *input-context* **attribute-rewrite** *name* [*output-context*]

validate *input-context* **dynamic-schema** *url* [*output-context*]

validate *input-context* **schema** *url* [*output-context*]

validate *input-context* **wsdl** *url* [*output-context*]

Parameters

input-context
Specifies the context whose contents are to be validated.

attribute-rewrite *name*
Specifies the name of the URL Rewrite Policy to rewrite the schema that is referenced by an `xsi:schemaLocation` attribute in the XML document. The rewritten schema reference usually specifies the location of a local, trusted copy of the schema to use for document validation.

dynamic-schema *url*
Regardless of `xsi:schemaLocation` attributes in the document, specifies the use of a dynamically generated schema to use for document validation. *url* identifies the URL of the dynamic schema to use for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

schema *url*
Regardless of `xsi:schemaLocation` attributes in the document, specifies the URL of the schema to for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

schema-rewrite *url*
Specifies the URL of the base schema for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

name Specifies the name of the URL Rewrite Policy to apply to the schema URL. The rewritten URL identifies the schema to use for document validation.

wsdl-url *url*
Regardless of `xsi:schemaLocation` attributes in the document, specifies the URL of the WSDL file that contains the schema for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

output-context
Optionally specifies the output context of the validated document.

Guidelines

The **validate** command adds a `validate` action to the current processing rule. This action defines a policy-based XML schema validation filter.

If no methodology is identified, documents are validated in accordance with `xsi:schemaLocation` attributes in the specific context. Documents that do not contain these attributes are considered valid.

Related Commands

filter

Examples

- Adds a validation action. Validates the XML documents in the INPUT context with instruction in the document. Rewrites the document with the `URL-RW-1` URL Rewrite Policy. Uses the rewritten schema reference to validate the document.

```
# validate INPUT attribute-rewrite URL-RW-1
#
```

- Adds a validation action. Validates XML documents in the INPUT context with the schema that is referenced by the `var://context/schemas/1` variable.

```
# validate INPUT schema var://context/schemas/1
#
```

- Adds a validation action. Validates XML documents in the INPUT context with the local SchemaOne.xsd schema. Possibly stores the transformed document in the Post-Validation context.

```
# validate INPUT schema store:///SchemaOne.xsd Post-Validation
#
```

xform

Adds an xform action.

Syntax

xform *input-context URL output-context*

xform *input-context* **dynamic-stylesheet** *objectName output-context*

Parameters

input-context
Identifies the context that contains the document transformed by this rule. Use INPUT to specify the initial policy input, that is the original client request or server response.

URL Identifies the style sheet used to transform documents. Takes the form of a local or remote URL or a variable that expands to a URL.

dynamic-stylesheet *objectName*
Identifies the object (for example, an XPath Routing Map) from which the dynamic style sheet is generated.

output-context
Identifies the context that receives the transformed document. Use OUTPUT to specify the final policy output, that is the transformed client request or transformed server response

Guidelines

Transformations are implemented by Style Policies. A Style Policy enables DataPower server to select an appropriate style sheet with which to filter or transform an input document. The selected style sheet can be used in conjunction with, or instead of, processing instructions in the input document.

Related Commands

convert-http, **xformbin**, **xformpi**

Examples

- Adds a transformation rule. Transforms the original input to the Processing Policy Rule with the xform-1.xsl style sheet and sends the transformed document to the final output of the rule.

```
# xform INPUT store:///xform-1.xsl OUTPUT
#
```

- Adds a transformation rule. Transforms the original input to the processing rule with the style sheet that is referenced by the var://stylesheets/1 variable, and sends the transformed document to the Step2 context.

```
# xform INPUT var://stylesheets/1 Step2
#
```

- Adds a transformation rule. Transforms the document in the Step2 context with the style sheet that is referenced by the var://stylesheets/5 variable, and sends the transformed document to the final destination of the rule.

```
# xform Step2 var://stylesheets/5 OUTPUT
#
```

xformbin

Adds an xformbin action.

Syntax

xformbin *input-context URL output-context*

xformbin *input-context* **dynamic-stylesheet** *object output-context*

Parameters

input-context
Identifies the context that contains the binary document transformed by this rule. Use INPUT to specify the initial policy input, that is the original client request or server response.

URL Identifies the stylesheet used to transform documents. Takes the form of a local or remote URL or a variable that expands to a location.

dynamic-stylesheet *object*
Identifies the object (for example, an XPath Routing Map) from which the dynamic style sheet is generated.

output-context
Identifies the context that receives the transformed document. Use OUTPUT to specify the final policy output, that is the transformed client request or transformed server response.

Related Commands

convert-http, **xform**, **xformpi**

Examples

- Adds a transformation rule. Transforms the original binary input to XML (with the local binToXML.xsl style sheet) and sends the XML to the final output of the rule.

```
# xformbin INPUT store:///binToXML.xsl OUTPUT
#
```

- Adds a transformation rule. Transforms the original binary input to XML (with the style sheet that is referenced by the var://stylesheets/bin-xform variable, and sends the transformed document to the FromBin context.

```
# xformbin INPUT var://stylesheets/bin-xform FromBin
#
```

xformpi

Adds an xformpi action.

Syntax

xformpi *input-context URL output-context*

xformpi *input-context* **dynamic-stylesheet** *object output-context*

Parameters

input-context
Identifies the context that contains the document transformed by this rule. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

URL Identifies the style sheet to transform documents that lack internal processing instructions. Takes the form of a local or remote URL or a variable that expands to a location.

dynamic-stylesheet *object*
Identifies the object (for example, an XPath Routing Map) from which the dynamic style sheet is generated.

output-context
Identifies the context that receives the transformed document. Use `OUTPUT` to specify the final policy output, that is the transformed client request or transformed server response.

Guidelines

XSL transformations are implemented by Processing Policies. A Processing Policy enables DataPower service to select an appropriate style sheet with which to filter or transform an input document. The selected style sheet can be used in conjunction with, or instead of, processing instructions in the input document.

Related Commands

convert-http, **xformbin**, **xform**

Examples

- Adds a transformation rule. Transform the original input to the Processing Policy Rule with internal processing instructions in the XML document and send the transformed document to the final output of the rule. Uses `identity.xsl` style sheet in the `store:` directory to perform the transform if the document does not contain processing instructions.

```
# xformpi INPUT store:///identity.xsl OUTPUT
#
```

Chapter 68. RADIUS configuration mode

This chapter provides an alphabetic listing of commands that are available in RADIUS configuration mode.

To enter this configuration mode, use the Global **radius** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in RADIUS configuration mode.

aaaserver

Identifies a RADIUS AAA server.

Syntax

aaaserver *position address* [*port*]

no aaaserver *position*

Parameters

position
Specifies the relative position of this server on the list of RADIUS AAA servers.

address Specifies the IP address of a RADIUS AAA server.

port Optionally specifies the port number on IP address that monitors RADIUS AAA requests. Use an integer in the range of 0 to 65535. The default is 1812.

Guidelines

The **aaaserver** command identifies a RADIUS AAA server, prompts for the secret unique to this appliance-RADIUS server pair, and adds the server to the list of RADIUS servers.

Use *position* to determine the order in which the appliance contacts RADIUS AAA servers. For example, assume that an appliance has a list of 3 RADIUS AAA servers, numbered 10, 20, and 30. RADIUS AAA requests are always first sent to server 10 (the lowest server number being the most preferred). Should the authentication request timeout, it is next sent to server 20. Should that request timeout, the request is sent to server 30 (the highest server number is the least preferred).

Use the **no aaaserver** command to delete a server from the list of RADIUS servers.

Examples

- Identifies a RADIUS AAA server at 172.16.1.1:1812.

```
# aaaserver 20 172.16.1.1 1812
 secret: YetAnotherPasswordServer20
#
```

- Identifies a RADIUS server at 172.16.100.100:1812

```
# aaaserver 30 172.16.100.100 1812
 secret: YetAnotherPasswordServer20
#
```

- Identifies a RADIUS server at 172.16.200.200:18120. RADIUS servers will be contacted in the following order: 172.16.200.200 18120, 172.16.1.1 1812, 172.16.100.100 1812.

```
# aaaserver 10 172.16.200.200 18120
 secret: YetAnotherPasswordServer10
#
```

- Deletes the RADIUS server at 172.16.200.200:18120. RADIUS servers will be contacted in the following order: 172.16.1.1 1812, 172.16.100.100 1812.

```
# no server 172.16.200.200
 secret: YetAnotherPasswordServer10
#
```

---

id

Specifies the NAS-identifier.

Syntax

id *value*

Parameters

value Identifies the appliance when acting as a RADIUS client.

Guidelines

The NAS-identifier (defined in Section 5.32 of RFC 2865) can be used in some RADIUS environments in the place of an IP address to identify a client appliance. It consists of one or more octets and must be unique within the scope of the RADIUS server.

The FQDN (fully qualified domain name) of the RADIUS client is often used as the NAS-Identifier.

Related Commands

server

Examples

- Specifies a NAS-identifier.

```
# id ragnarok.datapower.com
```

---

retries

Specifies the number of times the appliance retransmits an unanswered request.

Syntax

retries *number*

Parameters

number
Specifies the number of re-transmittals. The default is 3.

Guidelines

In conjunction with the **timeout** command, the **retries** command specifies the maximum amount of time that the appliance spends attempting to connect to a specific RADIUS server. At the expiration of this period, the appliance attempts to connect to the next server on its list of RADIUS servers.

The retries value is a global one that applies to all RADIUS servers known to the appliance.

Related Commands

timeout

Examples

- Specifies a retry value of 5.

```
# retries 5
#
```

---

server

Identifies a RADIUS server.

Syntax

server *position address* [*port*]

no server *position*

Parameters

position
Specifies the relative position of this server on the list of RADIUS servers.

address Specifies the IP address of a RADIUS server.

port Optionally identifies the port number on IP address that monitors RADIUS requests. Use an integer in the range of 0 to 65535. The default is 1812.

Guidelines

The **server** command identifies a RADIUS server, prompts for the secret unique to this appliance-RADIUS server pair, and adds the server to the list of RADIUS servers.

Use *position* to determine the order in which the appliance contacts RADIUS servers. For example, assume that an appliance has a list of 3 RADIUS servers, numbered 10, 20, and 30. RADIUS authentication requests are always first sent to server 10 (the lowest server number being the most preferred). Should the authentication request timeout, it is next sent to server 20. Should that request timeout, the request is sent to server 30 (the highest server number is the least preferred).

Use the **no server** command to delete a server from the list of RADIUS servers.

Examples

- Identifies a RADIUS server at 172.16.1.1:1812.

```
# server 20 172.16.1.1 1812
 secret: YetAnotherPasswordServer20
#
```

- Identifies a RADIUS server at 172.16.100.100:1812

```
# server 30 172.16.100.100 1812
 secret: YetAnotherPasswordServer20
#
```

- Identifies a RADIUS server at 172.16.200.200:18120. RADIUS servers will be contacted in the following order: 172.16.200.200 18120, 172.16.1.1 1812, 172.16.100.100 1812.

```
# server 10 172.16.200.200 18120
 secret: YetAnotherPasswordServer10
#
```

- Deletes the RADIUS server at 172.16.200.200:18120. RADIUS servers will be contacted in the following order: 172.16.1.1 1812, 172.16.100.100 1812.

```
# no server 172.16.200.200
 secret: YetAnotherPasswordServer10
#
```

---

timeout

Specifies the retransmit interval.

Syntax

timeout *milliseconds*

Parameters

milliseconds
Specifies the number of milliseconds to wait for a reply from a RADIUS server before retransmitting the outstanding request. The default is 1000.

Guidelines

The **timeout** command specifies the RADIUS retransmit interval. This interval is the number of seconds that the appliance waits for a reply from a RADIUS server before retransmitting the outstanding request.

The timeout value is a global one that applies to all RADIUS servers known to the appliance.

In conjunction with the **retries** command, the **timeout** command specifies the maximum amount of time that the appliance spends attempting to connect to a specific RADIUS server. At the expiration of this period, the appliance will attempt to connect to the next server on its list of RADIUS servers.

Related Commands

retries

Examples

- Specifies a retransmit interval of 1/2 second (500 milliseconds).

```
# timeout 500
#
```

Chapter 69. RBM Settings configuration mode

This chapter provides an alphabetic listing of commands that are available in RBM Settings configuration mode. All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in RBM configuration mode.

To enter this configuration mode, use the Global **rbm** command. In this configuration mode, you can define both RBM settings as well as the global password policy for all locally-defined users.

- To view the current configuration, use the **show** command.
- To restore default settings, use the **reset** command.
- To exit this configuration mode and save configuration changes to the running configuration, use the **exit** command. When you save changes to the running configure, these changes takes affect immediately. At this point, the new settings could disable access to the DataPower appliance for any user who does not have an active session (WebGUI, command line, Telnet, or serial port connection). In other words, the changes could disable future access through any of the following methods:
  - Any user who attempts to access the appliance through the WebGUI
  - Any user who attempts to access the appliance through the command line
  - Any user who attempts to access the appliance through a Telnet session
  - Any user who attempts to access the appliance through the serial port connection (WebGUI or command line)

apply-cli

Specifies whether the RBM policy applies to command line access.

Syntax

apply-cli {**on** | **off**}

Parameters

on Applies the RBM policy to command line access. The access profile applies to both WebGUI and command line access.

off (Default) Does not apply the RBM policy to command line access. The access profile applies to WebGUI access only. Command line access is controlled by the specified command groups for the user group and by the specified user accounts for application domains.

Guidelines

The **apply-cli** command controls whether the RBM policy applies to both WebGUI access and command line access or applies to WebGUI access only. When enabled, command groups assignment that is part of the User Group configuration and user access that is part of Application Domain configuration are ignored.

Note: Do not enable this option when the authentication method, as defined with the **au-method** command, is **client-ssl**. If you enable this option and use SSL

client certificates for authentication, only local fallback users, as defined with the **fallback-login** and **fallback-users** commands, will be able to access the appliance from the command line.

Related Commands

access-policy (User Group), **add** (User Group), **au-method**, **delete** (User Group), **domain-user** (Application Domain), **fallback-login**, **fallback-users**

Examples

- Applies the RBM policy to the WebGUI access and command line access.

```
# apply-cli on
#
```

- Removes the RBM policy from command line access. Does not affect WebGUI access.

```
# apply-cli off
#
```

---

au-cache-mode

Sets the caching mode for authentication results.

Syntax

au-cache-mode {**absolute** | **disabled** | **maximum** | **minimum**}

Parameters

absolute
Caches the results of user authentications for a period of time specified by the **au-cache-ttl** command (the explicit time-to-live).

disabled
Disables caching. The system will not cache any results and instead always authenticate every time a user requests access.

maximum
(Default) Compares the explicit TTL to the TTL contained in the response (if any) and cache authentication results for the maximum of the two values.

minimum
Compares the explicit TTL to the TTL contained in the response (if any) and cache authentication results for the minimum of the two values

Guidelines

The **au-cache-mode** command establishes the desired caching mode for authentication results. Use the **au-cache-ttl** command to establish the explicit TTL.

Related Commands

au-cache-ttl

Examples

- Caches authentication results for the maximum amount of time.

```
# au-cache-mode maximum
#
```

au-cache-ttl

Specifies the time-to-live for cached authentication results.

Syntax

au-cache-ttl *seconds*

Parameters

seconds

Specifies the time-to-live (TTL) in seconds. Use an integer in the range of 1 through 86400. The default is 600.

Guidelines

The **au-cache-ttl** command defines the explicit TTL in seconds for cached authentication results. This value is compared against the TTL in the authentication response in accordance with the cache mode, as defined with the **au-cache-mode** command.

Related Commands

au-cache-mode

Examples

- Sets the TTL to five minutes.

```
# au-cache-ttl 300
#
```

au-custom-url

Specifies the URL of the custom style sheet.

Syntax

au-custom-url *URL*

Parameters

URL Specifies the location of the style sheet.

Guidelines

The **au-custom-url** command defines the fully-qualified file name (URL) of the custom style sheet for authentication. This command is relevant when the authentication method, as defined with the **au-method** command, is **custom**.

Related Commands

au-method

Examples

- Identifies the `RBM-AU.xsl` style sheet in the `authn` directory of the `myserver.domain.com` server as the style sheet for custom authentication. File retrieval uses the HTTPS protocol.

```
# au-method custom
# au-custom-url https://myserver.domain.com/authn/RBM-AU.xsl
#
```

au-info-url

Specifies the URL of the authentication XML file.

Syntax

au-info-url *URL*

Parameters

URL Specifies the location of the XML file.

Guidelines

The **au-info-url** command defines the fully-qualified file name (URL) of the XML file for authentication. This command is relevant when the authentication method, as defined with the **au-method** command, is **xmlfile**.

Related Commands

au-method

Examples

- Identifies the `RBM-AU.xml` file in the `local:` directory as the authentication XML file.

```
# au-method xmlfile
# au-info-url local:///RBM-AU.xml
#
```

au-kerberos-keytab

Assigns the keytab for SPNEGO user authentication.

Syntax

au-kerberos-keytab *name*

Parameters

name Specifies the name of an existing Kerberos Keytab object.

Guidelines

The **au-kerberos-keytab** command is meaningful only when the authentication method, as defined with the **au-method** command, is **spnego**. A keytab (or key table) is an unencrypted file that contains a list of Kerberos principals and their passwords.

Use the Crypto **kerberos-keytab** command to create a Kerberos Keytab object.

Related Commands

au-method, **kerberos-keytab** (Crypto)

Examples

- Assigns the keytab-1 Kerberos Keytab object for SPNEGO authentication.

```
# au-method spnego
# au-kerberos-keytab keytab-1
#
```

au-ldap-bind-dn

Specifies the login DN (distinguished name) to access an LDAP server.

Syntax

au-ldap-bind-dn *DN*

Parameters

DN Specifies the login DN.

Guidelines

The **au-ldap-bind-dn** command specifies the login DN to access the target LDAP server. This command is relevant when the authentication method, as defined with the **au-method** command, is **ldap** and when the LDAP search for group name property, as defined with the **au-ldap-search** command, is enabled.

Beyond specifying the login DN when searching the LDAP for the group name, you need to use the **au-ldap-bind-password** command to specify the user's password and optionally use the **au-ldap-parameters** command to associate an existing LDAP Search Parameters object.

Related Commands

au-ldap-bind-password, **au-ldap-parameters**, **au-ldap-search**, **au-method**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
#
```

au-ldap-bind-password

Specifies the password to access an LDAP server.

Syntax

au-ldap-bind-password *password*

Parameters

password
Specifies the password for the login DN.

Guidelines

The **au-ldap-bind-password** command specifies the password for the login DN to access the target LDAP server. This command is relevant when the authentication method, as defined with the **au-method** command, is **ldap** and when the LDAP search for group name property, as defined with the **au-ldap-search** command, is enabled.

Beyond specifying the login DN when searching the LDAP for the group name, you need to use the **au-ldap-bind-password** command to specify the user's password and optionally use the **au-ldap-parameters** command to associate an existing LDAP Search Parameters object.

Related Commands

au-ldap-bind-dn, **au-ldap-parameters**, **au-ldap-search**, **au-method**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
#
```

---

au-ldap-parameters

Assigns the LDAP Search Parameters to perform an LDAP search.

Syntax

au-ldap-parameters *name*

Parameters

name Specifies the name of an existing LDAP Search Parameters object.

Guidelines

The **au-ldap-parameters** command assigns the LDAP Search Parameters object to perform an LDAP search. The search retrieves the user's distinguished name (DN).

This command is relevant only when LDAP search is enabled with the **au-ldap-search** command and when the authentication method is LDAP, as defined with the **au-method** command.

This command is not relevant in any other situation.

Related Commands

au-ldap-search, **au-method**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
# au-ldap-parameters ldap1-AU
#
```

au-ldap-search

Indicates whether to retrieve the distinguished name (DN) with an LDAP search.

Syntax

au-ldap-search {**on** | **off**}

Parameters

on Enables an LDAP search for the user's DN. The login name of the user along with the LDAP Search Parameters will be used as part of an LDAP search to retrieve the user's DN.

off (Default) Disables an LDAP search for the user's DN. The login name of the user along with the LDAP prefix and the LDAP suffix will be used to construct the user's DN.

Guidelines

The **au-ldap-search** command indicates whether to retrieve the distinguished name with an LDAP search.

- When enabled, use the **au-ldap-bind-dn** command to specify the user's DN to access the LDAP server, the **au-ldap-bind-password** command to specify the user's password, and optionally use the **au-ldap-parameters** command to associate an existing LDAP Search Parameters object.
- When disabled, use the **ldap-prefix** command to specify the LDAP prefix to add to the user name, and use the **ldap-suffix** command to specify the LDAP suffix to append to the user name. The provided prefix and suffix form the DN to submit to the LDAP server.

This command is relevant when the authentication method, as defined with the **au-method** command, is **ldap**.

Related Commands

au-ldap-bind-dn, **au-ldap-bind-password**, **au-method**, **au-ldap-parameters**, **ldap-prefix**, **ldap-suffix**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
# au-ldap-parameters ldapParams-1
```

- Identifies LDAP authentication without optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search off
# au-ldap-prefix "cn="
# au-ldap-suffix "O=example.com"
#
```

au-method

Specifies the authentication method.

Syntax

au-method {**client-ssl** | **custom** | **ldap** | **local** | **radius** | **spnego** | **xmlfile** | **zosnss**}

Parameters

client-ssl
Uses a SSL certificate from a connection peer. Requires an **au-valcred** value.

custom
Uses a custom file or method. Requires an **au-custom-url** value.

ldap Uses an LDAP server. Requires the **au-server-host** and **au-server-port** values.

local (Default) Uses the user configuration that is maintained on the local system. Does not access external resources.

radius Uses a RADIUS server.

spnego
Uses a SPNEGO server. Requires the **au-kerberos-keytab** value.

xmlfile
Uses a locally stored AAA Info file. Requires an **au-info-url** value.

zosnss Uses an NSS server. Requires the **au-zos-nss** value.

Guidelines

The **au-method** command sets the authentication method for RBM. The selected method must be fully configured before invoking this command.

If the `admin` account is not configured with all permissions, the `admin` account is locked out of the WebGUI. Use the command line to change this circumstance.

Related Commands

au-info-url, **au-custom-url**, **au-info-url**, **au-kerberos-keytab**, **au-server-host**, **au-server-port**, **au-valcred**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN and associates SSL Proxy Profile `profile-1` for secure communication.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# ldap-sslproxy profile-1
```

```
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
#
```

- Set the authentication method to local.

```
# au-method local
#
```

au-server-host

Specifies the IP address or domain name of a remote authentication server.

Syntax

au-server-host *host*

Parameters

host Specifies the IP address or domain name of the server.

Guidelines

The **au-server-host** command specifies the IP address or domain name of the authentication server. When the authentication method is **ldap**, as defined with the **au-method** command, you need to define the LDAP server in one of the following ways:

- The **au-server-host** and **au-server-port** commands
- The **loadbalancer-group** command

Related Commands

au-method, **au-server-port**, **loadbalancer-group**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
#
```

au-server-port

Specifies the port on the remote authentication server.

Syntax

au-server-port *port*

Parameters

port Specifies the port number of the authentication server.

Guidelines

The **au-server-port** command specifies the listening port of the authentication server defined with the **au-server-host** command. When the authentication method is **ldap**, as defined with the **au-method** command, you need to define the LDAP server in one of the following ways:

- The **au-server-host** and **au-server-port** commands
- The **loadbalancer-group** command

Related Commands

au-method, **au-server-host**, **loadbalancer-group**

Examples

- Identifies LDAP authentication with optional retrieval of the group DN.

```
# au-method ldap
# au-server-host ldap-1
# au-server-port 389
# au-ldap-search on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
#
```

---

au-zos-nss

Assigns a z/OS NSS Client for authentication with the NSS server.

Syntax

au-zos-nss *name*

Parameters

name Specifies the name of an existing z/OS NSS Client object.

Guidelines

The **au-zos-nss** command is meaningful only when the authentication method, as defined with the **au-method** command, is **zosnss**. The z/OS NSS Client object defines all parameters necessary to authenticate with the NSS server.

Use the Global **zos-nss** command to create a z/OS NSS Client object.

Related Commands

au-method, **zos-nss** (Global)

---

au-valcred

Assigns a Validation Credentials for SSL client certificate authentication.

Syntax

au-valcred *name*

Parameters

name Specifies the name of an existing Validations Credentials object.

Guidelines

The **au-valcred** command associates a Validation Credentials object for validating the identity presented in a client certificate from an SSL peer. This command is relevant when the authentication method, as defined with the **au-method** command, is **client-ssl**.

Use the Crypto **valcred** command to create a Validation Credentials object.

Related Commands

au-method, **valcred** (Crypto)

Examples

- Assigns the `valCred-1` Validation Credentials object for SSL client certificate authentication.

```
# au-method client-ssl
# au-valcred valCred-1
#
```

cli-timeout

Specifies the amount of idle time before closing a command line session because of inactivity.

Syntax

cli-timeout *seconds*

Parameters

seconds

Specifies the timeout value of the idle session in seconds. Use an integer in the range of 0 through 65535. The default is 0, which disables the timeout function.

Guidelines

The **cli-timeout** command specifies the amount of idle time in seconds before closing a command line session because of inactivity. When the session times out, you must reestablish a session and re-authenticate.

This command manages the session timeout for the command line. The **idle-timeout** command in Web Management Service mode manages the session timeout for the WebGUI.

Related Commands

idle-timeout (Web Management Service)

fallback-login

Specifies whether to use local users if the primary authentication method fails.

Syntax

fallback-login {**disabled** | **local** | **restricted**}

Parameters

disabled
(Default) Indicates that no locally-defined user can log in.

local Indicates that all locally-defined users can log in.

restricted
Indicates that only specific locally-defined users can log in.

Guidelines

The **fallback-login** command indicates whether to use local user accounts as fallback users when the primary authentication method fails. With fallback users, locally-defined users can log in to the appliance if the authentication method fails or in the event of a network outage that affects the primary authentication.

To limit fallback users to a specific set, use the **restricted** keyword. In this case, use the **fallback-user** command to define the specific, locally-defined users to allows as fallback users.

The **fallback-login** command is relevant only when remote authentication. In other words, this command is relevant when the setting for the **au-method** is any value except **local**.

Related Commands

au-method, **fallback-user**

Examples

- Allows all locally-defined users to log in.

```
# fallback-login local
#
```

- Designates `bobsmith` and `joselopez` as fallback users.

```
# fallback-login restricted
# fallback-user bobsmith
# fallback-user joselopez
#
```

- Disallows all locally-defined users from logging in.

```
# fallback-login disabled
#
```

fallback-user

Adds a locally-defined user as a fallback user.

Syntax

fallback-user *user*

no fallback-user *user*

Parameters

user Specifies the name of an existing user.

Guidelines

The **fallback-user** command allows a locally-defined user to be a fallback user. Invoke the **fallback-user** command for each fallback user.

This command is relevant when the **fallback-login** command is **restricted**.

Use the **no fallback-user** command to remove a user from the list of fallback users.

Related Commands

fallback-login

Examples

- Designates `bobsmith` and `joselopez` as fallback users.

```
# fallback-login restricted
# fallback-user bobsmith
# fallback-user joselopez
#
```

ldap-prefix

Specifies the LDAP prefix to add to the user name to form the DN.

Syntax

ldap-prefix *prefix*

Parameters

prefix Specifies an LDAP prefix.

Guidelines

The **ldap-prefix** command specifies the string to add as a prefix to the user name to form the distinguished name (DN) for LDAP authentication. The LDAP prefix and the user name are separated with a comma, and both are included within quotes.

For example, if the LDAP prefix is `cn=` and the user name is `Bob Smith`, then the beginning portion of the DN is `cn=Bob Smith`.

This command is relevant only when the **au-ldap-search** command is **off**.

Related Commands

au-method, **au-ldap-search**, **ldap-suffix**

Examples

- Sets the LDAP prefix to `cn=`.

```
# ldap-prefix "cn="
#
```

ldap-sslproxy

Assigns the SSL Proxy Profile for LDAP authentication.

Syntax

ldap-sslproxy *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ldap-sslproxy** command assigns an existing SSL Proxy Profile to secure communication with the LDAP server during LDAP authentication. When specified, LDAP communication uses the configuration in the assigned SSL Proxy Profile. If not specified, the communication is nonsecure.

This command is relevant only when the authentication method, as specified with the **au-method** command, is **ldap**.

Related Commands

au-method

Examples

- Uses the `ldapone` SSL Proxy Profile for secure communications.

```
# ldap-sslproxy ldapone
#
```

---

ldap-suffix

Specifies the LDAP suffix to add to the user name to form the DN.

Syntax

ldap-suffix *suffix*

Parameters

suffix Specifies an LDAP suffix.

Guidelines

The **ldap-suffix** command specifies the string to add after the user name to form the base distinguished name (DN) for LDAP authentication. The LDAP suffix and the user name are separated with a comma, and both are included within quotes.

For example, if LDAP suffix is `O=example.com` and the user name is `Bob`, the DN is `CN=Bob,O=example.com`.

This command is relevant only when the **au-ldap-search** command is **off**.

Related Commands

au-method, **au-ldap-search**, **ldap-prefix**

Examples

- Sets the LDAP suffix to `O=example.com`.

```
# ldap-suffix "O=example.com"
#
```

ldap-version

Specifies the LDAP version.

Syntax

ldap-version {**v2** | **v3**}

Parameters

v2 (Default) Uses LDAP version 2.

v3 Uses LDAP version 3.

Guidelines

The **ldap-version** command specifies the LDAP version for LDAP authentication. This command is relevant only when the authentication method is **ldap**, as defined with the **au-method** command.

Related Commands

au-method

Examples

- Sets the LDAP version to v3.

```
# ldap-version v3
#
```

loadbalancer-group

Assigns a load balancer group for LDAP authentication.

Syntax

loadbalancer-group *name*

Parameters

name Specifics the name of an existing load balancer group.

Guidelines

The **loadbalancer-group** command assigns a load balancer group for LDAP authentication. When the authentication method is **ldap**, as defined with the **au-method** command, you need to define the LDAP server in one of the following ways:

- The **au-server-host** and **au-server-port** commands
- The **loadbalancer-group** command

To create a new load balancer group, use the Global **loadbalancer** command.

Related Commands

au-method, **au-server-host**, **au-server-port**, **loadbalancer** (Global), **show loadbalancer-status**

Examples

- Sets the LDAP load balancer to LBGroup1.

```
# au-method ldap
# loadbalancer-group LBGroup1
# au-ldap-serach on
# au-ldap-bind-dn proxyuser
# au-ldap-bind-password p@Ssw0rd
#
```

lockout-duration

Specifies the duration to lock out the local account.

Syntax

lockout-duration *minutes*

Parameters

minutes

Specifies the number of minutes to lock out an account after exceeding the maximum number of failed login attempts. A value of 0 indicates that accounts are locked out until reset by a privileged administrator. Use an integer in the range of 0 through 1000. The default is 1.

Guidelines

The **lockout-duration** command specifies the duration to lock out an account in minutes after exceeding the permitted number of failed login attempts defined by the **max-login failure** command. Instead of locking out an account for a specific duration, the account can be locked out until re-enabled by a privileged administrator. To lock out accounts until reset, set the duration to 0.

Note: The **lockout-duration** commands applies to all accounts including the admin account. The only difference is that the admin account cannot be locked out until reset. When the duration is 0, the admin account is locked out for 120 minutes or until re-enabled by another administrator.

Related Commands

max-login-failure

Examples

- Enables lockout behavior for accounts that on the fifth login failure, the account is locked out locked out until reset by a privileged administrator:

```
# lockout-duration 0
# max-login-failure 4
```

max-login-failure

Whether to lock out a local user account after a specific number of failed login attempts.

Syntax

max-login-failure *count*

Parameters

count Specifies the maximum number of failed login attempts to allow before lockout. A value of 0 disables account lockout. Use an integer in the range of 0 through 64. The default is 3.

Guidelines

The **max-login-failure** command defines the number of failed login attempts to permit before a successful login. If the value is 3 and the user has failed three consecutive login attempts, the behavior on the next login attempt for this user is as follows:

- If failure, the account is locked out. The duration of the lockout depends on the value defined by the **lockout-duration** command.
- If successful, the account is not locked out and the count is reset.

Note: The **max-login failure** command applies to all accounts including the `admin` account. The only difference is that the `admin` account cannot be locked out until reset. When the duration is 0, the `admin` account is locked out for 120 minutes or until re-enabled by another administrator.

Related Commands

lockout-duration

Examples

- Enables lockout behavior for accounts that on the fifth login failure, the account is locked out locked out until reset by a privileged administrator:

```
# lockout-duration 0
# max-login-failure 4
```

- Disables lockout behavior.

```
# max-login failure 0
```

---

mc-custom-url

Specifies the URL of the custom style sheet.

Syntax

mc-custom-url *URL*

Parameters

URL Specifies the location of the style sheet.

Guidelines

The **mc-custom-url** command defines the fully-qualified file name (URL) of the custom style sheet for mapping credentials method. This command is relevant when the mapping credentials method, as defined with the **mc-method** command, is **custom**.

Related Commands

mc-method

Examples

- Identifies the RBM-MC.xsl style sheet in the mapCred directory of the myserver.domain.com server as the style sheet for custom authentication. File retrieval uses the HTTPS protocol.

```
# mc-method custom
# mc-custom-url https://myserver.domain.com/mapCred/RBM.xsl
#
```

---

mc-info-url

Specifies the URL of the mapping credentials XML file.

Syntax

mc-info-url *URL*

Parameters

URL Specifies the location of the XML file.

Guidelines

The **mc-info-url** command defines the fully-qualified file name (URL) of the XML file for credentials mapping. This command is relevant when the mapping credentials method, as defined with the **mc-method** command, is **xmlfile**.

Related Commands

mc-method

Examples

- Identifies the RBM-MC.xml in the local: directory.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
#
```

---

mc-ldap-bind-dn

Specifies the login DN (distinguished name) to access an LDAP server.

Syntax

mc-ldap-bind-dn *DN*

Parameters

DN Specifies the login name to access the target LDAP server.

Guidelines

The **mc-ldap-bind-dn** command specifies the login DN to access the target LDAP server.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

Beyond specifying the login DN when searching the LDAP for the group name, you need to use the following properties:

- How to connect to the LDAP server. Use either approach:
  - The **mc-server-host** and **mc-server-port** commands
  - The **mc-loadbalancer-group** command
- Optionally associate an existing SSL Proxy Profile object to use secure communication with the LDAP server with the **mc-ldap-sslproxy** command
- Specify the user's password with the **mc-ldap-bind-password** command
- Optionally associate an existing LDAP Search Parameters object with the **mc-ldap-parameters** command

Related Commands

mc-ldap-bind-password, **mc-ldap-parameters**, **mc-ldap-search**, **mc-ldap-sslproxy**, **mc-loadbalancer-group**, **mc-server-host**, **mc-server-port**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-server-host ldap.mydomain.com
# mc-server-port 389
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
# mc-ldap-parameters ldap1-MC
#
```

---

mc-ldap-bind-password

Specifies the password for the login DN to access an LDAP server.

Syntax

mc-ldap-bind-password *password*

Parameters

password
Specifies the password for the login DN.

Guidelines

The **mc-ldap-bind-password** command specifies the password for the login DN to access the target LDAP server.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

Beyond specifying the password for the login DN when searching the LDAP for the group name, you need to use the following properties:

- How to connect to the LDAP server. Use either approach:
  - The **mc-server-host** and **mc-server-port** commands

- The **mc-loadbalancer-group** command

- Optionally associate an existing SSL Proxy Profile object to use secure communication with the LDAP server with the **mc-ldap-sslproxy** command
- Specify the login DN to access the LDAP server with the **mc-ldap-bind-dn** command
- Optionally associate an existing LDAP Search Parameters object with the **mc-ldap-parameters** command

Related Commands

mc-ldap-bind-dn, **mc-ldap-parameters**, **mc-ldap-search**, **mc-ldap-sslproxy**, **mc-loadbalancer-group**, **mc-server-host**, **mc-server-port**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-server-host ldap.mydomain.com
# mc-server-port 389
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
# mc-ldap-parameters ldap1-MC
#
```

mc-ldap-parameters

Assigns the LDAP Search Parameters to perform an LDAP search.

Syntax

mc-ldap-parameters *name*

Parameters

name Specifies the name of an existing LDAP Search Parameters object.

Guidelines

The **mc-ldap-parameters** command assigns the LDAP Search Parameters object to perform an LDAP search. The search retrieves the user's group.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

Beyond associating an existing LDAP Search Parameters object when searching the LDAP for the group name, you need to use the following properties:

- How to connect to the LDAP server. Use either approach:
  - The **mc-server-host** and **mc-server-port** commands
  - The **mc-loadbalancer-group** command
- Optionally associate an existing SSL Proxy Profile object to use secure communication with the LDAP server with the **mc-ldap-sslproxy** command

- Specify the login DN to access the LDAP server with the **mc-ldap-bind-dn** command
- Specify the user's password with the **mc-ldap-bind-password** command

Related Commands

mc-ldap-bind-dn , **mc-ldap-bind-password**, **mc-ldap-search**, **mc-ldap-sslproxy**, **mc-loadbalancer-group**, **mc-method**, **mc-server-host**, **mc-server-port**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-server-host ldap.mydomain.com
# mc-server-port 389
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
# mc-ldap-parameters ldap1-MC
#
```

---

mc-ldap-search

Indicates whether to retrieve the group name with an LDAP search.

Syntax

mc-ldap-search {**on** | **off**}

Parameters

on Enables an LDAP search for the user's group. The authenticated DN of the user along with the LDAP Search Parameters will be used as part of an LDAP search to retrieve the user's group.

off (Default) Disables an LDAP search for the user's group. The authenticated identity of the user (DN or user group of local user) will be used directly as the input credential.

Guidelines

The **mc-ldap-search** command indicates whether to retrieve the distinguished name with an LDAP search.

This command is relevant when the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

When enabled, you need to use the following properties:

- How to connect to the LDAP server. Use either approach:
  - The **mc-server-host** and **mc-server-port** commands
  - The **mc-loadbalancer-group** command
- Optionally associate an existing SSL Proxy Profile object to use secure communication with the LDAP server with the **mc-ldap-sslproxy** command
- Specify the login DN to access the LDAP server with the **mc-ldap-bind-dn** command
- Specify the user's password with the **mc-ldap-bind-password** command

- Optionally associate an existing LDAP Search Parameters object with the **mc-ldap-parameters** command

Related Commands

mc-ldap-bind-dn, **mc-ldap-bind-password**, **mc-ldap-parameters**, **mc-ldap-sslproxy**, **mc-loadbalancer-group**, **mc-server-host**, **mc-server-port**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-server-host ldap.mydomain.com
# mc-server-port 389
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
#
```

---

mc-ldap-sslproxy

Assigns the SSL Proxy Profile to the LDAP credentials server.

Syntax

mc-ldap-sslproxy *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **mc-ldap-sslproxy** command assigns an existing SSL Proxy Profile to use secure communication with the LDAP credentials server. When specified, LDAP communication uses the configuration that is defined in the assigned SSL Proxy Profile. If not specified, communications do not use SSL and are nonsecure.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

Beyond assigning an SSL Proxy Profile for secure communication when searching the LDAP for the group name, you need to use the following properties:

- How to connect to the LDAP server. Use either approach:
  - The **mc-server-host** and **mc-server-port** commands
  - The **mc-loadbalancer-group** command
- Specify the login DN to access the LDAP server with the **mc-ldap-bind-dn** command
- Specify the user's password with the **mc-ldap-bind-password** command
- Optionally, the **mc-ldap-parameters** command to associate an existing LDAP Search Parameters object

Related Commands

mc-ldap-bind-dn, **mc-ldap-parameters**, **mc-ldap-bind-password**, **mc-ldap-search**, **mc-loadbalancer-group**, **mc-server-host**, **mc-server-port**

Examples

- Uses the `ldapone` SSL Proxy Profile for secure communications.

```
# ldap-sslproxy ldapone
#
```

---

mc-loadbalancer-group

Assigns a load balancer group to for LDAP credentials searching.

Syntax

mc-loadbalancer-group *name*

Parameters

name Specifics the name of an existing load balancer group.

Guidelines

The **mc-loadbalancer-group** command assigns an LDAP load balancer group instead of a single LDAP server for performing an LDAP search to retrieve the user's group.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

This command is mutually exclusive with the combination of the **mc-server-host** and **mc-server-port** commands.

Related Commands

mc-ldap-search, **mc-method**, **mc-server-host**, **mc-server-port**, **show loadbalancer-status**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-loadbalancer-group LBGroup1
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
#
```

---

mc-method

Specifies the credentials mapping method.

Syntax

mc-method {**custom** | **local** | **xmlfile**}

Parameters

custom
Uses a custom style sheet. Requires an **mc-custom-url** value.

local Uses the user group configuration that is maintained on the local system. Does not access external resources.

xmlfile
Uses a locally stored AAA Info file. Requires an **mc-info-url** value.

Guidelines

The **mc-method** command sets the credential mapping (authorization) method for RBM.

Table 16 lists the supported credential mapping methods for each user authentication method.

Table 16. Authentication methods and supported credential mapping methods

	mc-method		
au-method	**local**	**xmlfile**	**custom**
custom	No	Yes	Yes
ldap	No	Yes	Yes
local	Yes	Yes	Yes
radius	No	Yes	Yes
spnego	No	Yes	Yes
client-ssl	No	Yes	Yes
xmlfile	Yes	Yes	Yes

When the credentials mapping method is **local** or **xmlfile**, you can use the **mc-ldap-search** command to retrieve the distinguished name with an LDAP search.

Notes:

1. The selected credentials mapping method must be fully configured before invoking this command.
2. If the `admin` account is not configured with all permissions, the `admin` user is locked out of the WebGUI. Access the command line to change this circumstance.

Related Commands

au-method, **mc-custom-url**, **mc-info-url**, **mc-ldap-search**

Examples

- Sets the authorization method to `xmlfile` and identifies the location of the file.

```
# mc-method xmlfile
# mc-method "local:///RBMPolicy.xml"
#
```

- Sets the authorization method to local.

```
# mc-method local
#
```

mc-server-host

Specifies the IP address or domain name of a remote credentials server.

Syntax

mc-server-host *host*

Parameters

host Specifies the IP address or domain name of the server.

Guidelines

The **mc-server-host** command specifies the IP address or domain name of the credentials server.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

This command is mutually exclusive with the **mc-loadbalancer-group** command.

Beyond specifying the LDAP server when searching the LDAP for the group name, you need to use the following commands:

- The **mc-server-port** command to specify the listening port on the LDAP server
- Optionally, the **mc-ldap-sslproxy** command to associate an existing SSL Proxy Profile object to use secure communication with the LDAP server
- The **mc-ldap-bind-dn** command to specify the login DN to access the LDAP server
- The **mc-ldap-bind-password** command to specify the user's password
- Optionally, the **mc-ldap-parameters** command to associate an existing LDAP Search Parameters object

Related Commands

mc-ldap-bind-dn, **mc-ldap-parameters**, **mc-ldap-bind-password**, **mc-ldap-search**, **mc-ldap-sslproxy**, **mc-loadbalancer-group**, **mc-method**, **mc-server-port**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-server-host ldap.mydomain.com
# mc-server-port 389
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
#
```

mc-server-port

Specifies the port on the credentials server.

Syntax

mc-server-port *port*

Parameters

port Specifies the port number of the credentials server.

Guidelines

The **mc-server-port** command specifies the listening port on the credentials server.

This command is relevant only in the following situation:

- LDAP search is enabled with the **mc-ldap-search** command
- When the credentials mapping method, as defined with the **mc-method** command, is **local** or **xmlfile**.

This command is mutually exclusive with the **mc-loadbalancer-group** command.

Beyond specifying the password for the listening port on the LDAP server when searching the LDAP for the group name, you need to use the following commands:

- The **mc-server-host** command to specify the LDAP server
- Optionally, the **mc-ldap-sslproxy** command to associate an existing SSL Proxy Profile object to use secure communication with the LDAP server
- The **mc-ldap-bind-password** command to specify the user's password
- Optionally, the **mc-ldap-parameters** command to associate an existing LDAP Search Parameters object

Related Commands

mc-ldap-bind-dn, **mc-ldap-parameters**, **mc-ldap-bind-password**, **mc-ldap-search**, **mc-ldap-sslproxy**, **mc-loadbalancer-group**, **mc-method**, **mc-server-host**

Examples

- Uses a local XML file to map credentials and performs an LDAP search to retrieve the distinguished name.

```
# mc-method xmlfile
# mc-info-url local:///RBM-MC.xml
# mc-ldap-search on
# mc-server-host ldap.mydomain.com
# mc-server-port 389
# mc-ldap-bind-dn "cn=proxyuser"
# mc-ldap-bind-password p@Ssw0rd
#
```

pwd-aging

Specifies whether users must be periodically change their passwords.

Syntax

pwd-aging {**on** | **off**}

Parameters

on Requires the periodic change of passwords.

off (Default) Allows continued use of passwords.

Guidelines

If password aging is enabled, use the **pwd-max-age** command to specify the maximum shelf-life of a user password.

Related Commands

pwd-max-age

Examples

- Requires passwords to be changed every 15 days.

```
# pwd-aging on
# pwd-max-age 15
#
```

pwd-digit

Specifies whether passwords must contain at least one numeric character.

Syntax

pwd-digit {**on** | **off**}

Parameters

on Indicates that passwords must contain at least one numeric characters.

off (Default) Indicates that passwords do not require numeric characters.

Guidelines

When enabled, `p4AssWord` would be acceptable, but `password` or `PASSWORD` would not be acceptable.

When disabled, `p4AssWord`, `password`, or `PASSWORD` would be acceptable.

Related Commands

pwd-minimum-length, **pwd-mixed-case**, **pwd-nonalphanumeric**, **pwd-username**

Examples

- Requires passwords to contain one or more numeric characters.

```
# pwd-digit on
#
```

- Restores the default state.

```
# pwd-digit off
#
```

pwd-history

Specifies whether recent passwords can be reused.

Syntax

pwd-history {**on** | **off**}

Parameters

on Indicates that passwords can be reused.

off (Default) Indicates that passwords cannot be reused.

Guidelines

When enabled, use the **pwd-max-history** command to specify the number of passwords to retain. Passwords that are retained are not eligible for reuse.

Related Commands

pwd-max-history

Examples

- Specifies that the three most recent passwords cannot be reused.

```
# pwd-history on
# pwd-max-history 3
#
```

pwd-max-age

Specifies the maximum duration of passwords.

Syntax

pwd-max-age *duration*

Parameters

duration

Specifies the maximum number of days that a password is valid. Use an integer in the range of 1 through 65535. The default is 300.

Guidelines

If password aging is enabled, use the **pwd-max-age** command to specify the maximum shelf-life of a user password.

Related Commands

pwd-aging

Examples

- Specifies that passwords must be changed every 15 days.

```
# pwd-aging on
# pwd-max-age 15
#
```

pwd-max-history

Specifies the number of passwords to retain.

Syntax

pwd-max-history *count*

Parameters

count Specifies the number of passwords to retain. Use an integer in the range of 1 through 65535. The default is 5.

Guidelines

If password reuse is enabled, use the **pwd-max-history** command to specify the number of recent passwords to retain. Passwords that are retained are not eligible for reuse.

Related Commands

pwd-history

Examples

```
# pwd-history on
# pwd-max-history 3
#
```

specifies that the three most recent passwords cannot be reused.

---

pwd-minimum-length

Specifies the minimum length of passwords.

Syntax

pwd-minimum-length *length*

Parameters

length Specifies the minimum length. Use an integer in the range of 1 through 128. The default is 6.

Related Commands

pwd-digit, **pwd-mixed-case**, **pwd-nonalphanumeric**, **pwd-username**

Examples

- Sets the minimum password length of 10 characters.

```
# pwd-minimum-length 10
#
```

---

pwd-mixed-case

Specifies whether passwords must contain uppercase and lowercase characters.

Syntax

pwd-mixed-case {**on** | **off**}

Parameters

on Indicates that passwords must contain uppercase and lowercase characters.

off (Default) Indicates that passwords do not require uppercase and lowercase characters.

Guidelines

When enabled, `pAssWord` is acceptable, but `password` or `PASSWORD` is not acceptable.

When disabled, `pAssWord`, `password`, or `PASSWORD` is acceptable.

Related Commands

pwd-digit, **pwd-minimum-length**, **pwd-nonalphanumeric**, **pwd-username**

Examples

- Requires passwords to contain both uppercase and lowercase characters.

```
# pwd-mixed-case on
#
```

- Restores the default state.

```
# pwd-mixed-case off
#
```

pwd-nonalphanumeric

Specifies whether passwords must contain nonalphanumeric characters.

Syntax

pwd-nonalphanumeric {**on** | **off**}

Parameters

on Indicates that passwords must contain nonalphanumeric characters.

off (Default) Indicates that passwords do not require nonalphanumeric characters.

Guidelines

When enabled, `pa$$word` is acceptable, but `pAssWord` or `pa33word` is not acceptable.

When disabled, `pa$$word`, `pAssWord`, or `pa33word` is acceptable

Related Commands

pwd-digit, **pwd-minimum-length**, **pwd-mixed-case**, **pwd-username**

Examples

- Requires passwords to contain nonalphanumeric characters.

```
# pwd-nonalphanumeric on
#
```

- Restores the default state.

```
# pwd-nonalphanumeric off
#
```

pwd-username

Specifies whether passwords can contain the user name string.

Syntax

pwd-username {**on** | **off**}

Parameters

on Indicates that passwords can contain the user name.

off (Default) Indicates that passwords cannot contain the user name.

Guidelines

When enabled, the password `BobPassword` or `password4Bob` is acceptable for user name `Bob`.

When disabled, the password `BobPassword` or `password4Bob` is not acceptable for user name `Bob`.

Related Commands

pwd-digit, **pwd-minimum-length**, **pwd-mixed-case**, **pwd-nonalphanumeric**

Examples

- Allows passwords to contain the user name.

```
# pwd-username on
#
```

- Restores the default state.

```
# pwd-username off
#
```

---

restrict-admin

Specifies whether to restrict access by the `admin` account to the command line on the serial port.

Syntax

restrict-admin {**on** | **off**}

Parameters

on Restricts the `admin` account to command line access on the serial port.

off (Default) Allows the `admin` account to all access methods.

Guidelines

The **restrict-admin** command specifies whether to restrict access by the `admin` account to the command line on the serial port.

- When enabled, the access method for the `admin` account is through the command line when connected to the serial port.
- When disabled, the default state, the `admin` account can use all of the available access methods.

Examples

- Restrict command line access by the `admin` account to the serial port.

```
# restrict-admin on
#
```

- Allow access by the `admin` account to all access methods.

```
# restrict-admin off
#
```

Chapter 70. Schema Exception Map configuration mode

This chapter provides an alphabetic listing of commands that are available in Schema Exception Map configuration mode.

To enter this configuration mode, use the Global **schema-exception-map** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Schema Exception Map configuration mode.

original-schema

Identifies the target schema which is subject to the exception rules defined by the current Schema Exception Map.

Syntax

original-schema *URL*

Parameters

URL Identifies the URL of the target schema.

Related Commands

rule

Examples

- Specifies `store:///schema-12b.xsd` as the target schema.

```
# original-schema store:///schema-12b.xsd
#
```

rule

Adds an exception rule to the current Schema Exception Map.

Syntax

rule *expression* {**allowEncrypted** | **requireEncrypted**}

Parameters

expression
Specifies an XPath expression that identifies a schema element or elements subject to this rule.

allowEncrypted
Specifies that elements subject to this rule can be encrypted.

requireEncrypted
Specifies that elements subject to this rule must be encrypted.

Related Commands

original-schema

Examples

- Creates the SEM-1 Schema Exception Map. Specifies store:///schema-12b.xsd as the target schema Adds a rule to the current Schema Exception Map, which requires that all SSN nodes be encrypted.

```
# schema-exception-map SEM-1
Schema Exception Map configuration mode
# original-schema store:///schema-12b.xsd
# rule //SSN requireEncrypted
#
```

Chapter 71. SFTP Server Front Side Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in SFTP Server Front Side Handler configuration mode.

To enter this configuration mode, use the Global **source-ssh-server** command.

Most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SFTP Server Front Side Handler configuration mode.

aaa-policy

Assigns an AAA policy for SSH user authentication.

Syntax

aaa-policy *name*

Parameters

name Specifies the name of an existing AAA Policy object.

Guidelines

The **aaa-policy** selects the AAA Policy to perform SSH user authentication of the information provided during SSH key exchange.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifies the name of an existing Access Control List object.

Guidelines

The **acl** command specifies an Access Control List object that allows or denies access to the SFTP server based on the IP address of the SFTP client.

When attached to a server, the default for an Access Control List is to deny all access. To deny access only to select IP addresses, first grant access to all addresses (`allow 0.0.0.0`). Then, create deny entries for the hosts to be excluded.

address

Specifies the IP address on which the SFTP server listens.

Syntax

address *address*

Parameters

address Specifies the IP address or local host alias on which the SFTP server listens. The default is `0.0.0.0`.

Guidelines

The **address** command specifies the local IP address or local host alias on which the SFTP server service listens. The default of `0.0.0.0` indicates that the service is active on all IP addresses.

The use of local host aliases can help to ease migration tasks among machines.

Related Commands

port

---

allow-backend-listings

Specifies whether backend directory listings are allowed.

Syntax

allow-backend-listings {**on** | **off**}

Parameters

on (Default) Allow backend directory listings.

off Do not allow backend directory listings.

Guidelines

The **allow-backend-listings** command requires Transparent filesystem and an FTP server on the back-end.

---

default-directory

Specifies the working directory on the SFTP server.

Syntax

default-directory *directory*

Parameters

directory
Specifies the initial working directory for all users on this SFTP server. The default is the root directory (/).

Guidelines

The **default-directory** command specifies the current working directory for all users of this SFTP server. This directory will be the initial working directory after users connect and authenticate.

Examples

- Sets the initial working directory on the SFTP server after users connect and authenticate to `/test`.

```
# source-ssh-server sftpServer-1
New SFTP Server Front-Side Handler configuration
# default-directory /test
```

filesystem

Controls the file system type that is presented by the SFTP server.

Syntax

filesystem {**transparent**}

Parameters

transparent
The files and directories shown are those on the back end of the associated service.

Guidelines

The **transparent** file system shows the contents of the equivalent path of the server on the backend of the service with which this SFTP server is associated.

host-private-key

Specifies a private host key used by the SFTP Server during Host Authentication.

Syntax

host-private-key *keyName*

Parameters

keyname
Identifies a Crypto Key object used for SSH keys.

Guidelines

Assignment of a **host-private-key** is optional. If no keys are specified, the default RSA and DSA Host Private Keys are used. Keys are used in the order specified. There is a limit of 256 keys. Only SSH-2 RSA and DSA keys are allowed.

idle-timeout

Specifies how long SFTP control connections can be idle before they time out.

Syntax

idle-timeout *seconds*

Parameters

seconds
Specifies the number of seconds that the SFTP control connection can be idle. The default is 0, which disables the timeout.

Guidelines

The **idle-timeout** command specifies the number of seconds that the SFTP control connection can be idle before the SFTP server closes the control connection.

---

port

Specifies the monitored port.

Syntax

port *port*

Parameters

port Specifies the port that is monitored by the SFTP Server Front Side Handler object. Use an integer in the range of 1 through 65534. The default is 22.

Guidelines

The **port** command specifies the port that is monitored by the SFTP Server Front Side Handler object.

Related Commands

address

---

user-auth

Specifies which SSH user authentication methods are allowed for client authentication.

Syntax

user-auth *type*[+*type*]

Parameters

type Identifies the authentication method. Use the following keywords:

publickey
Specifies that SSH public key user authentication is allowed.

password
Specifies that SSH password user authentication is allowed.

Guidelines

Authentication can be one or more methods. At least one method is required. By default, both **publickey** and **password** are allowed.

Chapter 72. Simple Rate Limiter configuration mode

This chapter provides an alphabetic listing of commands that are available in Simple Rate Limiter configuration mode.

To enter this configuration mode, use the Global **simple-rate-limiter** command. While in this mode, define the Simple Rate Limiter.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Simple Rate Limiter configuration mode.

action

Determines the action to take when the rate of requests exceeds the threshold set.

Syntax

action {**notify** | **reject** | **shape**}

Parameters

notify Generate log message in the appropriate application domain. Log targets must subscribe to this event to capture message.

reject Requests are rejected until transaction rate drops below the configured limit.

shape Delay requests as much as possible to lower the transaction rate to the configured limit. Once too many messages are buffered, creating a low memory state, transactions are rejected until rate drops. The ability to shape transactions is limited when concurrent connections are high.

Related Commands

tps

Examples

- Buffers or delays requests until such time as the rate of transactions drops below the rate threshold.

```
# action shape
```

concurrent-connection-limit

Determines the number of concurrent connections allowed.

Syntax

concurrent-connection-limit *limit*

Parameters

limit Specifies the number of simultaneous connections to allow per user. Set to 0 to disable enforcement.

Related Commands

distinct-sources, **tps**

distinct-sources

Determines the number of distinct sources, or user identities, tracked by the limiter.

Syntax

distinct-sources *count*

Parameters

count Specifies the number of distinct sources tracked by this limiter. The default is 10000.

Related Commands

concurrent-connection-limit

tps

Determines the number of transactions per second to allow per user identity.

Syntax

tps *count*

Parameters

count Specifies the number of transactions to allow. The default is 500.

Related Commands

action

Chapter 73. SLM Action configuration mode

This chapter provides an alphabetic listing of commands that are available in SLM Action configuration mode. SLM is the abbreviation for Service Level Monitor.

To enter this configuration mode, use the Global **slm-action** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SLM Action configuration mode.

log-priority

Specifies the log priority of the message that is generated when the current SLM Action is triggered.

Syntax

log-priority *priority*

no log-priority

Parameters

priority

Specifies the message level. Use one of the following keywords:

- **emergency**
- **alert**
- **critical**
- **error**
- **warning**
- **notice**
- **info**
- **debug** (Default)

Guidelines

Use the **no log-priority** command to restore the default **debug** setting.

Examples

- Creates the `minorPenalty` SLM Action. Sets the priority to informational.

```
# slm-action minorPenalty
SLM Action configuration mode
# log-priority info
```

- Restores the default priority.

```
# no log-priority
#
```

type

Specifies the administrative procedure followed when the current SLM Action is triggered.

Syntax

type *type*

Parameters

type Identifies the administrative procedure. Use one of the following keywords:

log-only
Generates a log message when the current action is triggered and continues to process transactions.

reject Generates a log message and drops traffic when the current action is triggered.

shape Generates a log message and queues traffic when the current action is triggered.

Guidelines

When the SLM Action type is **reject**, the appliance drops all traffic until the monitored entity is within conformance levels.

When the SLM Action type is **shape**, the appliance queues the next 2500 transactions for later transmission when the monitored entity is within conformance levels. After 2500 transactions are queued, further transactions are rejected.

You must explicitly designate an SLM Action type to complete object configuration.

Examples

- Creates the `minorPenalty` SLM Action. Sets the priority to informational. Specifies that nonconforming transactions are queued.

```
# slm-action minorPenalty
SLM Action configuration mode
# log-priority info
# type shape
```

Chapter 74. SLM Credential Class configuration mode

This chapter provides an alphabetic listing of commands that are available in SSL Credential Class configuration mode.

To enter this configuration mode, use the Global **slm-cred** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SSL Credential Class configuration mode.

header

Specifies the name of the header that contains the match.

Syntax

header *string*

Parameters

string Identifies the name of the HTTP header to match.

Guidelines

The **header** command is meaningful only if the match type, as defined by the **type** command is **ip-from-header** or **request-header**.

Related Commands

type

match-type

Identifies the match type.

Syntax

match-type {**exact-match** | **per-extracted-value** | **regexp-match**}

Parameters

exact-match
Specifies that only a subset of the group that is defined by the **type** command is subject an SLM policy. The subset is defined by one or more entries that are specified by the **value** command. The SLM policy is enforced only in the event of an exact match.

per-extracted-value
Specifies that the system extracts and keeps a list of all unique credentials that is defined by the **type** command. All configured policies apply to each of the extracted credentials.

regexp-match
Specifies that only PCRE-style expressions that match the values are subject

to the SLM policy. The subset is defined by one or more entries specified by the **value** command. The policy statement is evaluated only in the event of a match.

Guidelines

A Credential Class defines a user group subject to an SLM policy. It consists of:

- A credential type (defined by the **type** command), which specifies a method used to obtain credentials
- A match type (defined by this command), which specifies if all or selected members of the group identified by **type** are subject to the an Policy
- A credential value (defined by the value command), which is used when the match type is **exact-match** to identify specific members of a Credential Class subject to an SLM policy

The **aaa-mapped-credential** and **aaa-username** types can only be used if the Document Processing that uses Credentials Class (as part of an SLM policy) has previously implemented an AAA policy which provides the needed credentials.

The match type is ignored when the credential type is **custom-stylesheet**, and is otherwise required.

Examples

- Creates the `extranetPartner` SLM Credential Class. Specifies that Credential Class membership is based on source IP address, and that only a subset of IP addresses is subject to an SLM policy.

```
# slm-cred extranetPartner
SLM Credential Class configuration mode
# type client-ip
# match-type exact-match
#
```

---

stylesheet

Specifies the location of the style sheet to establish Credential Class membership.

Syntax

stylesheet *URL*

no stylesheet *URL*

Parameters

URL Specifies the location of the style sheet.

Guidelines

Meaningful only if the Credential Class type is **custom-stylesheet**.

Use the **no stylesheet** command to remove a custom style sheet from the Credentials Class.

Examples

- Creates the `extranetPartner` SLM Credential Class. Specifies that Credential Class membership is determined by the style sheet at the specified location.

```
# slm-cred extranetPartner
SLM Credential Class configuration mode
# type custom-stylesheet
# stylesheet local:///extranetPartner.xsl
#
```

- Removes the specified style sheet from the Credentials Class.

```
# no stylesheet local:///extranetPartner.xsl
#
```

type

Specifies the group of credentials subject to the SLM policy.

Syntax

type *type*

Parameters

type Identifies the group of credentials. Use one of the following keywords:

aaa-mapped-credential
(Default) Specifies that the credentials returned by an AAA policy mapping credentials operation are members of this Credentials Class

aaa-username
Specifies that the user identities extracted by an AAA policy are members of this Credentials Class

client-ip
Specifies that members of this Credentials Class are defined by source IP addresses

custom-stylesheet
Specifies that members of this Credentials Class are defined by an style sheet

ip-from-header
Specifies that members of this Credentials Class are defined the name of the HTTP header that contains the client IP address (for example, `X-Client-IP`). When specified, use the **header** command to define the name of the header.

mq-application
Specifies that members of this Credentials Class are defined by MQ application names

request-header
Specifies that members of this Credentials Class are defined the name of the HTTP header that contains the credential to use. When specified, use the **header** command to define the name of the header.

Guidelines

A Credential Class defines a user group subject to an SLM policy. It consists of:

- A credential type (defined by this command), which specifies a method used to obtain credentials

- A match type (defined by the **match-type** command), which specifies if all or selected members of the group identified by this command are subject to the policy
- A credential value (defined by the **value** command), which is used when the match type is **exact-match** to identify specific members of a Credential Class subject to an SLM policy

The **aaa-mapped-credential** and **aaa-username** types can only be used if the processing rule that uses this Credentials Class (as part of an SLM policy) previously implemented an AAA policy to provide the needed credentials.

Related Commands

header, **match-type**, **value**

Examples

- Creates the `extranetPartner` SLM Credential Class. Specifies that Credential Class membership is based on source IP address.

```
# slm-cred extranetPartner
SLM Credential Class configuration mode
# type client-ip
#
```

value

Specifies which members are subject to the SLM policy.

Syntax

value *string*

no value *string*

Parameters

string Identifies the exact match criteria.

- If the type is **aaa-mapped-credential**, specify a string in the format returned by the mapping operation.
- If the type is **aaa-username**, specify a user name in the format extracted by the AAA Policy.
- If the type is **client-ip**, specify an IP address and network-prefix mask.
- If the type is **mq-application**, specify an MQ application name.

Guidelines

Meaningful only if the match type is **exact-match**.

A Credential Class defines a user group subject to an SLM policy. It consists of:

- A credential type (defined by the **type** command), which specifies a method used to obtain credentials
- A match type (defined by the **match-type** command), which specifies if all or selected members of the group identified by **type** are subject to the an Policy
- A credential value (defined by this command), which is used when the match type is **exact-match** to identify specific members of a Credential Class subject to an SLM policy

Use the **value** command one or many times to define any number of exact matches.

The **value** command is ignored when the Credential Class type is **custom-stylesheet**.

Use the **no value** command to remove an exact match value.

Examples

- Creates the `extranetPartner` SLM Credential Class. Specifies that Credential Class membership is based on source IP address, and that only the defined subset of IP addresses is subject to an SLM policy.

```
# slm-cred extranetPartner
SLM Credential Class configuration mode
# type client-ip
# match-type exact-match
# value 10.119.10.3/32
# value 192.168.3.012/27
# value 192.168.12.0/24
#
```

- Removes an exact match value from the Credential Class.

```
# no value 192.168.12.0/24
#
```

Chapter 75. SLM Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in SLM Policy configuration mode. SLM is the abbreviation for Service Level Monitor.

To enter this configuration mode, use the Global **slm-policy** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SLM Policy configuration mode.

eval-method

Specifies the behavior when an action is triggered.

Syntax

eval-method *behavior*

no eval-method

Parameters

behavior
Specifies the SLM Policy. Use one of the following keywords:

execute-all-statements
Indicates that policy execution continues until all associated statements are evaluated.

terminate-at-first-action
Indicates that policy execution ceases upon the triggering of any action by an associated statement.

terminate-at-first-reject
Indicates that policy execution ceases upon the triggering of any action that drops traffic by an associated statement.

Guidelines

Use the **no eval-method** command to delete the current behavior.

Examples

- Creates the `SLM-Policy-1` SLM Policy. Specifies that Policy processing ceases when a called action drops traffic.

```
# action SLM-Policy-1
SLM Policy configuration mode
# eval-method terminate-at-first-reject
#
```

- Deletes the current behavior and assigns the `execute-all-statements` behavior.

```
# action SLM-Policy-1
SLM Policy configuration mode
# no eval-method
# eval-method execute-all-statements
```

peer-group

Associates a peer group.

Syntax

peer-group *name*

Parameters

name Specifies the name of an existing Peer Group object.

Guidelines

The **peer-group** command assigns a Peer Group object to the SLM policy. This peer group enables the aggregation and sharing of SLM date across similarly configured DataPower appliances.

Related Commands

peer-group (Global)

Examples

- Associates the `SLM-Group-1` SLM Peer Group with the current SLM Policy.

```
# peer-group SLM-Group-1
#
```

statement

Constructs an SLM Statement and adds the statement.

Syntax

statement *index credential-class resource-class schedule action interval-length interval-type algorithm threshold-type threshold-level high-low burst reporting-interval records*

no statement *index*

Parameters

index Specifies the order in which the statement is executed. Statements are executed from least to greatest.

Note: Adding a statement with the same index value as an existing statement deletes the existing statement and replaces it with the newer statement.

credential-class
Specifies the name of the SLM Credentials Class.

resource-class
Specifies the name of the SLM Resource Class.

schedule
Specifies the name of the SLM Schedule.

action Specifies the name of the SLM Action.

interval-length

Specifies the length of the measurement interval in seconds. The default is 0, which allows all messages and never triggers the threshold to enforce the SLM Action.

interval-type

Specifies the threshold type and takes one of the following values:

fixed Indicates a fixed interval. A fixed interval is a discrete block of time. For example, from 8:00 A.M. to 9:00 A.M. Fixed intervals are calculated from the start of the assigned SLM Schedule. Without an SLM Schedule, fixed intervals are calculated from 12:00 A.M.

moving

Indicates a moving interval in seconds. A moving interval is a sliding-window; for example, the last 60 minutes.

algorithm

Specifies the threshold algorithm. Use one of the following keywords:

greater-than

Specifies a simple numeric algorithm that triggers the SLM Action when the threshold level is greater than the defined value.

high-low-thresholds

Specifies an algorithm that triggers the SLM Action at the *high* threshold and continues to trigger until the *low* threshold is reached.

less-than

Specifies a simple numeric algorithm that triggers the SLM action when the threshold level is less than the defined value.

token-bucket

Specifies a rate-based algorithm that allows bursting. The algorithm consists of a bucket with a maximum capacity of "N" tokens that refills at a rate of "R" tokens per second. Each token typically represent a quantity of whatever resource is being rate-limited.

threshold-type

Specifies the threshold measurement target. Use one of the following keywords:

count-all

Specifies that the threshold measures all requests for the resource set identified by *resource-class*.

count-errors

Specifies that the threshold measures only errors.

latency-backend

Specifies that the threshold measures appliance-to-server latency.

latency-internal

Specifies that the threshold measures internal latency (processing time).

latency-total

Specifies that the threshold measures both backend and internal latency.

threshold-level
Specifies the threshold that triggers the SLM Action. If the algorithm is **high-low-thresholds**, specifies the *high* threshold. The units of measure depends on the threshold type.

- If the threshold is a count, specify an integer for the aggregate count.
- If the threshold is latency, specify an integer for the latency in seconds.

The default is 0, which means that no message is accepted.

high-low
If the algorithm is **high-low-thresholds**, specifies the *low* threshold. For all other algorithms, this argument is meaningless.

burst If the algorithm is **token-bucket**, specifies the size of the committed burst. The committed burst defines how much traffic can be send during a reporting interval. The burst size should be at least twice the value of the threshold level. The default is 0, which means the burst limit is the configured threshold level.

reporting-interval
Specifies the base aggregation level in minutes for reporting of statistics. This value does not affect the threshold intervals.

records Specifies the maximum records to save per reporting interval.

Guidelines

Use the **no statement** command to delete an SLM Statement from the current SLM Policy.

Related Commands

slm-action (Global), **slm-cred** (Global), **slm-policy** (Global), **slm-rsrc** (Global), **slm-sched** (Global)

Examples

- Creates a statement with the index value of 1.

```
# action SLM-Policy-1
SLM Policy configuration mode
# statement 1 slmCred-1 slmRsrc-1 skedPeak
 minorPenalty 600 moving greater-than count-errors 3 0 0 5000
#
```

- Deletes the statement with the index value of 1.

```
# action SLM-Policy-1
SLM Policy configuration mode
# no statement 1
#
```

Chapter 76. SLM Resource Class configuration mode

This chapter provides an alphabetic listing of commands that are available in SLM Action configuration mode. SLM is an abbreviation for Service Level Monitor.

To enter this configuration mode, use the Global **slm-rsrc** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SLM Action configuration mode.

match-type

Specifies match criteria.

Syntax

match-type {**exact-match** | **per-extracted-value** | **regexp-match**}

Parameters

exact-match
(Default) Specifies that only a subset of the members of the group that is defined by the **type** command is covered by this SLM Resource Class. Only resources with an exact match to the resource values are subject to the SLM policy. Resources that do not match the values that are provided by the **value** command are not subject to the SLM policy.

per-extracted-value
Specifies that the system extracts and keeps a list of all unique resources that is defined by the **type** command. All configured policies apply to each of the extracted resources.

regexp-match
Specifies that only a subset of the members of the group that is defined by the **type** command is covered by this SLM Resource Class. Only resources with a PCRE-style expression match to the resource values are subject to the SLM policy. Resources that do not match the values that are provided by the **value** command are not subject to the SLM policy.

Guidelines

match-type is used if the Resource Method (defined by the **type** command) is **aaa-mapped-resource**, **destination-URL**, **error-code**, **front-URL**, **reply-mq-qname**, **request-mq-qname**, **wsdl**, **wsdl-operation**, **wsdl-port**, or **wsdl-service** to specify match criteria.

match-type is not relevant if the Resource Method is **concurrent-connections**, **custom-stylesheet**, **request-message**, **response-message**, **soap-fault**, or **xpath-filter** since these types test only for existence.

Related Commands

type, **value**

Examples

- Creates the `profitLossStatements` resource class. Specifies that membership in the resource class is defined by the destination URL method. Coverage by the resource class is restricted to a specific subset of destination URLs that contain `www.datapower.com`.

```
# slm-rsrc profitLossStatements
SLM Resource configuration mode
# type destination-url
# exact-match
# value *www.datapower.com*
#
```

---

stylesheet

Specifies the style sheet to produce resource identification.

Syntax

stylesheet *URL*

Parameters

URL Identifies the location of the style sheet.

Guidelines

Used only if the Resource Method (defined by the **type** command) is **custom-stylesheet** to specify the style sheet to produce resource identification.

Related Commands

type

Examples

- Identifies the local `extractResourceID.xsl` style sheet as the source of resource identification data.

```
# type custom-stylesheet
# stylesheet local:///extractResourceID.xsl
#
```

- Removes the local `extractResourceID.xsl` style sheet as the source of resource identification data.

```
# no stylesheet local:///extractResourceID.xsl
#
```

---

subscription

Specifies the UDDI subscription key.

Syntax

subscription *key*

no subscription *key*

key Specifies the subscription key.

Guidelines

Specifies the subscription key. Applicable only when the Resource Method (as defined by the **type** command) is **uddi-subscription**.

Use the **no subscription** command to delete a UDDI-based credential-source.

Examples

- Specifies the `uddi:8b071240-428d-11db-a30b-47fc0b00a30a` subscription key.

```
# type uddi-subscription
# subscription uddi:8b071240-428d-11db-a30b-47fc0b00a30a
#
```

---

type

Specifies the method to obtain the resource value.

Syntax

type *method*

Parameters

method Identifies the method to identify the requested resource. Use one of the following keywords:

aaa-mapped-resource
(Default) Defines membership by the identities that are returned from an AAA map resource operation.

This method can be used only when the processing policy implements an AAA Policy that provides the required resource mapping.

concurrent-connections
Defines membership by concurrent TCP connections. The TCP connection is from the requesting client to the DataPower appliance. Generally, a client opens only one connection to the DataPower appliance at a time.

Concurrent connections are not specific to user credentials.

concurrent-transactions
Defines membership by concurrent transactions.

custom-stylesheet
Defines membership by a style sheet.

destination-URL
Defines membership by the destination URL. The destination URL is the URL output to the destination server. The destination URL might be identical to the URL that was requested by the client.

error-code
Defines membership by error code values.

front-URL
Defines membership by a client-requested URL or by a rewritten client-requested URL.

reply-mq-qname
Defines membership by the MQ reply queue.

request-message
Restricts membership to all client requests.

request-mq-qname
Defines membership by the MQ request queue.

response-message
Restricts membership to all server requests.

soap-fault
Restricts membership to SOAP fault messages.

uddi-subscription
Defines membership by a UDDI Subscription key.

wsdl Defines membership by a WSDL file.

wsdl-operation
Defines membership by the name of a WSDL operation.

wsdl-port
Defines membership by the name of a WSDL port.

wsdl-service
Defines membership by the name of a WSDL service.

wsrr-subscription
Defines membership by a WSRR subscription.

xpath-filter
Defines membership by an XPath expression.

Related Commands

match-type, **value**

---

value

Specifies which members of an SLM Resource Class are subject to coverage by the current resource class.

Syntax

value *string*

no value

Parameters

string Identifies a specific resource that is subject to coverage by the current SLM Resource Class.

Guidelines

The **value** command is used when the match type as defined by the **match-type** command is **exact-match** or **regexp-match**. The **value** command specifies which members of an Resource Class are subject to coverage by the SLM policy.

The **value** command is not relevant if the match type is **per-extracted-value**.

Use the **no value** command to delete a class member.

Related Commands

match-type

Examples

- Creates the `profitLossStatements` resource class. Specifies that membership in the resource class is defined by the destination URL method. Coverage by the resource class is restricted to a specific subset of destination URLs that contain `www.datapower.com`.

```
# slm-rsrc profitLossStatements
SLM Resource configuration mode
# type destination-url
# exact-match
# value *www.datapower.com*
#
```

---

wsrr-subscription

Identities a WSRR subscription as the resource to define membership.

Syntax

wsrr-subscription *name*

no wsrr-subscription *name*

Parameters

name specifies the WSRR subscription object name.

Guidelines

Specifies the WSRR subscription object name. Used only if the Resource Method (as defined by the **type** command) is **wsrr-subscription**.

Use the **no wsrr-subscription** command to remove a WSRR-based credential-source.

Related Commands

type

Examples

- Specifies the `update-WS-Proxy-1` WSRR subscription that provides class membership data.

```
# type wsrr-subscription
# wsrr-subscription update-WS-Proxy-1
#
```

---

xpath-filter

Specifies the XPath expression used to produce resource identification.

Syntax

xpath-filter *expression*

no xpath-filter *expression*

Parameters

expression
Specifies the operative XPath Expression.

Guidelines

Specifies the XPath expression to produce resource identification. Used only if the Resource Method (as defined by the **type** command) is **xpath-filter**.

Use the **no xpath-filter** command to delete an XPath-based credential-source.

Related Commands

type

Examples

- Identifies the XPath Expression `@destination` as the resource identification tool.

```
# type xpath-filter
# xpath-filter @destination
#
```

- Removes the XPath Expression `@destination` as the resource identification tool.

```
# no xpath-filter @destination
#
```

Chapter 77. SLM Schedule configuration mode

This chapter provides an alphabetic listing of commands that are available in SLM Schedule configuration mode. SLM is an abbreviation for Service Level Monitor.

To enter this configuration mode, use the Global **slm-sched** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SLM Schedule configuration mode.

days

Specifies the days of the week that the schedule is operational.

Syntax

days *day*

Parameters

day Species the days of the week. Use one of the following keywords:
- **Sunday**
- **Monday**
- **Tuesday**
- **Wednesday**
- **Thursday**
- **Friday**
- **Saturday**

Guidelines

Use this command as often as necessary to specify all days during which the schedule is operational.

Examples

- Creates the `weekEnds` SLM Schedule. Specifies `Saturday` and `Sunday` operation.

```
# sched weekEnds
SLM Schedule configuration mode
# days Saturday
# days Sunday
#
```

duration

Specifies the number of minutes per day that the current SLM Schedule is operational.

Syntax

duration *minutes*

Parameters

minutes

Specifies the number of minutes that the current SLM Schedule is operational. Use an integer in the range of 0 through 1439. The default is 1439.

Guidelines

Use the command in conjunction with **start** to define specific time blocks during which this SLM Schedule is operational.

Related Commands

start

Examples

- Creates the `weekEnds` SLM Schedule, which is active on Saturdays and Sundays from midnight to 6:00 PM.

```
# sched weekEnds
SLM Schedule configuration mode
# days Saturday
# days Sunday
# duration 1080
# start 24:00:00
#
```

start

Specifies the time-of-day for the SLM Schedule to become operational.

Syntax

start *time*

Parameters

time Specifies the time-of-day (in military *hh*:*mm*: *ss* format) at which the schedule becomes operational.

Guidelines

Use the command in conjunction with **duration** to define specific time blocks during which this SLM Schedule is operational.

Related Commands

duration

Examples

- Creates the `weekEnds` SLM Schedule, which is active on Saturdays and Sundays from midnight to 6:00 PM.

```
# sched weekEnds
SLM Schedule configuration mode
# days Saturday
# days Sunday
# duration 1080
# start-time 24:00:00
#
```

Chapter 78. SNMP Settings configuration mode

This chapter provides an alphabetic listing of commands that are available in SNMP Settings configuration mode.

To enter this configuration mode, use the Global **snmp** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in SNMP Settings configuration mode.

access

Identifies an SNMP manager granted access to the local SNMP agent or engine, and specifies the access privileges.

Syntax

access *community* **read-only** [*address*]

access *community* **read-write** [*address*]

no access *community-string* [*address*]

Parameters

community
is an SNMP community name (essentially a password) that is included within the SNMP message header. Any SNMP message that contains this community name is considered valid by the SNMP agent or engine.

read-only
managers are restricted to SNMP **get** operations meaning that such managers can read, but cannot change system values contained in the Management Information Base (MIB).

read-write
managers have access to both SNMP **get** and **set** operations, meaning that these managers can both read and change MIB values.

address is an optional IP address of an SNMP manager the belongs to *community*. This SNMP specific manager is granted access to the local agent or engine.

In the absence of an IP address, access is granted to all SNMP managers using *community*.

Guidelines

Many environments support only two SNMP communities:
- A public (read-only) community
- A private (read-write) community

There is no limit to the number of communities that can be supported. Nor is there any limit to the number of SNMP managers contained within a specific community.

Use the **no access** command to delete a previously configured SNMP manager.

Examples

- Creates a read-only community. Any SNMP manager, using the `public` community is granted read-only access to the local agent.

```
# access public read-only
#
```

- Specifies two SNMP managers granted access to the local agent. Both managers are granted read-write access using the `private` community.

```
# access private read-write 10.10.10.23
# access private read-write 192.168.1.100
```

- Denies access to all SNMP managers using the public community name.

```
# no access public
#
```

- Denies access to the SNMP manager at 10.10.10.23 and belonging to the `private` community.

```
# no access private 10.10.10.23
#
```

port

Identifies the appliance UDP port monitored by the SNMP agent or engine for SNMP requests.

Syntax

port [*address*] *port*

Parameters

address Specifies an optional IP address that identifies a specific local interface as a recipient of SNMP requests.

port Identifies the UDP port monitored by the SNMP agent or engine for SNMP requests. Use an integer in the range of 0 to 65535. The default is 161.

Guidelines

In the absence of an IP address argument, the SNMP agent or engine monitors the specified port on all interfaces for SNMP requests. With a IP address argument provided (for example the address of the management port), the SNMP agent or engine monitors only the specified interface-port pair for SNMP requests.

Examples

- Identifies a nonstandard SNMP port, 65161, on all interfaces as the recipient of SNMP requests.

```
# port 65161
#
```

- Identifies a specific IP address and port pair as the recipient of SNMP requests.

```
# port 10.10.10.10 161
#
```

trap-code

Adds an event code to the trap list.

Syntax

trap-code *code*

no trap-code *code*

Parameters

code Specifies the hexadecimal identifier of an event code.

Guidelines

The **trap-code** command specifies individual event codes to add to the trap list. Invoke this command for each event to add to the list.

Use the **no trap-code** command to delete a previously configured code from the trap list.

Examples

- Adds the "Out of memory" parser event (hexadecimal code `0x00030002`) to the list.

```
# trap-code 0x00030002
#
```

trap-priority

Specifies the minimum priority for trap events.

Syntax

trap-priority *priority*

Parameters

priority
Identifies the event priority. The default is `error`.

Guidelines

The **trap-priority** command specifies the minimum priority for trap events. The priorities are hierarchical (in descending order of criticality) as `emergency`, `alert`, `critic`, `error`, `warn`, `notice`, `info`, and `debug`.

Examples

- Sets trap priority to `warn` or greater criticality.

```
# trap-priority warn
#
```

trap-target

Specifies the recipient of SNMP traps issued by the local SNMP agent or engine.

Syntax

trap-target *address port community*

no trap-target *address port*

Parameters

address Specifies the IP address that receives traps.

port Optionally identifies a UDP port at the IP address. Use an integer in the range of 0 to 65535. The default is 162.

community
Optionally provides a community name (essentially a password) that is included within the SNMP message header. The default is `public`.

Guidelines

The local SNMP agent or engine issues the following generic traps:

- `coldStart`
- `linkDown`
- `linkUp`
- `authenticationFailure`

Use the **no trap-target** command to delete a previously configured recipient of SNMP traps.

Examples

- Specifies a recipient of SNMP traps at `10.10.10.11:162`. The trap recipient is accessed with the `public` community.

```
# trap-target 10.10.10.11
#
```

- Specifies a recipient of SNMP traps at `10.10.100.19:162`. The trap recipient is accessed with the `OpenView` community.

```
# trap-target 10.10.100.19 OpenView
#
```

- Deletes `10.10.10.11:162` from the list of trap recipients.

```
# no trap-target 10.10.10.11
#
```

version

Specifies the supported SNMP version.

Syntax

version {**1** | **2c** | **3**}

Parameters

1 Specifies support for SNMP Version 1 as defined in RFC 1155, RFC 1156, and RFC 1157.

2c (Default) Specifies support for SNMP Version 2c as originally defined in RFC 1901 through RFC 1908

3 Specifies support for SNMP Version 3 as defined in RFC 2261 through RFC 2265.

Examples

- Specifies support for SNMP Version 1.

```
# version 1
#
```

- Specifies support for SNMP Version 2c, the default state.

```
# version 2c
#
```

Chapter 79. SOAP Header Disposition Table configuration mode

This chapter provides an alphabetic listing of commands that are available in SOAP Header Disposition Table configuration mode.

To enter this configuration mode, use the Global **soap-disposition** command. A SOAP Header Disposition Table object contains a list of instructions that controls how to handle SOAP headers, child elements, or both SOAP headers and child elements. This object is used by an `xform` action that uses the `store:///soap-refine.xsl` style sheet.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

refine

Adds an instruction to process a SOAP header or child element.

Syntax

refine *namespace header element action*

no refine

Parameters

namespace
Specifies the namespace URI to match a SOAP header element. The default is a blank string (""), which indicates no restriction.

header Specifies the string to match a SOAP header element local name. The default is a blank string (""), which indicates no restriction.

element
Specifies the string of the child element local name of a SOAP header. The default is a blank string (""), which indicates no restriction.

action Indicates the refinement action to take. The following values are available:

processed
Take the default SOAP action, because the specified element was processed.

unprocessed
Take the default SOAP action, because the specified element was not processed.

keep Keep this SOAP header or child element.

remove
Remove this SOAP header or child element.

fault Generate a SOAP fault if the element exists.

Guidelines

The **refine** command defines an item of SOAP header processing instruction to include in the list of items returned by the SOAP Header Disposition Table object. Issue this command as many times as needed to include all desired items.

Use the **no refine** command to delete the entire list of items tat are configured for the object.

Examples

- Adds three SOAP header processing instructions. The first removes the `Foo` header if the namespace URI is `my-namespace`. The second keeps the `abc` child element local name of a SOAP header named `Bar` regardless of the namespace URI. The last instruction uses the default processed header rules for all the headers in `some-namespace` namespace.

```
# soap soap1
New SOAP Header Disposition Table configuration
# refine "my-namespace" "Foo" "" "remove"
# refine "" "Bar" "abc" "keep"
# refine "some-namespace" "" "" "processed"
# exit
```

- Deletes all SOAP header processing instructions.

```
# soap soap1
Modify SOAP Header Disposition Table configuration
# no refine
# exit
```

Chapter 80. SQL Data Source configuration mode

This chapter provides an alphabetic listing of commands that are available in SQL Data Source configuration mode.

To enter this configuration mode, use the Global **sql-source** command. The **sql-source** command creates the specified data source if it does not exist. While in this configuration mode, define the parameters that define a new SQL Data Source object or modify an existing one.

While in this configuration mode, all of the commands that are listed in "Common commands" on page 2 and most of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are available.

db

Identifies the type of database to access.

Syntax

db {**DB2** | **DB2v9** | **MSSQLServer** | **Oracle** | **Sybase**}

Parameters

DB2 (Default) IBM DB2® version 8 or earlier

DB2v9
IBM DB2 version 9

MSSQLServer
Microsoft SQL Server

Oracle Oracle database products

Sybase
Sybase database products

host

Determines the remote location of the data source.

Syntax

host *host*

Parameters

host Specifies the IP address or host name of the machine that host the database instance.

Related Commands

port

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Sets the host to IP address `192.168.1.109` on port 5551.

```
# sql-source db2source
# type DB2
# host 192.168.1.109
# port 5551
```

id

Sets the identifier of the data source.

Syntax

id *identifier*

Parameters

identifier
Specifies the identifier (for example, an Oracle SID) of the data source.

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` named. Sets the database identifier to `Orders`. Authentication and commands will be executed against this database.

```
# sql-source db2source
# id Orders
```

limit

Indicates whether to limit the data from a **SELECT** statement.

Syntax

limit {**on** | **off**}

Parameters

on Limits the data.

off (Default) Does not limit the data.

Guidelines

The **limit** indicates whether to limit the amount of data that an SQL **SELECT** statement returns. When limiting data, use the **limit-size** command to set the limit.

Related Commands

limit-size

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Turns on data limiting. Limits the size of returned data to 256 KB.

```
# sql-source db2source
# limit on
# limit-size 262144
```

limit-size

Sets the limit on data from a **SELECT** statement.

Syntax

limit-size *bytes*

Parameters

bytes Specifies the maximum size in bytes.

Guidelines

The **limit-size** command sets the maximum number of bytes that a **SELECT** command can return. This command is relevant only after using the **limit** command to enable limiting.

Related Commands

limit

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Turns on data limiting. Limits the size of returned data to 256 KB.

```
# sql-source db2source
# limit on
# limit-size 262144
```

maximum-connections

Sets the maximum number of concurrent connections.

Syntax

maximum-connections *connections*

Parameters

connections
Specifies the maximum number of concurrent connections. The default is 10.

Guidelines

The **maximum-connections** command specifies the maximum number of concurrent connections.

Related Commands

db

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Sets the maximum number of concurrent connections to the DB2 data store to 20.

```
# sql-source db2source
# db DB2v9
# maximum-connections 20
```

password

Sets the password to establish a connection to the database.

Syntax

password *password*

Parameters

password
Specifies the string to pass to the database instance as the password.

Guidelines

The **password** command sets the password to establish a connection to the database identified by the **id** command.

Related Commands

id

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Sets the username to `frederica` with a password of `francisco`.

```
# sql-source db2source
# username frederica
# password franciso
```

port

Determines the TCP port to monitor for requests.

Syntax

port *port*

Parameters

port Specifies the TCP port number.

Related Commands

host

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Sets the host to IP address `192.168.1.109` on port 5551.

```
# sql-source db2source
# type DB2
# host 192.168.1.109
# port 5551
```

read-only

Determines whether or not the data source will only accept SQL **SELECT** statements, to read rather than change, delete or add data.

Syntax

read-only {**on** | **off**}

Parameters

on Accepts only **SELECT** statements.

off (Default) Accepts all statements.

Related Commands

limit

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Restricts accepted commands to **SELECT** only.

```
# sql-source db2source
# read-only on
```

sql-config-param

Defines configuration parameters for the connection.

Syntax

sql-config-param *name value*

Parameters

name Specifies the name of the configuration parameter.

value Specifies the value for the configuration parameter.

Guidelines

Define optional but valid ODBC (or CLI) configuration parameters for your data server connection. Configuration parameters modify the behavior of the services that run with a data server. Some configuration parameters in the configuration file are informational and define characteristics about the environment. These configuration parameters cannot be modified.

username

Sets the user to establish the connection.

Syntax

username *name*

Parameters

name Specifies the string to pass to the database instance as the user name.

Related Commands

password

Examples

- Enters SQL Data Source configuration mode and creates the `db2source` object. Sets the username to `frederica` with a password of `francisco`.

```
# sql-source db2source
# username frederica
# password franciso
```

Chapter 81. Stateful Raw XML Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in Stateful Raw XML Handler configuration mode.

To enter this configuration mode, use the Global **source-stateful-tcp** command. While in this mode, define the client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

no acl *name*

Parameters

name Identifies the ACL to be assigned to the Stateful Raw XML Handler.

Guidelines

Only those IP addresses explicitly granted access by the assigned ACL are able to access the Stateful Raw XML Handler.

Use the **no acl** command to remove an ACL from a Stateful Raw XML Handler.

Examples

- Enters Stateful Raw XML Handler configuration mode to create the `avecTCP` Stateful Raw XML Handler. Assigns the designated ACL to the handler.

```
# source-stateful-tcp avecTCP
New Stateful Raw XML Handler configuration
# acl aclRestrictive-1
#
```

- Removes the `aclRestrictive` ACL from the current Stateful Raw XML Handler.

```
# no acl aclRestrictive-1
#
```

close-on-fault

Controls session behavior in the event of a fault condition.

Syntax

close-on-fault {**on** | **off**}

Parameters

on Abandons the session in the event of a fault condition.

off (Default) Maintains the session in the event of a fault.

Examples

- Causes the DataPower appliance to close front and back TCP connections if the appliance generates a fault.

```
# close-on-fault on
#
```

or

```
# no close-on-fault
#
```

- Restores the default state.

```
# close-on-fault off
#
```

or

```
# close-on-fault
#
```

---

local-address

Specifies the local interface to monitor for client requests.

Syntax

local-address {*address* | **0**}

Parameters

address Binds the Stateful Raw XML Handler to a single, specific interface-port pair.

0 Binds the Stateful Raw XML Handler to the specified port on all enabled interfaces.

Guidelines

This command only sets the IP address for the Stateful Raw XML Handler. Use the `port` command to set the TCP port on which the Stateful Raw XML Handler listens.

Related Commands

port, **remote-address**, **remote-port**

Examples

- Binds the current Stateful Raw XML Handler to the specified IP address.

```
# local-address 192.168.1.10
#
```

- Binds the current Stateful Raw XML Handler to all enabled interfaces. This configuration is strongly discouraged for production environments.

```
# local-address 0
#
```

port

Specifies the TCP port to monitor for client requests.

Syntax

port *port*

Parameters

port Binds the Stateful Raw XML Handler to a specific port.

Guidelines

This command only sets the TCP port for the Stateful Raw XML Handler. This port applies to all configured local addresses. Use the **local-address** command to set the IP address on which the handler listens.

Related Commands

local-address, **remote-address**, **remote-port**

Examples

- Binds the current Stateful Raw XML Handler to the specified port number.

```
# port 16000
#
```

remote-address

Specifies the remote server address.

Syntax

remote-address *address*

Parameters

address Binds the Stateful Raw XML Handler to a single, specific interface-port pair.

Guidelines

This command only sets the IP address for the remote server. Use the `remote-port` command to set the remote TCP port.

Related Commands

remote-port

Examples

- Binds the current Stateful Raw XML Handler to the specified remote IP address.

```
# remote-address 192.168.1.11
#
```

remote-port

Specifies the remote server TCP port.

Syntax

remote-port *port*

Parameters

port Binds the Stateful Raw XML Handler to a specific port.

Guidelines

This command only sets the remote TCP port for the Stateful Raw XML Handler. Use the `remote-address` command to set the remote IP address.

Related Commands

local-address, **port**, **remote-address**

Examples

- Binds the current Stateful Raw XML Handler to the specified remote port.

```
# remote-port 16001
#
```

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

no ssl

Parameters

name Specifies the name of the existing SSL Proxy Profile assigned to the Stateful Raw XML Handler.

Guidelines

The SSL Proxy Profile identified here must already exist in the current application domain. Use the Global configuration mode command **sslproxy** to create a new SSL Proxy Profile.

Use the **no ssl** command to remove the SSL Proxy Profile assignment.

Examples

- Assigns the `SSL-1` SSL Proxy to the current Stateful Raw XML Handler.

```
# ssl SSL-1
#
```

- Removes the assignment of the `SSL-1` SSL Proxy named from the current Stateful Raw XML Handler.

```
# no ssl
#
```

Chapter 82. Stateless Raw XML Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in Stateless Raw XML Handler configuration mode.

To enter this configuration mode, use the Global **source-raw** command. While in this mode, define the client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

no acl *name*

Parameters

name Identifies the ACL to be assigned to the Stateless Raw XML Handler.

Guidelines

Only those IP addresses explicitly granted access by the assigned ACL are able to access the Stateless Raw XML Handler.

Use the **no acl** command to remove an ACL from a Stateless Raw XML Handler.

Examples

- Enters Stateless Raw XML Handler configuration mode to create the `sansTCP` Stateless Raw XML Handler. Assigns the `aclRestrictive-1` ACL to the handler.

```
# source-raw sansTCP
Stateless Raw XML Handler configuration mode
# acl aclRestrictive-1
#
```

- Removes the `aclRestrictive-1` ACL from the current Stateless Raw XML Handler.

```
# no acl aclRestrictive-1
#
```

local-address

Specifies the local interface to monitor for client requests.

Syntax

local-address {*address* | **0**}

Parameters

address Binds the Stateless Raw XML Handler to a single, specific interface-port pair.

0 Binds the Stateless Raw XML Handler to the specified port on all enabled interfaces.

Guidelines

This command only sets the IP address for the Stateless Raw XML Handler. Use the **port** command to set the TCP port on which the Stateless Raw XML Handler listens.

Related Commands

port, **remote-address**, **remote-port**

Examples

- Binds the current Stateless Raw XML Handler to the specified IP address.

```
# local-address 192.168.1.10
#
```

- Binds the current Stateless Raw XML Handler to all enabled interfaces. This configuration is strongly discouraged for production environments.

```
# local-address 0
#
```

---

persistent-connections

Indicates whether to establish persistent connections

Syntax

persistent-connections {**on** | **off**}

Parameters

on (Default) Enables the establishment of persistent connections.

off Disables the establishment of persistent connections.

Guidelines

With persistent connections enabled, the default state for both HTTP 1.0 and HTTP 1.1, the Stateless Raw XML Handler negotiates with the remote HTTP peer and establishes a persistent connection if agreeable to the peer.

With persistent connections disabled, the Stateless Raw XML Handler refuses to negotiate the establishment of persistent connections. Alternatively, use the **no persistent-connections** command.

Examples

- Disables persistent connection negotiation.

```
# persistent-connections off
#
```

or

```
# no persistent-connections
#
```

- Enables persistent connection negotiation, which restores the default state.

```
# persistent-connections on
#
```

or

```
# persistent-connections
#
```

port

Specifies the TCP port to monitor for client requests.

Syntax

port *port*

Parameters

port Binds the Stateless Raw XML Handler to a specific port.

Guidelines

This command only sets the TCP port for the Stateless Raw XML Handler. This port applies to all configured local addresses. Use the **local-address** command to set the IP address on which the handler listens.

Related Commands

local-address, **remote-address**, **remote-port**

Examples

- Binds the current Stateless Raw XML Handler to the specified port number.

```
# port 16000
#
```

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

no ssl *name*

Parameters

name Specifies the name of the existing SSL Proxy Profile.

Guidelines

Assignment of an SSL Proxy Profile provides a secure (SSL-enabled) connection. The SSL Proxy Profile must already exist in the current application domain. Use the Global configuration mode command **sslproxy** to create a new SSL Proxy Profile.

Use the **no ssl** command to remove the SSL Proxy Profile assignment.

Examples

- Assigns the SSL-1 SSL Proxy to the current Stateless Raw XML Handler.

```
# ssl SSL-1
#
```

- Removes the assignment of the SSL-1 SSL Proxy from the current Stateless Raw XML Handler.

```
# no ssl SSL-1
#
```

Chapter 83. System Settings configuration mode

This chapter provides an alphabetic list of commands for System Settings configuration mode.

To enter this configuration mode, use the Global **system** command.

Most, but not all, of the commands in "Common commands" on page 2 and in Chapter 129, "Monitoring commands," on page 1053 are available in this configuration mode.

audit-reserve

Reserves disk space for the audit log.

Syntax

audit-reserve *kilobytes*

Parameters

kilobytes

Specifies the amount of disk space in kilobytes to reserve for the audit log. The reserve space must be at least four kilobytes less than the total amount of free space that is currently available on the file system. The value 0 indicates that the reserve function is disabled. Use an integer in the range of 0 through 10000. The default is 40.

Guidelines

The **audit-reserve** command specifies the amount of disk space in kilobytes to reserve for the audit log. Use this command to alter the amount of disk space to reserve to prevent the loss of audit events in case of a full disk. This function is disabled if the value is 0.

If the appliance is forced to release the audit reserve:

- All data services will be forced into an operational `down` state and cease to process traffic.
- All administrative services, such as the WebGUI, Telnet, and so forth, will continue to work.

When the appliance forces the release, the log will contain a message that states that the disk space for audit events is low.

Before restoring the appliance to service, a privileged administrator needs to free up disk space. When there is enough available disk space for normal operations, the administration can restart the appliance, which will resume the processing of traffic.

contact

Identifies the person or function responsible for appliance maintenance.

Syntax

contact *contact*

Parameters

contact Identifies the person or function responsible for appliance maintenance.

Guidelines

The **contact** command identifies the person who is responsible for managing the appliance. This information identifies the person who is responsible for managing this appliance by name, telephone number, email address, or a combination of these items.

Related Commands

location, **name**

Examples

- Specifies a system contact.

```
# contact "Tector Gorch; 555.555.5555; tector.gorch@datapower.com"
#Custom User Interface File
```

---

custom-ui-file

Specifies the URL of the custom user interface file.

Syntax

custom-ui-file *URL*

no custom-ui-file

Parameters

URL Specifies the location of the file on the appliance.

Guidelines

The **custom-ui-file** command specifies the location of the custom user interface file. The file must reside in the local: or store: directory on the appliance. The file cannot reside on a mounted file system, such as iSCSI.

This XML file contains custom user interface messages to display in the WebGUI and from the command line. This file also defines the custom prompt for the command line. After creating the custom user interface file, use the **test schema** command to validate that the XML file is conformant with the `dp-user-interface.xsd` schema.

Use the **no custom-ui-file** command to remove the use of custom messages and the command line prompt that are defined in the custom user interface file.

For information on creating a custom user interface file, refer to the *IBM WebSphere DataPower SOA Appliances: Administrators Guide*.

Related Commands

test schema

Examples

Specifies the `xyzbanner.xml` file in the `store:` directory as the custom user interface file.

```
# custom-ui-file store:///xyzbanner.xml
#
```

entitlement

Specifies the original serial number.

Syntax

entitlement *original-serial-number*

Parameters

original-serial-number
Specifies the original serial number.

Guidelines

The **entitlement** command specifies the serial number of the original appliance after receiving a replacement appliance. Without the serial number of the original appliance, IBM cannot entitle the replacement appliance for future maintenance or warranty service.

location

Specifies the location of the appliance.

Syntax

location *location*

Parameters

location
Specifies the appliance location.

Guidelines

The **location** command identifies the location of the appliance.

Related Commands

contact, contact name

Examples

- Specifies the appliance location.

```
# location "Corporate Headquarters"
#
```

name

Specifies an identifier for the appliance.

Syntax

name *identifier*

Parameters

identifier
Specifies the identifer. Use a string up to 127 characters in length.

Guidelines

The **name** command specifies the system identifier of the appliance. When the custom user interface file defines the command line extension, this identifier is added before the prompt.

Examples

- Specifies the name of the appliance.

```
# name Duluth
#
```

- Specifies the name of the appliance. Use double quotes (“”) to bracket a appliance name that contains spaces.

```
# name "Tango Lake"
#
```

Chapter 84. TAM configuration mode

This chapter provides an alphabetic listing of commands that are available in TAM configuration mode. TAM is an abbreviation for IBM Tivoli Access Manager.

To enter this configuration mode, use the Global **tam** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in TAM configuration mode.

file

Specifies the location of the TAM configuration file.

Syntax

file *name*

Parameters

name Specifies the name of the TAM configuration file.

ldap-ssl-key-file

Specifies the location of an SSL key file that contains a certificate used for LDAP access

Syntax

ldap-ssl-key-file *name*

Parameters

name Specifies the name of the TAM SSL key file to use for LDAP server access.

Guidelines

The TAM client application configuration file must contain an `[ldap] ssl-keyfile-pwd` to access the file specified by this command. This key file contains the LDAP server certificate and handles the certificates that are used in LDAP communication.

Applicable only when **use-ldap-ssl** is **on**.

Related Commands

ldap-ssl-key-file-password, **use-ldap-ssl**

ldap-ssl-key-file-dn

Specifies the subject DN of the certificate.

Syntax

ldap-ssl-key-file-dn *label*

Parameters

label Specifies the subject DN of the certificate.

Guidelines

The **ldap-ssl-key-file-dn** command specifies the subject DN of the certificate. When using client-side SSL and the key file contains multiple certificates, the DN specifies which certificate to use. This property is relevant for mutually-authenticated SSL only.

Applicable only when **use-ldap-ssl** is **on**.

Related Commands

use-ldap-ssl

---

ldap-ssl-key-file-password

Specifies the password for the LDAP key file.

Syntax

ldap-ssl-key-file-password *password*

Parameters

password
Specifies the password for the LDAP key file.

Guidelines

Applicable only when **use-ldap-ssl** is **on**.

Related Commands

ldap-ssl-key-file, **use-ldap-ssl**

---

ldap-ssl-port

Specifies the SSL port for the LDAP server.

Syntax

ldap-ssl-port *port*

Parameters

port Specifies the port number on the LDAP server for SSL communication. The default is 636.

Guidelines

Applicable only when **use-ldap-ssl** is **on**.

Related Commands

use-ldap-ssl

ssl-key

Specifies the location of the TAM SSL key file.

Syntax

ssl-key *name*

Parameters

name Specifies the name of the TAM SSL key file.

ssl-key-stash

Specifies the location of the TAM SSL key password stash file.

Syntax

ssl-key *name*

Parameters

name Specifies the name of the TAM SSL key password stash file.

use-fips

Determines whether to enable FIPS mode.

Syntax

use-fips {**yes** | **no**}

Parameters

yes Uses TLS version 1 as the secure communication protocol.

no (Default) Uses SSL version 3 as the secure communication protocol.

Guidelines

The **use-fips** command determines whether to enable Federal Information Processing Standard (FIPS) mode for secure communication between the DataPower appliance (the TAM client) and the TAM authorization server.

In the following situations, the TAM client will be down after configuration, because the TAM client cannot establish a secure connection with the TAM server:

- If the TAM server requires TLS and the TAM client uses SSL.
- If the TAM server requires SSL and the TAM client uses TLS.

use-ldap-ssl

Indicates whether the connection to the LDAP server uses SSL.

Syntax

use-ldap-ssl {**on** | **off**}

Parameters

on The connection is secured by SSL.

off The connection is not secure.

Related Commands

ldap-ssl-key-file, **ldap-ssl-key-file-dn**, **ldap-ssl-key-file-password**, **ldap-ssl-port**

Chapter 85. TFIM configuration mode

This chapter provides an alphabetic listing of commands that are available in TFIM configuration mode. TFIM is the abbreviation for IBM Tivoli Federated Identity Manager.

To enter this configuration mode, use the Global **tfim** command.

The DataPower appliance integrates with TFIM through the exchange of WS-Trust SOAP messages. The TFIM management object centralizes the configuration of the TFIM endpoint and prevents parameter duplication between the Map Credential and the Post Processing phases in AAA. During the Map Credential phase, an authenticated identity can be mapped to the identity used for authorization. During the Post Processing phase, an authorized identity can be mapped to the output AAA identity.

When integrating with TFIM, the provided input credentials must be able to be expressed in the request token format that is configured for the TFIM endpoint. For example, a WS-Security Username TokenType that is to be used as the request token cannot be created when the available user credential is an X.509 certificate.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in TAM configuration mode.

tfim-60-req-tokenformat

Specifies the format of the TFIM 6.0 request token.

Syntax

tfim-60-req-tokenformat *format*

Parameters

format Specifies the format of the token. Only the following values are supported:

Custom
Indicates a custom token. When specified, requires the use of the **tfim-custom-req-url** command

SAML1.0
Indicates a SAML Assertion 1.0

SAML1.1
Indicates a SAML Assertion 1.1

WSUserNameToken
(Default) Indicates a WS-Security Username Token

Guidelines

The **tfim-60-req-tokenformat** command is required when **tfim-compatible** is **v6.0**; otherwise, it is ignored.

Related Commands

tfim-compatible, **tfim-custom-req-url**

Examples

- Indicates that the request token format for TFIM version 6.0 is SAML Assertion 1.0.

```
# tfim-compatible v6.0
# tfim-60-req-tokenformat SAML1.0
#
```

- Indicates that the request token format for TFIM version 6.0 is a custom token that is defined in the specified style sheet.

```
# tfim-compatible v6.0
# tfim-60-req-tokenformat custom
# tfim-custom-req-url local:///tfim-custom.xsl
#
```

tfim-61-req-tokenformat

Specifies the format of the TFIM 6.1 request token.

Syntax

tfim-61-req-tokenformat *format*

Parameters

format Specifies the format of the token. Only the following values are supported:

BinarySecurityToken
Indicates a WS-Security BinarySecurityToken.

Custom
Indicates the use of a style sheet to generate TFIM requests. When specified, requires the use of the **tfim-custom-req-url** command

CustomToken
Indicates a custom token

SAML1.0
Indicates a SAML Assertion 1.0

SAML1.1
Indicates a SAML Assertion 1.1

SAML2.0
Indicates a SAML Assertion 2.0

WSUserNameToken
(Default) Indicates a WS-Security Username Token

WSKerberosToken
Indicates a WS-Security Kerberos Token

WSX509Token
Indicates a WS-Security X.509 Token

Guidelines

The **tfim-61-req-tokenformat** command is required when **tfim-compatible** is **v6.1**; otherwise, it is ignored.

Related Commands

tfim-compatible, **tfim-custom-req-url**

Examples

- Indicates that the request token format for TFIM version 6.1 is a WS-Security X.509 Token.

```
# tfim-compatible v6.1
# tfim-61-req-tokenformat WSX509Token
#
```

- Indicates that the request token format for TFIM version 6.1 is a custom token that is defined in the specified style sheet.

```
# tfim-compatible v6.1
# tfim-61-req-tokenformat custom
# tfim-custom-req-url local:///tfim-custom.xsl
#
```

tfim-62-req-tokenformat

Specifies the format of the TFIM 6.2 request token.

Syntax

tfim-62-req-tokenformat *format*

Parameters

format Specifies the format of the token. Only the following values are supported:

BinarySecurityToken
Indicates a WS-Security BinarySecurityToken.

Custom
Indicates the use of a style sheet to generate TFIM requests. When specified, requires the use of the **tfim-custom-req-url** command

CustomToken
Indicates a custom token

SAML1.0
Indicates a SAML Assertion 1.0

SAML1.1
Indicates a SAML Assertion 1.1

SAML2.0
Indicates a SAML Assertion 2.0

WSUserNameToken
(Default) Indicates a WS-Security Username Token

WSKerberosToken
Indicates a WS-Security Kerberos Token

WSX509Token
Indicates a WS-Security X.509 Token

Guidelines

The **tfim-62-req-tokenformat** command is required when **tfim-compatible** is **v6.2**; otherwise, it is ignored.

Related Commands

tfim-compatible, **tfim-custom-req-url**

Examples

- Indicates that the request token format for TFIM version 6.2 is a WS-Security X.509 Token.

```
# tfim-compatible v6.2
# tfim-62-req-tokenformat WSX509Token
#
```

- Indicates that the request token format for TFIM version 6.2 is a custom token that is defined in the specified style sheet.

```
# tfim-compatible v6.2
# tfim-62-req-tokenformat custom
# tfim-custom-req-url local:///tfim-custom.xsl
#
```

---

tfim-addr

Specifies the address of the TFIM server.

Syntax

tfim-addr *address*

Parameters

address Specifies the host name or IP address of the TFIM server.

Guidelines

The **tfim-addr** command specifies the host name or IP address of the TFIM server.

Related Commands

tfim-port

Examples

- Indicates that `FIMHost.ibm.com` is the fully qualified host name of the TFIM server and that this server is using the port 9080 (the default port).

```
# tfim-addr FIMHost.ibm.com
#
```

- Indicates that 9.33.97.251 is the IP address of the TFIM server and that this server is using port 19080.

```
# tfim-addr 9.33.97.251
# tfim-port 19080
#
```

---

tfim-compatible

Indicates the version of TFIM.

Syntax

tfim-compatible {**v6.0** | **v6.1** | **v6.2**}

Parameters

v6.0 Indicates Tivoli Federated Identity Manager, version 6.0.

v6.1 Indicates Tivoli Federated Identity Manager, version 6.1.

v6.2 Indicates Tivoli Federated Identity Manager, version 6.2.

Guidelines

The **tfim-compatible** command indicates the currently configured version of Tivoli Federated Identity Manager. The specified value determines the details for the namespace and WS-Trust messages.

Selecting Version 6.2 as the compatibility mode will cause the TFIM client/endpoint to generate WS-Trust messages using version 1.3 of the WS-Trust specification. In this case, trust chains in the TFIM 6.2 server must use the Validate OASIS URI as the Request Type. To use WS-Trust version 1.2 messages with a TFIM 6.2 server, select TFIM 6.1 as the compatibility mode. If the 6.1 compatibility mode is selected, TFIM 6.2 will behave the same as TFIM 6.1.

Examples

- Indicates that the current version of Tivoli Federated Identity Manager is version 6.1.

```
# tfim-compatible v6.1
#
```

---

tfim-custom-req-url

Specifies the location of the custom style sheet.

Syntax

tfim-custom-req-url *stylesheet*

Parameters

stylesheet
Specifies the location of the custom style sheet.

Guidelines

The **tfim-custom-req-url** command specifies the location of the custom style sheet that is used for TFIM requests.

This command is required when the request token type is **Custom**; otherwise, it is ignored.

Related Commands

tfim-60-req-tokenformat, **tfim-61-req-tokenformat**, **tfim-62-req-tokenformat**, **tfim-compatible**

Examples

- Indicates that the request token format for TFIM version 6.1 is a custom token that is defined in the specified style sheet.

```
# tfim-compatible v6.1
# tfim-61-req-tokenformat custom
# tfim-custom-req-url local:///tfim-custom.xsl
#
```

tfim-issuer

Specifies the identity that issued the request.

Syntax

tfim-issuer *issuer*

Parameters

issuer Specifies the identity that issued the request in the following format:

```
urn:itfim:wssm:tokenconsumer
```

Guidelines

The **tfim-issuer** command specifies the issuer of the request. In the WS-Security Management (WSSM) component, the issuer is either the WSSM token generator or the WSSM token consumer. To determine the correct value, consult your TFIM administrator.

This command is required for all TFIM request tokens except **Custom**; otherwise, it is ignored.

Related Commands

tfim-60-req-tokenformat, **tfim-61-req-tokenformat**, **tfim-62-req-tokenformat**, **tfim-compatible**, **tfim-operation**, **tfim-pathaddr**, **tfim-porttype**

Examples

- Indicates that the WSSM token consumer issued the request to access the TFIM web service located at `/itfim-wssm/wssm-default/EchoWSDL/EchoService` using the `EchoService` port type and the `echo` operation.

```
# tfim-issuer urn:itfim:wssm:tokenconsumer
# tfim-pathaddr /itfim-wssm/wssm-default/EchoWSDL/EchoService
# tfim-porttype EchoService
# tfim-operation echo
#
```

tfim-operation

Specifies the name of the Web services operation.

Syntax

tfim-operation *operation*

Parameters

operation

Specifies name of the Web services operation. For example:

- `echo`
- `whoami`

Guidelines

The **tfim-operation** command specifies the name of the Web services operation. To determine the correct value, consult your TFIM administrator.

This command is optional for all TFIM version 6.1 or 6.2 request tokens except **Custom**; otherwise, it is ignored.

Related Commands

tfim-61-req-tokenformat, **tfim-62-req-tokenformat**, **tfim-compatible**, **tfim-issuer**, **tfim-pathaddr**, **tfim-porttype**

Examples

- Indicates that the WSSM token consumer issued the request to access the TFIM web service located at `/itfim-wssm/wssm-default/EchoWSDL/EchoService` using the `EchoService` port type and the `echo` operation.

```
# tfim-issuer urn:itfim:wssm:tokenconsumer
# tfim-pathaddr /itfim-wssm/wssm-default/EchoWSDL/EchoService
# tfim-porttype EchoService
# tfim-operation echo
#
```

tfim-pathaddr

Specifies the scope for the security token.

Syntax

tfim-pathaddr *destination*

Parameters

destination

Specifies the scope for the security token. For example:

- `http://itfim.ibm.com:9080/EchoApplication/services/EchoServiceUsername`
- `http://9.33.97.251:9080/EchoApplication/services/EchoServiceUsername`

Guidelines

The **tfim-pathaddr** command specifies the scope for this security token. Within the TFIM service, this information specifies the destination of the request. The TFIM trust service uses this information to determine which partner is being accessed.

To determine the correct value, consult your TFIM administrator.

This command is required for all TFIM request tokens except **Custom**; otherwise, it is ignored.

Related Commands

tfim-60-req-tokenformat, **tfim-61-req-tokenformat**, **tfim-62-req-tokenformat**, **tfim-compatible**, **tfim-issuer**, **tfim-operation**, **tfim-porttype**

Examples

- Indicates that the WSSM token consumer issued the request to access the TFIM web service located at `/itfim-wssm/wssm-default/EchoWSDL/EchoService` using the `EchoService` port type and the `echo` operation.

```
# tfim-issuer urn:itfim:wssm:tokenconsumer
# tfim-pathaddr /itfim-wssm/wssm-default/EchoWSDL/EchoService
# tfim-porttype EchoService
# tfim-operation echo
#
```

---

tfim-port

Specifies the port number of the TFIM server.

Syntax

tfim-port *port*

Parameters

port Specifies the port of the TFIM server. The default is 9080.

Guidelines

The **tfim-port** command specifies the port number of the TFIM server.

Related Commands

tfim-addr

Examples

- Indicates that 9.33.97.251 is the IP address of the TFIM server and that this server is using port 19080.

```
# tfim-addr 9.33.97.251
# tfim-port 19080
#
```

---

tfim-porttype

Specifies the Web services port type.

Syntax

tfim-porttype *type*

Parameters

type Specifies the Web services port type. For example, `EchoService`.

Guidelines

The **tfim-porttype** command specifies the Web services port type to use. A port type is a group of Web services operations.

To determine the correct value, consult your TFIM administrator.

This command is optional for all TFIM version 6.1 or 6.2 request tokens except **Custom**; otherwise, it is ignored.

Related Commands

tfim-61-req-tokenformat, **tfim-62-req-tokenformat**, **tfim-compatible**, **tfim-issuer**, **tfim-operation**, **tfim-pathaddr**

Examples

- Indicates that the WSSM token consumer issued the request to access the TFIM web service located at `/itfim-wssm/wssm-default/EchoWSDL/EchoService` using the `EchoService` port type and the `echo` operation.

```
# tfim-issuer urn:itfim:wssm:tokenconsumer
# tfim-pathaddr /itfim-wssm/wssm-default/EchoWSDL/EchoService
# tfim-porttype EchoService
# tfim-operation echo
#
```

tfim-schema-validate

Indicates whether TFIM responses are schema-validated.

Syntax

tfim-schema-validate {**on** | **off**}

Parameters

on Indicates that TFIM responses are schema-validated.

off (Default) Indicates that TFIM responses are not schema-validated.

Guidelines

The **tfim-schema-validate** command indicates whether TFIM responses are schema-validated. When validating TFIM responses, the response is validated against the WS-Trust version indicated by the **tfim-compatible** command.

Related Commands

tfim-compatible

Examples

- Indicates that TFIM responses are schema-validated against the configured WS-Trust version.

```
# tfim-schema-validate on
#
```

tfim-sslproxy

Specifies the SSL Proxy Profile to manage SSL communications.

Syntax

tfim-sslproxy *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **tfim-sslproxy** command specifies the name of an existing SSL Proxy Profile to manage SSL communications with peers. The SSL Proxy Profile identifies the keys and certificates that are used in the handshake.

Examples

- Specifies that `TFIM-SSLProxy-1` is the SSL Proxy Profile to manage SSL communications with peers.

```
# tfim-sslproxy TFIM-SSLProxy-1
#
```

Chapter 86. Telnet Service configuration mode

This chapter provides an alphabetic listing of commands that are available in Telnet Service configuration mode.

To enter this configuration mode, use the Global **cli telnet** command. While in Telnet configuration mode, define a Telnet server that supports client-initiated access to the command line interface.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Telnet Service configuration mode.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

Parameters

name Specifics the name of an ACL.

Guidelines

Assignment of an ACL to a Telnet Service is optional. If an ACL is assigned to the service, only those IP addresses specifically allowed by the ACL can initiate Telnet access to the appliance; if an ACL is not assigned, Telnet access to the appliance is unrestricted.

Related Commands

acl (Global), **allow** (ACL), **deny** (ACL)

ip-address

Specifies the local IP address to monitor for incoming CLI traffic.

Syntax

ip-address {*address* | **0**}

Parameters

address Specifies the IP address (primary or secondary) of a DataPower Ethernet interface.

0 Indicates all DataPower Ethernet interfaces.

Guidelines

In conjunction with the **port** command, identifies the IP addresses and ports that the Telnet service monitors.

Related Commands

port

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current Telnet service monitor.

```
# cli telnet telnet-1
Telnet Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

---

port

Specifies the local port to monitor for incoming CLI traffic.

Syntax

port *port*

Parameters

port Specifies the port on one or all IP interfaces. Use an integer in the range of 0 through 65535.

Guidelines

In conjunction with the **ip-address** command, identifies the IP address and port that the Telnet service monitors.

Related Commands

ip-address

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current Telnet service monitor.

```
# cli telnet telnet-1
Telnet Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

Chapter 87. Throttle Settings configuration mode

This chapter provides an alphabetic listing of commands that are available in Throttle Settings configuration mode.

To enter this configuration mode, use the Global **throttle** command. While in Throttle Settings configuration mode, you define a thresholds settings for the DataPower appliance.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Throttle Setting configuration mode.

memory-terminate

Specifies the memory kill-threshold.

Syntax

memory-terminate *percent*

Parameters

percent Specifies the percentage of minimum free memory. Use an integer in the range of 0 through 100. The default is 5.

Guidelines

The **memory-terminate** command specifies the memory kill-threshold. This threshold is the point at which the appliance reboots. The appliance reboots after the duration defined by the **timeout** command.

Related Commands

memory-throttle, **timeout**

memory-throttle

Specifies the memory throttle-threshold.

Syntax

memory-throttle *percent*

Parameters

percent Specifies the percentage of minimum free memory. Use an integer in the range of 0 through 100. The default is 20.

Guidelines

The **memory-throttle** command specifies the memory throttle-threshold. This threshold is the point at which the appliance stops accepting new connections. No new connection is accepted for the duration defined by the **timeout** command.

Related Commands

memory-terminate, **timeout**

qcode-warn

Specifies the namespace-threshold for QCodes.

Syntax

qcode-warn *percent*

Parameters

percent Specifies the percentage of available namespace QCodes. Use an integer in the range of 5 through 100. The default is 10.

Guidelines

The **qcode-warn** command specifies the *namespace-threshold*. This threshold is the point at which the number of available QCodes fall below the namespace-threshold (a measure of free QCodes expressed as a percentage of the total QCodes), the appliance writes an alert to the log. This message indicates that the appliance detected a shortage of free QCodes. When you receive this alert, the percentage of available QCodes is below the defined threshold. After you receive this alert, schedule a system reboot as soon as possible to prevent an unscheduled reboot. If the available QCodes is less than 5% available, the system reboots.

sensors-log

Controls the collection of environmental log messages.

Syntax

sensors-log {**on** | **off**}

Parameters

on (Default) Enables the collection of environmental log messages.

off Disables the collection of environmental log messages.

Guidelines

The **sensors-log** command controls the collection of environment log messages. The firmware monitors system fan speed and power module status. When enabled, log messages are generated when anomalous conditions are detected.

status-log

Controls the collection of throttle log messages.

Syntax

status-log {**on** | **off**}

Parameters

on (Default) Enables throttle settings log messages.

off Disables throttle settings log messages.

Guidelines

The **status-log** command controls the collection of throttle log messages. These messages pertain to available memory, available temporary file space, and available namespace QCodes. The criticality of these messages is set by the value of the **status-loglevel** command.

Related Commands

status-loglevel

---

status-loglevel

Sets the criticality of throttle messages.

Syntax

status-loglevel *priority*

Parameters

priority

Specifies the message priority. Use one of the following keywords or integers:

- **emerg** or 0
- **alert** or 1
- **critic** or 2
- **error** or 3
- **warn** or 4
- **notice** or 5
- **info** or 6
- **debug** or 7 (Default)

Guidelines

The **status-loglevel** command sets the criticality of throttle messages. This command is meaningful only if the **status-log** command is set to **on**.

Related Commands

status-log

---

temp-fs-terminate

Specifies the temporary file space kill-threshold.

Syntax

temp-fs-terminate *percent*

Parameters

percent Specifies the minimum percentage of free temporary file space. Use an integer in the range of 0 through 100. The default is 2.

Guidelines

The **memory-terminate** command specifies the free temporary file space kill-threshold. This threshold is the point at which the appliance reboots. The appliance reboots after the duration defined by the **timeout** command.

Related Commands

temp-fs-throttle, **timeout**

---

temp-fs-throttle

Specifies the temporary file space throttle-threshold.

Syntax

temp-fs-throttle *percent*

Parameters

percent Specifies the minimum percentage of free temporary file space. Use an integer in the range of 0 through 100. The default is 5.

Guidelines

The **temp-fs-throttle** command specifies the available temporary space throttle-threshold. This threshold is the point at which the appliance stops accepting new connections. No new connection is accepted for the duration defined by the **timeout** command.

Related Commands

temp-fs-terminate, **timeout**

---

timeout

Specifies the interval between the threshold trigger and the subsequent action.

Syntax

timeout *seconds*

Parameters

seconds

Specifies the interval, in seconds. The value is unbounded. The default is 30.

Guidelines

The **timeout** command specifies the interval between the threshold trigger and the subsequent action.

- If the action is throttle, the interval is how long to stop accepting connections.
- If the action is terminate, the interval is how long before the appliance reboots.

Related Commands

memory-terminate, **memory-throttle**, **temp-fs-terminate**, **temp-fs-throttle**

Examples

- Specifies that the appliance reboots 20 seconds after free memory drops to 10% of total memory.

```
# throttle
Throttle Settings configuration mode
# memory-terminate 10
# timeout 20
#
```

Chapter 88. TIBCO EMS configuration mode

This chapter provides an alphabetic listing of commands that are available in TIBCO EMS configuration mode.

To enter this configuration mode, use the Global **tibems-server** command. While in this mode, define the parameters to locate and to access a TIBCO EMS server. The TIBCO EMS server is used in conjunction with a TIBCO Front Side Handler.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in TIBCO EMS configuration mode.

auto-retry

Enables an automatic critical error-recovery procedure that attempts to reestablish a connection that has been broken in response to an error condition.

Syntax

auto-retry {**on** | **off**}

Parameters

on (Default) Enables error recovery.

off Disables error recovery.

Guidelines

The **auto-retry** command enables or disables automatic critical error-recovery procedure that attempts to reestablish a connection that has been broken in response to an error condition

Related Commands

retry-interval

connection-client-id

Identifies the connection client.

Syntax

connection-client-id *string*

Parameters

string Specify the string that identifies the connection client.

Guidelines

The **connection-client-id** command specifies the string to set as the TIBCO EMS Connection `clientID`.

default-message-type

Specifies the default JMS message type.

Syntax

default-message-type {**byte** | **text**}

Parameters

byte (Default) Specifies that the message payload is accessed as a Java™ byte array.

text Specifies that the message payload is accessed as a Java string value.

Guidelines

The **default-message-type** command specifies the default JMS message type. This message type is provided by the TIBCO EMS object only if the message type cannot be determined from the JMS message headers.

enable-logging

Controls expanded JMS logging facility.

Syntax

enable-logging {**on** | **off**}

Parameters

on Enables expanded JMS-specific logging.

off (Default) Disables expanded JMS-specific logging.

hostname

Identifies the TIBCO EMS server by domain name or IP address.

Syntax

hostname *server*[:*port*]

Parameters

server[:*port*]
Specifies domain name or IP address with the listening port of the TIBCO EMS server. Without a port specification, the default is port 7222.

Guidelines

The **hostname** command identifies the TIBCO EMS server by domain name or IP address. This command is ignored when used with the **loadbalance-faulttolerance** command.

Related Commands

loadbalance-faulttolerance

Examples

- Sets `ragnarok.datapower.com` as the host name to identify the target TIBCO EMS server.

```
# tibems-server TIBCO-1
Tibco EMS configuration mode
# hostname ragnarok.datapower.com
#
```

- Uses the IP address 192.168.13.35 to identify the target TIBCO EMS server.

```
# tibems-server TIBCO-2
Tibco EMS configuration mode
# hostname 192.168.13.35
#
```

---

load-balancing-algorithm

Identifies the algorithm for load-balancing.

Syntax

load-balancing-algorithm {**byte-rate** | **least-connections** | **none** }

Parameters

byte-rate
Creates a connection to the server that has the lowest total byte rate (input and output).

least-connections
Creates a connection to the server that has the least number of active connections.

none (Default) Load-balancing is not enabled.

Guidelines

The **load-balancing-algorithm** command identifies the algorithm for load-balancing. This command is relevant only when used with the **loadbalance-faulttolerance** command.

Related Commands

loadbalance-faulttolerance

Examples

- Creates a connection to the TIBCO EMS server `192.168.45.32` or `192.168.45.33` that has the fewest number of connections.

```
# tibems-server TIBCO-1
Tibco EMS configuration mode
# load-balancing-algorithm least-connections
# loadbalance-faulttolerance 192.168.45.32
# loadbalance-faulttolerance 192.168.45.33
#
```

- Creates a connection to the TIBCO EMS server `192.168.45.32`. Although server `192.168.45.33` is defined, the algorithm is `none`. Therefore, the connection is to the first listed server.

```
# tibems-server TIBCO-1
Tibco EMS configuration mode
# load-balancing-algorithm none
# loadbalance-faulttolerance 192.168.45.32
# loadbalance-faulttolerance 192.168.45.33
#
```

loadbalancing-faulttolerance

Defines load-balancing and fault-tolerance capabilities.

Syntax

For fault-tolerance

loadbalancing-faulttolerance *server*[:*port*]

For load-balancing

loadbalancing-faulttolerance *server*[:*port*] *server*[:*port*]

Parameters

server[:*port*]

Specifies domain name or IP address with the listening port of the TIBCO EMS server. Without a port specification, the default is port 7222.

Guidelines

The **loadbalancing-faulttolerance** command defines load-balancing and fault-tolerance capabilities. This command overrides the setting of the **hostname** command.

In a load-balancing situation:

- Use the **load-balancing-algorithm** to identify the algorithm to use.
- Use the **loadbalancing-faulttolerance** command to define each member server.

In a fault-tolerance situation, use **loadbalancing-faulttolerance** command to define each primary-backup server pair. The first server is the primary server, and the second server is the backup server.

Load-balancing takes precedence over fault-tolerance.

Related Commands

hostname, **load-balancing-algorithm**

Examples

- Creates a connection to the TIBCO EMS server `192.168.45.32`, `192.168.45.33`, `192.168.45.34`, `192.168.45.35`, or `192.168.45.36` that has the fewest number of connections.

```
# tibems-server TIBCO-1
Tibco EMS configuration mode
# load-balancing-algorithm least-connections
# loadbalance-faulttolerance 192.168.45.32
# loadbalance-faulttolerance 192.168.45.33
# loadbalance-faulttolerance 192.168.45.34
# loadbalance-faulttolerance 192.168.45.35
# loadbalance-faulttolerance 192.168.45.36
#
```

- Identifies a fault-tolerant pair of TIBCO EMS servers.

```
# tibems-server TIBCO-1
Tibco EMS configuration mode
# loadbalance-faulttolerance 192.168.45.32 192.168.45.33
#
```

- Identifies two fault-tolerant pairs of TIBCO EMS servers and one non-fault-tolerant server.

```
# tibems-server TIBCO-1
Tibco EMS configuration mode
# load-balancing-algorithm least-connections
# loadbalance-faulttolerance 192.168.45.32 192.168.45.33
# loadbalance-faulttolerance 192.168.45.34 192.168.45.35
# loadbalance-faulttolerance 192.168.45.36
#
```

maximum-message-size

Specifies the maximum message size.

Syntax

maximum-message-size *bytes*

Parameters

bytes Specifies the maximum message size in bytes Use an integer in the range of 0 through 1073741824. The default is 1048576 (1 MB). A value of 0 disables the enforcement of a maximum message size.

Related Commands

memory-threshold

memory-threshold

Specifies the maximum memory allocation for pending messages.

Syntax

memory-threshold *bytes*

Parameters

bytes Specifies the maximum memory to allocate in bytes. Use an integer in the range of 1048576 through 1073741824. This default is 268435456.

Related Commands

maximum-message-size

password

Specifies the password to access the TIBCO EMS server.

Syntax

password *string*

Parameters

string Specifies the password to access the remote server.

Guidelines

The **password** command specifies the password to use in conjunction with the value provided by the **username** command to access the remote server.

Related Commands

username

---

retry-interval

Specifies the interval between attempts to reestablish a connection.

Syntax

retry-interval *seconds*

Parameters

seconds
Specifies the interval, in seconds, between attempts to reestablish a downed connection. The default is 1.

Related Commands

auto-retry

---

sessions-per-connection

Specifies the maximum number of concurrent multiplexed sessions that a single connection can support.

Syntax

sessions-per-connection *sessions*

Parameters

sessions
Specifies the maximum concurrent sessions to support. Use an integer greater than 4. The default is 20.

Guidelines

Session requests in excess of the value of the **sessions-per-connection** command trigger the establishment of a new connection to the server. A new connection cannot be established unless the number of current connections is less than the value of the **total-connection-limit** command.

Assume default values (20 sessions per connection and 5 total connections) and 3 active fully-subscribed connections, a new session request generates the establishment of a 4th connection.

Related Commands

total-connection-limit

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Identifies the existing SSL Proxy Profile. In the absence of an explicitly assigned SSL Proxy Profile, the firmware establishes a nonsecure connection to the server.

total-connection-limit

Specifies the maximum number of open connections to the server.

Syntax

total-connection-limit *connections*

Parameters

connections
Specifies the maximum number of open connections that can be established to the server. The minimum is 1. The default is 5.

Related Commands

sessions-per-connection

transactional

Enables or disables transaction-based processing.

Syntax

transactional {**on** | **off**}

Parameters

on Enables transaction-based processing.

off Disables transaction-based processing.

Guidelines

The **transactional** command enables or disables transactional processing. When transactional processing is enabled, messages are acknowledged only after the transaction succeeds.

username

Specifies the account name to access the server.

Syntax

username *string*

Parameters

string Specifies the account name to use in conjunction with the value of the **password** command to access the server.

Related Commands

password

Chapter 89. TIBCO Front Side Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in TIBCO Front Side Handler configuration mode.

To enter this configuration mode, use the Global **source-tibems** command. While in this mode, define the client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

get-queue

Specifies the name of the queue that contains client-originated TIBCO EMS request messages.

Syntax

get-queue *name*

Parameters

name Specifies the name of the queue that contains TIBCO EMS request messages.

Related Commands

put-queue, **selector**

Guidelines

Identification of a GET queue is required.

The TIBCO Front Side Handler monitors the GET queue for incoming client requests. Upon message receipt, the handler forwards the extracted message to the a local TIBCO EMS object that will gateway the message to a remote TIBCO EMS server.

Examples

- Enter TIBCO Front Side Handler configuration mode to create the `tibcoFSH` TIBCO Front Side Handler. Identifies the `tibcoEmsRequest` queue as the GET queue that contains client-originated request messages.

```
# source-tibems tibcoFSH
New Tibco Front Side Handler configuration
# get-queue tibcoEmsRequest
#
```

put-queue

Specifies the name of the queue that contains server-originated TIBCO EMS reply messages.

Syntax

put-queue *name*

Parameters

name Specifies the name of the queue that contains TIBCO EMS reply messages.

Related Commands

get-queue

Guidelines

TIBCO EMS reply messages are originated by a remote TIBCO EMS server and put into this queue by a local TIBCO EMS object.

Identification of a PUT queue is optional. A PUT queue should be configured if server replies are expected. If reply messages are not expected, a PUT queue need not be configured. In the absence of a PUT queue, any received replies are dropped.

Examples

- Identifies the `tibcoEmsResponse` queue as the PUT queue that contains server-originated response messages.

```
# put-queue tibcoEmsResponse
#
```

---

selector

Specifies the SQL-like expression to filter messages in the GET queue.

Syntax

selector *expression*

Parameters

expression
Defines the SQL-like expression to select messages in the GET queue.

Guidelines

The TIBCO EMS Message Selector is a conditional expression based on a subset of SQL92 conditional expression syntax. The conditional expression enables the TIBCO EMS Front Side Handler to identify *messages of interest*.

The conditional expression does not operate on the body of the message, rather is examines the required TIBCO EMS headers and EMS properties (proprietary user-created headers that can appear between the required headers and the message body). The required TIBCO EMS headers are as follows:

`Destination`
Contains the destination (queue) to which the message is being sent

`DeliveryMode`
Contains the delivery mode (`PERSISTENT` or `NON_PERSISTENT`)

`Expiration`
Contains a message TTL or a value of 0 indicating an unlimited TTL

`Priority`
Contains the message priority expressed as a digit from 0 (lowest priority) to 9 (highest priority)

`MessageID`
Contains a unique message identifier starting with the prefix `ID:`, or a null value, effectively disabling message ID

`Timestamp`
Contains the time the message was handed off for transmission, not the time it was actually sent

`CorrelationID`
Contains a means of associating one message (for example, a response) with another message (for example, the original request)

`ReplyTo`
Contains the destination (queue) to which a reply to this message should be sent

`Type` Contains a message identifier provided by the application

`Redelivered`
Contains a Boolean indicating that the message has been delivered in the past, but not yet acknowledged

Configuration of a message selector is optional. If a message selector is not specified, all incoming client request messages are transferred by the TIBCO Front Side Handler to the TIBCO EMS object for processing.

If a message selector is specified, only those client requests that match the criteria specified by the SQL expression are forwarded to the TIBCO EMS object for processing. All other messages are dropped from the GET queue.

Related Commands

get-queue

Examples

- Indicates that only client requests that have a `DeliveryMode` of `PERSISTENT` are forwarded to the TIBCO EMS object for processing. All other messages are dropped from the GET queue.

```
# selector DeliveryMode LIKE PERSISTENT
#
```

server

Specifies the TIBCO EMS object supported by this protocol handler.

Syntax

server *name*

Parameters

name Specifies the name of the TIBCO EMS object.

Examples

- Designates the `emsServer-1` TIBCO EMS object as supported by this TIBCO Front Side Handler.

```
# server emsServer-1
#
```

Chapter 90. Timezone configuration mode

This chapter provides an alphabetic listing of commands that are available in Timezone configuration mode.

To enter this configuration mode, use the Global **timezone** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Timezone configuration mode.

custom

Specifies the name of a custom timezone.

Syntax

custom *name*

Parameters

name Specifies the name of the custom timezone.

daylight-name

Specifies the name of the timezone when in daylight savings time. This name is appended to the time display when applicable.

Syntax

daylight-name *name*

Parameters

name Specifies the name of the timezone when in daylight savings time.

Guidelines

Applies to the timezone set by the **name** or **custom** command.

daylight-offset

Specifies the offset, in hours, of daylight savings time.

Syntax

daylight-offset *hours*

Parameters

hours Specifies the offset (difference) in hours between daylight savings time and regular time.

Guidelines

Specifies the offset, in hours, of daylight savings time. This is typically 1, meaning that the clock moves forward or back 1 hour when the time boundary is crossed. Applies to the timezone that is identified by the **name** or **custom** command.

---

daylight-start-day

Specifies the day of the week when daylight savings time starts.

Syntax

daylight-start-day *day*

Parameters

day Specifies the day of the week when daylight savings time starts. Use one of the following keywords:

- **Monday**
- **Tuesday**
- **Wednesday**
- **Thursday**
- **Friday**
- **Saturday**
- **Sunday**

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-start-hours, **daylight-start-minutes**, **daylight-start-month**, **daylight-start-week**

Examples

- Sets `Sunday` as the day of the week when daylight savings time starts.

```
# daylight-start-day Sunday
#
```

---

daylight-start-hours

Specifies the hour of the day when daylight savings time starts.

Syntax

daylight-start-hours *hours*

Parameters

hours Specifies the hour of the day when daylight savings time starts. Use an integer between 0 and 23.

Guidelines

Uses the 24 hour clock. A setting of 2 is 2 AM; a setting of 14 is 2 PM. Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-start-day, **daylight-start-minutes**, **daylight-start-month**, **daylight-start-week**

Examples

- Sets 2 AM as the hour of the day when daylight savings time starts.

```
# daylight-start-hour 2
#
```

daylight-start-minutes

Specifies the minutes of the hour when daylight savings time starts.

Syntax

daylight-start-minutes *minutes*

Parameters

minutes
Specifies the minutes of the hour when daylight savings time starts. Use an integer between 0 and 59.

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-start-day, **daylight-start-hours**, **daylight-start-month**, **daylight-start-week**

Examples

- Sets 0 as the minutes of the hour when daylight savings time starts.

```
# daylight-start-minutes 0
#
```

daylight-start-month

Specifies the month of the year when daylight savings time starts.

Syntax

daylight-start-month *month*

Parameters

month Specifies the month of the year when daylight savings time starts. Use one of the following keywords:
- **January**
- **February**
- **March**
- **April**
- **May**
- **June**
- **July**
- **August**

- **September**
- **October**
- **November**
- **December**

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-start-day, **daylight-start-hours**, **daylight-start-minutes**, **daylight-start-week**

Examples

- Sets `April` as the month of the year when daylight savings time starts.

```
# daylight-start-month April
#
```

daylight-start-week

Specifies the week of the month when daylight savings time starts.

Syntax

daylight-start-week *week*

Parameters

week Specifies the week of the month when daylight savings time starts. Use an integer between 1 and 5.

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-start-day, **daylight-start-hours**, **daylight-start-minutes**, **daylight-start-month**

Examples

- Sets 1 as the week of the month when daylight savings time starts.

```
# daylight-start-week 1
#
```

daylight-stop-day

Specifies the day of the week when daylight savings time stops.

Syntax

daylight-stop-day *day*

Parameters

day Specifies the day of the week when daylight savings time stops. Use one of the following keywords:

- **Monday**
- **Tuesday**
- **Wednesday**
- **Thursday**
- **Friday**
- **Saturday**
- **Sunday**

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-stop-hours, **daylight-stop-minutes**, **daylight-stop-month**, **daylight-stop-week**

Examples

- Sets `Sunday` as the day of the week when daylight savings time stops.

```
# daylight-stop-day Sunday
#
```

daylight-stop-hours

Specifies the hour of the day when daylight savings time stops.

Syntax

daylight-stop-hours *hours*

Parameters

hours Specifies the hour of the day when daylight savings time stops. Use an integer between 0 and 23.

Guidelines

Uses the 24 hour clock. A setting of 2 is 2 AM. A setting of 14 is 2 PM. Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-stop-day, **daylight-stop-minutes**, **daylight-stop-month**, **daylight-stop-week**

Examples

- Sets 2 AM as the hour of the day when daylight savings time stops.

```
# daylight-stop-hour 2
#
```

daylight-stop-minutes

Specifies the minutes of the hour when daylight savings time stops.

Syntax

daylight-stop-minutes *minutes*

Parameters

minutes

Specifies the minutes of the hour when daylight savings time stops. Use an integer between 0 and 59.

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-stop-day, **daylight-stop-hours**, **daylight-stop-month**, **daylight-stop-week**

Examples

- Sets 0 as the minutes of the hour when daylight savings time stops.

```
# daylight-stop-minutes 0
#
```

---

daylight-stop-month

Specifies the month of the year when daylight savings time stops.

Syntax

daylight-stop-month *month*

Parameters

month Specifies the month of the year when daylight savings time stops. Use one of the following keywords:

- **January**
- **February**
- **March**
- **April**
- **May**
- **June**
- **July**
- **August**
- **September**
- **October**
- **November**
- **December**

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-stop-day, **daylight-stop-hours**, **daylight-stop-minutes**, **daylight-stop-week**

Examples

- Sets `October` as the month of the year when daylight savings time stops.

```
# daylight-stop-month October
#
```

daylight-stop-week

Specifies the week of the month when daylight savings time stops.

Syntax

daylight-stop-week *week*

Parameters

week Specifies the week of the month when daylight savings time stops. Use an integer between 1 and 5.

Guidelines

Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

daylight-stop-day, **daylight-stop-hours**, **daylight-stop-minutes**, **daylight-stop-month**

Examples

- Sets 5 as the week of the month when daylight savings time stops.

```
# daylight-stop-week 5
#
```

direction

Specifies the direction, relative to GMT, of the timezone. North America is West. Asia is East.

Syntax

direction *direction*

Parameters

direction

Specifies the direction, relative to GMT, of the timezone. Use one of the following keywords:

- **East**
- **West**

Guidelines

Determines whether the offset is added to (**West**) or subtracted from (**East**) GMT. A timezone that is in GMT could have an offset of 0. Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

offset-hours, **offset-minutes**

Examples

- Sets `East` as the direction, relative of GMT, for the current timezone.

```
# direction East
#
```

name

Specifies the name of the timezone. This name is appended to the displayed time.

Syntax

name *name*

Parameters

name Specifies the name of a preset timezone.

Value	Meaning	
HST10	Honolulu	10 hrs West of UTC, no DST
AKST9AKDT	Alaska	9 hrs West, US DST rules
PST8PDT	Pacific	8 hrs West, US DST rules
MST7MDT	Mountain	7 hrs West, US DST rules
CST6CDT	Central	6 hrs West, US DST rules
EST5EDT	Eastern	5 hrs West, US DST rules
AST4ADT	Atlantic	4 hrs West, Canada DST rules
UTC	Universal Time	UTC, no DST
GMT0BST	GMT	UTC, UK DST rules
CET-1CEST	Central Europe	1 hr East, EU DST rules
EET-2EEST	Eastern Europe	2 hrs East, EU DST rules
MKS-3MSD	Moscow Time	3 hrs East, Russian DST rules
AST-3	Saudi Arabia	3 hrs East, no DST
KRT-5	Pakistan	5 hrs East, no DST
IST-5:30	India	5:30 hrs East, no DST
CST-8	China	8 hrs East, no DST
WST-8	Western Australia	8 hrs East, no DST
JST-9	Japan	9 hrs East, no DST
CST-9:30	Central Australia	9:30 hrs East, no DST
EST-10	Eastern Australia	10 hrs East, no DST

Guidelines

Use one of the preset timezone names to automatically set the timezone and corresponding daylight values. **UTC** has no Daylight Savings Time, while **GMT0BST** is the same time but DST applies. Use the daylight-related commands to adjust the values set by using the **name** command.

Related Commands

All other commands

Examples

- Sets the current timezone to `IST-5:30` (India, 5:30 East of GMT, no DST).

```
# name IST-5:30
#
```

offset-hours

Specifies the offset in hours, relative to GMT, of the timezone.

Syntax

offset-hours *hours*

Parameters

hours Specifies the offset in hours, relative to GMT, of the timezone. Use an integer between 0 and 12.

Guidelines

Determines the number of hours the timezone is offset from GMT. Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

direction, **offset-minutes**

Examples

- Offsets the current timezone 5 hours relative to GMT.

```
# offset-hours 5
#
```

---

offset-minutes

Specifies the offset in minutes, relative to GMT, of the timezone.

Syntax

offset-minutes *minutes*

Parameters

minutes

Specifies the offset in minutes, relative to GMT, of the timezone. Use an integer between 0 and 59.

Guidelines

Determines the number of minutes the timezone is offset from GMT. Applies to the timezone that is identified by the **name** or **custom** command.

Related Commands

direction, **offset-hours**

Examples

- Offsets the current timezone 30 minutes relative to GMT.

```
# offset-minutes 30
#
```

Chapter 91. UDDI Registry configuration mode

This chapter provides an alphabetic listing of commands that are available in UDDI Registry configuration mode.

To enter this configuration mode, use the Global **uddi-registry** command. While in UDDI Registry configuration mode, define the parameters needed to locate and to access a UDDI Registry.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in UDDI Registry configuration mode.

hostname

Sets the IP address or hostname.

Syntax

hostname *host*

Parameters

host Specifies the IP address or host name of the remote UDDI Registry. If a host name, the appliance must have a method to resolve the name through DNS.

Related Commands

port

inquiry-url

Sets the URI to send inquiry requests.

Syntax

inquiry-url *URI*

Parameters

URI The local path (URI) portion of the URL used to query the Registry. UDDI inquiry requests will be sent to `http(s)://hostname:port/inquiry-url`. A typical default looks like `https://192.18.1.120:443/uddi/inquiry`.

Examples

- Enters UDDI Registry configuration mode to create the `Registry1` object. Sets the inquiry URI.

```
# uddi-registry Registry1
New UDDI Registry Registry1
# inquiry-url "/web/uddi/inquiry"
```

port

Sets the TCP port.

Syntax

port *port*

Parameters

port The TCP port number the Registry uses to listen for requests. The default is 80.

publish-url

Sets the URI to send Publish requests.

Syntax

publish-url *URI*

Parameters

URI Specifies the local path (URI) portion of the URL used to send Publish requests the Registry. UDDI inquiry requests will be sent to `http(s)://hostname:port/publish-url`. A typical default looks like `https://192.18.1.120:443/uddi/publish`.

Guidelines

The DataPower appliance can publish WSDL files to a UDDI Registry. This command sets the URI used by the appliance to send WSDL files to the Registry for publication. This operation is typically protected by SSL communications, requiring both an SSL Proxy Profile and setting the **use-ssl** value to **publish** at minimum.

Examples

- Enters UDDI Registry configuration mode to create the `Registry1` object. Sets the Publish URI.

```
# uddi-registry Registry1
New UDDI Registry Registry1
# publish-url "/web//uddi/publish"
```

security-url

Sets the URI to send security information requests.

Syntax

security-url *URI*

Parameters

URI The local path (URI) portion of the URL used to send Security-related requests the Registry. UDDI inquiry requests will be sent to `http(s)://hostname:port/security-url`. A typical default looks like `https://192.18.1.120:443/uddi/security`.

Examples

- Enters UDDI Registry configuration mode to create the `Registry1` object. Sets the Security URI.

```
# uddi-registry Registry1
New UDDI Registry Registry1
# security-url "/web/uddi/security"
```

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies name of an existing SSL Proxy Profile in the current application domain. This Profile determines the cryptographic keys used by the appliance when negotiating SSL communications with the Registry.

Guidelines

The SSL Proxy Profile set here must already exist. To create a new Profile for this use, or to alter an existing Profile for this use, employ the `sslproxy` Global Configuration command.

Examples

- Enters UDDI Registry configuration mode to create the `Registry1` object. Assigns the `StdProxyProfile` SSL Proxy Profile.

```
# uddi-registry Registry1
New UDDI Registry Registry1
# ssl StdProxyProfile
```

ssl-port

Sets the TCP port for HTTPS connections.

Syntax

ssl-port *port*

Parameters

port The TCP port number the Registry uses to listen for requests. The default is 443.

Examples

- Enters UDDI Registry configuration mode to create the `Registry1` object. Sets the TCP port to 8443.

```
# uddi-registry Registry1
New UDDI Registry Registry1
# ssl-port 8443
```

subscription-url

Sets the URI to request subscription information requests.

Syntax

subscription-url *URI*

Parameters

URI The local path (URI) portion of the URL used to send Subscription-related requests the Registry. UDDI inquiry requests will be sent to `http(s)://hostname:port/subscription-url`. A typical default looks like `https://192.18.1.120:443/uddi/subscription`.

Examples

- Enters UDDI Registry configuration mode to create the `Registry1` object. Sets the Security URI.

```
# uddi-registry Registry1
New UDDI Registry Registry1
# security-url "/web/uddi/subscription"
```

use-ssl

Determines when to use HTTPS connections.

Syntax

use-ssl {**always** | **publish**}

Parameters

always
Uses SSL for all communications.

publish
Uses SSL for Publish requests only.

version

Determines which level of the UDDI Specification.

Syntax

version {**UDDIv2** | **UDDIv3**}

Parameters

UDDIv2
Adheres to specification version 2 for communications.

UDDIv3
Adheres to specification version 3 for communications.

Chapter 92. UDDI Subscription configuration mode

This chapter provides an alphabetic listing of commands that are available in UDDI Subscription configuration mode.

To enter this configuration mode, use the **uddi-subscription** command. While in this mode, define the UDDI subscription.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in UDDI Subscription configuration mode.

key

Determines the remote registry subscription keys used by this object.

Syntax

key *key*

Parameters

key Specifies the subscription key, as defined on the remote UDDI registry.

Guidelines

Use this command as many times as needed to include all of the subscription keys desired for this object. Subscription keys are defined on the remote UDDI registry. The remote UDDI registry is defined with the **registry** command.

Related Commands

registry

Examples

- Creates the `ActivityEndpoint1` subscription, and sets the subscription key.

```
# uddi-subscription ActivityEndpoint1
New UDDI Subscription ActivityEndpoint1
# key uddi:8b071240-428d-11db-a30b-47fc0b00a30a
```

password

Sets the password used to authenticate with the remote UDDI registry.

Syntax

password *password*

Parameters

password

Specifies the password sent to the remote UDDI registry to authenticate the appliance with the registry. This authentication is then used to retrieve the subscription data.

Related Commands

username

registry

Determines the remote UDDI registry that holds the subscriptions.

Syntax

registry *name*

Parameters

name Specifies the name of an existing UDDI registry object.

Related Commands

uddi-registry (Global)

username

Sets the username to authenticate with the remote UDDI registry.

Syntax

username *username*

Parameters

username
Specifies the username sent to the remote UDDI registry to authenticate the appliance with the registry. This authentication retrieves the subscription data.

Related Commands

password

Chapter 93. URL Map configuration mode

This chapter provides an alphabetic listing of commands that are available in URL Map configuration mode.

To enter this configuration mode, use the Global **urlmap** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in URL Map configuration mode.

match

Adds a match pattern to the current URL map.

Syntax

match *match*

no match

Parameters

match Defines a shell-style match pattern that defines a set of URLs.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Guidelines

URL maps are used in the implementation of Stylesheet Refresh Policies and Compile Options (Profiling) policies.

Use the **no match** command to reset the URL map, that is remove all match patterns from the map.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details about the creation and implementation of URL maps and Stylesheet-Refresh Policies.

Refer to Appendix D, "Compile Options Policy configuration," on page 1115 for procedural details about the creation and implementation of URL maps and Compile Options Policies.

Related Commands

disable cache, **disable flush**, **interval urlmap**, **test urlmap**, **test urlrefresh**, **urlmap**, **urlrefresh**, **xslrefresh**

Examples

- Creates the URLmap-1 URL Map. Adds the match pattern https://www.amajoraccount.com/Zeus/*xsl to the map.

```
# urlmap URLmap-1
URL Map configuration mode
# match https://www.amajoraccount.com/Zeus/*xsl
#
```

- Creates the URLmap-2 URL Map. Adds two match patterns to the map.

```
# urlmap URLmap-2
URL Map configuration mode
# match https://www.company.com/XML/stylesheets/*
# match https://www.distributer.com/*xsl
#
```

- Removes all match patterns from the URLmap-2 URL Map.

```
# urlmap URLmap-2
URL Map configuration mode
# match https://www.company.com/XML/stylesheets/*
# match https://www.distributer.com/*xsl
# no match
#
```

Chapter 94. URL Refresh Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in URL Refresh Policy configuration mode.

To enter this configuration mode, use the Global **urlrefresh** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in URL Refresh Policy configuration mode.

disable cache

Defines a policy for style sheets in the URL Map are not cached.

Syntax

disable cache *map*

Parameters

map Specifies the name of a URL map.

Guidelines

Use the **disable cache** command to identify style sheets that are frequently updated. Disabling caching for such style sheets ensures that the most recent version of the style sheet is obtained by the XML Manager and used for filtering or transformation.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of Stylesheet Refresh Policies.

Related Commands

urlmap

disable flush

Defines a policy for style sheets in the URL Map are preferentially cached.

Syntax

disable flush *map interval*

Parameters

map Specifies the name of a URL map.

interval
Specifies the frequency, in seconds, at which style sheets obtained via the URL Map are refreshed.

Guidelines

Use the **disable flush** command to identify style sheets that should be preferentially cached. These style sheets remain in the cache for the full duration of the refresh cycle. This command overrides the setting in the XML Manager for caching rules for a particular URL that matches the URL Map.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of Stylesheet Refresh Policies.

Related Commands

urlmap

---

interval urlmap

Defines a policy for style sheets in the URL Map are cached.

Syntax

interval urlmap *map interval*

Parameters

map Specifies the name of a URL map.

interval
Specifies the frequency, in seconds, at which style sheets obtained via the URL Map are refreshed.

Guidelines

The **interval urlmap** command assigns a URL map to a URL Refresh Policy. Candidate style sheets that match the rules in this URL map are cached by an XML Manager.

Cached style sheets cached do not receive preferential treatment within the stylesheet cache. They can be deleted from the cache before their scheduled refresh (for example, if the cache exceeds its maximum size).

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of Stylesheet Refresh Policies.

Related Commands

urlmap

Examples

- Adds the URL maps `URLmap-3` and `URLmap-4` to the `2aday` URL Refresh Policy. Matching rules in either URL map are cached by an XML manager for 12 hours (43200 seconds). Returns to Global configuration mode.

```
# urlrefresh 2aday
URL Refresh Policy configuration mode
# interval urlmap URLmap-3 43200
# interval urlmap URLmap-4 43200
# exit
#
```

protocol-specified

Defines a policy in which style sheets are cached on protocol semantics.

Syntax

protocol-specified *map interval*

Parameters

map Specifies the name of a URL map.

interval
Specifies the frequency, in seconds, at which style sheets obtained via the URL Map are refreshed.

Guidelines

Use the **protocol-specified** command to indicate that style sheets should be cached in accordance with the expiration semantics that are supplied by protocols.

Refer to Appendix C, "Stylesheet Refresh Policy configuration," on page 1113 for procedural details regarding the creation and implementation of Stylesheet Refresh Policies.

Related Commands

urlmap

Chapter 95. URL Rewrite Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in URL Rewrite Policy configuration mode.

To enter this configuration mode, use the Global **urlrewrite** command. While in this configuration, define rewrite rules that perform the following types of replacements:

- Rewrite an entire URL or a portion of a URL based on a URL match.
- Replace the value of the `Content-Type` header based on a URL match.
- Replace the value of an arbitrary header based on its value.
- Rewrite the body of an HTTP POST request.

Rewrite rules that are defined in the URL Rewrite Policy occur before document processing.

- Any Matching Rule must match the rewritten URL.
- Any action in the Processing Policy can change the URI that is sent to the backend server.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in URL Rewrite Policy configuration mode.

absolute-rewrite

Rewrites the entire URL or a portion of the URL based on a URL match.

Syntax

absolute-rewrite *expression input-replace style-replace* [*input-unescape*] [*style-unescape*] [*normalize*]

Parameters

expression

Specifies a PCRE that defines the match condition that triggers the rewrite rule. A candidate URL that matches this PCRE triggers the rule. For examples:

`.*` or `*`

Matches any string.

`(.*)xsl=(.*)\?(.*)`

Matches a string of the following format:

1. A text subpattern.
2. Followed by `xsl=`.
3. Followed by a text subpattern.
4. Followed by `?`. The backward slash (\) in the PCRE is a URL escape.
5. Followed by a text subpattern.

`(.*)&[Xx][Ss][Ll]=([^&]+)(.*)`

Matches a string of the following format:

1. A text subpattern.
2. Followed by `&`.
3. Followed by `X` or `x`.
4. Followed by `S` or `s`.
5. Followed by `L` or `l`.
6. Followed by `=`.
7. Followed by a text subpattern that does not contain an ampersand (`&`) character.
8. Followed by a text subpattern.

input-replace

Specifies a PCRE that defines the rewritten URL that is passed to a backend server. For example:

- If the match pattern is `.*` or `*`, specify the complete replacement.
- If the match pattern is `(.*)xsl=(.*)\?(.*)`, specify the PCRE evaluation replacement for any text subpattern or retain the original text subpattern. To retain the first text subpattern, specify `&1`; to retain the second text subpattern, specify `&2`, and so forth. To replace the second text subpattern only, specify `$1xsl=ident.xsl?&3`.

If a rewritten URL begins with a host name or port that is different from the **remote-address** that is configured for the DataPower service, the host name or port portion of the rewritten URL is ignored.

style-replace

Specifies a PCRE that identifies the replacement style sheet. This style sheet filters or transforms the XML document that is referenced by the rewritten URL. For example:

- If the match pattern is `.*` or `*`, specify the complete replacement.
- If the match pattern is `(.*)xsl=(.*)\?(.*)`, specify the PCRE evaluation replacement for any text subpattern or retain the original text subpattern. To retain the first text subpattern, specify `&1`; to retain the second text subpattern, specify `&2`, and so forth. To retain the second text subpattern only and not use the third text subpattern, specify `http://mantis:8000/$2`.

input-unescape

Specifies whether to replace URL-encoded characters (for example, `%2F`) in the rewritten URL with literal character equivalents.

true Enables substitution.

false (Default) Disables substitution.

style-unescape

Specifies whether to replace URL-encoded characters (for example, `%2F`) in the replacement URL with literal character equivalents.

true (Default) Enables substitution.

false Disables substitution.

normalize

Specifies whether URL strings are normalized. Normalizing a URL compresses `'.'` and `'..'` and converts backward slashes (\) to forward slashes (/).

true (Default) Enables normalization.

false Disables normalization.

Guidelines

The **absolute-rewrite** command creates a rewrite rule that rewrites the entire URL based on a URL match and adds the URL rewrite rule to the current URL Rewrite Policy. This rewrite rule operates on an entire URL.

The decoding (unescape) process replaces URL escape sequences with character equivalents. For example, `/image%20library` is decoded as `/image library`.

PCRE documentation is available at the http://www.pcre.org web site.

Examples

- Adds an absolute rewrite rule to the current URL Rewrite Policy. If the candidate URL is `http://mantis:8000/foo/bar/my.cgi?x=y&xsl=style.xsl?/input.xml`, the rule rewrites the input URL to `http://mantis:8000/foo/bar/my.cgi?x=y&xsl=ident.xsl?/input.xml` and rewrites the URL to `http://mantis:8000/style.xsl`.

```
# absolute-rewrite (.*)xsl=(.*)\?(.*) $1xsl=ident.xsl?$3
  http://mantis:8000/$2
#
```

content-type

Rewrites the contents of the `Content-Type` header based on a URL match.

Syntax

content-type *expression input-replace* [*normalize*]

Parameters

expression

Specifies a PCRE that defines the match condition that triggers the rewrite rule. A candidate URL that matches this PCRE triggers the rule. For examples:

`.*` or `*`

Matches any string.

`(.*)xsl=(.*)\?(.*)`

Matches a string of the following format:

1. A text subpattern.
2. Followed by `xsl=`.
3. Followed by a text subpattern.
4. Followed by `?`. The backward slash (\) in the PCRE is a URL escape.
5. Followed by a text subpattern.

`(.*)&[Xx][Ss][Ll]=([^&]+)(.*)`

Matches a string of the following format:

1. A text subpattern.
2. Followed by `&`.
3. Followed by `X` or `x`.
4. Followed by `S` or `s`.
5. Followed by `L` or `l`.
6. Followed by `=`.

7. Followed by a text subpattern that does not contain an ampersand (&) character.
8. Followed by a text subpattern.

input-replace
Specifies the replacement value for the `Content-Type` header.

normalize
Specifies whether URL strings are normalized. Normalizing a URL compresses `'.'` and `'..'` and converts backward slashes (\) to forward slashes (/).

true (Default) Enables normalization.

false Disables normalization.

Guidelines

The **content-type** command creates a rewrite rule that rewrites the contents of the `Content-Type` header based on a URL match and adds the URL rewrite rule to the current URL Rewrite Policy.

Backend servers often identify the character set and document type of documents they send in the `Content-Type` header that is prefixed to the document.

PCRE documentation is available at the http://www.pcre.org web site.

Examples

- Adds a `Content-Type` header rewrite rule to a URL Rewrite Policy. If the candidate URL contain `/web/Server`, the rule replaces the value of the `Content-Type` header with `text/xml`.

```
# content-type .*/web/Server.* text/xml
#
```

header-rewrite

Rewrites the contents of an arbitrary header based on a name match.

Syntax

header-rewrite *name expression input-replace* [*normalize*]

Parameters

name Identifies the name of the header to have its value rewritten. The header name must be entered exactly as it is defined in the message.

expression
Specifies a PCRE that defines the match condition that triggers the rewrite rule. A candidate value of the specified header that matches this PCRE triggers the rule. For example `*.*` matches any value.

input-replace
Specifies the replacement value for the specified header.

normalize
Specifies whether URL strings are normalized. Normalizing a URL compresses `'.'` and `'..'` and converts backward slashes (\) to forward slashes (/).

true (Default) Enables normalization.

false Disables normalization.

Guidelines

Use the **header-rewrite** command to replace the contents of an arbitrary header.

PCRE documentation is available at the http://www.pcre.org web site.

Related Commands

Examples

- Adds a header rewrite rule to a URL Rewrite Policy. If the message contains the `Age` header, the rule replaces its value with 1.

```
# header-rewrite Age *.* 1
#
```

no rule

Deletes all rewrite rules from the current URL Rewrite Policy.

Syntax

no rule

Guidelines

The **no rule** command deletes all rules from the current policy.

Examples

- Deletes all rules from the current policy.

```
# no rule
#
```

post-body

Rewrites the contents of the HTTP POST body based on a URL match.

Syntax

post-body ***.*** *expression input-replace style-replace* [*input-unescape*] [*style-unescape*] [*normalize*]

Parameters

. Indicates a required argument for backward compatibility.

expression
Specifies a PCRE that defines the match condition that triggers the rewrite rule. A candidate URL that matches this PCRE triggers the rule. For examples:

`.*` or `*`
Matches any string.

`(.*)xsl=(.*)\?(.*)`
Matches a string of the following format:

1. A text subpattern.
2. Followed by `xsl=`.
3. Followed by a text subpattern.
4. Followed by `?`. The backward slash (\) in the PCRE is a URL escape.
5. Followed by a text subpattern.

`(.*)&[Xx][Ss][Ll]=([^&]+)(.*)`

Matches a string of the following format:

1. A text subpattern.
2. Followed by `&`.
3. Followed by `X` or `x`.
4. Followed by `S` or `s`.
5. Followed by `L` or `l`.
6. Followed by `=`.
7. Followed by a text subpattern that does not contain an ampersand (`&`) character.
8. Followed by a text subpattern.

input-replace

Specifies a PCRE that defines the rewritten POST that is passed to the backend server. For example:

- If the match pattern is `.*` or `*`, specify the complete replacement.
- If the match pattern is `(.*)xsl=(.*)\?(.*)`, specify the PCRE evaluation replacement for any text subpattern or retain the original text subpattern. To retain the first text subpattern, specify `&1`; to retain the second text subpattern, specify `&2`, and so forth. To omit the second text subpattern only, specify `$1&3`.

style-replace

Specifies a PCRE that identifies the replacement style sheet. This style sheet filters or transforms the XML document that is referenced by the rewritten POST. For example:

- If the match pattern is `.*` or `*`, specify the complete replacement.
- If the match pattern is `(.*)xsl=(.*)\?(.*)`, specify the PCRE evaluation replacement for any text subpattern or retain the original text subpattern. To retain the first text subpattern, specify `&1`; to retain the second text subpattern, specify `&2`, and so forth. To retain the second text subpattern only and not use the first or third text subpattern, specify `http://10.10.10.200:909/$2`.

input-unescape

Specifies whether to replace URL-encoded characters (for example, `%2F`) in the rewritten URL with literal character equivalents.

true Enables substitution.

false (Default) Disables substitution.

style-unescape

Specifies whether to replace URL-encoded characters (for example, `%2F`) in the replacement URL with literal character equivalents.

true (Default) Enables substitution.

false Disables substitution.

normalize

Specifies whether URL strings are normalized. Normalizing a URL compresses `'.'` and `'..'` and converts backward slashes (\) to forward slashes (/).

true (Default) Enables normalization.

false Disables normalization.

Guidelines

The decoding (unescape) process replaces URL escape sequences with character equivalents. For example, `/image%20library` is decoded as `/image library`.

PCRE documentation is available at the http://www.pcre.org web site.

Examples

- Adds a post-body rewrite rule to the current URL Rewrite Policy. If the candidate URL is `http://mantis:8000/foo/bar/my.cgi?x=y&xsl=style.xsl?/input.xml`, the rule rewrites the body of the HTTP POST to `http://mantis:8000/foo/bar/my.cgi?x=y/input.xml` and rewrites the URL to `http://10.10.10.200:909/&xsl=style.xsl`.

```
# post-body *.* (.*)&[Xx][Ss][Ll]=([^&]+)(.*) $1$3
  http://10.10.10.200:909/$2
#
```

rewrite (deprecated)

Comments

This command is deprecated.

Chapter 96. User Agent configuration mode

This chapter provides an alphabetic listing of commands that are available in User Agent configuration mode.

To enter this configuration mode, use the Global **user-agent** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in User Agent configuration mode.

add-header-policy

Creates a Header Injection Policy.

Syntax

add-header-policy *pattern field value*

no add-header-policy *pattern*

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this Header Injection Policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

field Identifies a proprietary HTTP header field.

value Specifies the field value. Can contain a character string or an integer.

Guidelines

A Header Injection Policy defines a URL set and performs HTTP header injection on a specified set of URLs. The Header Injection Policy specifies an HTTP header field and value contained in that field.

URLs processed by the User Agent are evaluated against the defined URL set. Matching URLs are altered as need be to include the associated HTTP header field and value.

Use the **no add-header-policy** command to remove the Header Injection Policy.

Examples

- Injects the `ProcInt` HTTP header field that contains a value of 0 into all URLs matching the `*datapower.com*` match expression.

```
# add-header-policy *datapower.com* ProcInst 0
#
```

- Removes the Header Injection Policy.

```
# no add-header-policy *datapower.com*
#
```

---

basicauth

Creates a basic authentication policy.

Syntax

basicauth *pattern user password*

no basicauth-policy *pattern*

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this basic authentication policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

user Specifies the user name used to access the URL set.

password
Specifies the user-password used to access the URL set.

Guidelines

A Basic Authentication Policy defines a URL set and uses a simple user name and password to manage access to requested resources.

Use the **no basicauth-policy** command to remove the Basic Authentication Policy.

Examples

- Adds a Basic Authentication Policy, which uses the user name, `customer`, and the password, `query`, to access the URL set defined by `*datapower.com*`.

```
# basicauth-policy *datapower.com* customer query
#
```

- Removes the Basic Authentication Policy.

```
# no basicauth-policy *datapower.com*
#
```

chunked-uploads-policy

Creates a chunked uploads policy.

Syntax

chunked-uploads *pattern* {**on** | **off**}

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this chunked uploads policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

on Enables chunked encoding.

off (default) Disables chunked encoding. Alternatively, use the **no chunked-uploads-policy** command.

Guidelines

A Chunked Uploads Policy defines a URL set and enables or disables chunked encoding for that group of URLs.

The User Agent may send an HTTP 1.1 request to the back end server. In this case, the body of the document can be delimited by either Content-Length or chunked encoding. All servers will understand how to interpret Content-Length, and many applications will fail to understand chunked, so Content-Length is generally used. However doing so interferes with the ability of the appliance to fully stream. If you must stream full documents towards the back side, this property should be turned on. However, you must know beforehand that the server with which you are communicating is RFC 2616 compatible, unlike all other HTTP 1.1 features which can be negotiated down at runtime if necessary. You might also consider leaving this property turned off and turning it on a per-URL basis with the User Agent configuration.

Examples

- Adds a Chunked Uploads Policy, which enables HTTP 1.1 chunked encoding on the URL set defined by `*datapower.com*`; subsequently disables chunked encoding support, thus restoring the default state.

```
# chunked-uploads-policy *datapower.com* on
⋮
# chunked-uploads-policy *datapower.com* off
#
```

compression-policy

Creates a compression policy.

Syntax

compression-policy *pattern* {**on** | **off**}

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this compression policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

on Enables compression negotiation.

off (Default) Disables compression negotiation. Alternatively, use the **no compression-policy** command.

Guidelines

A Compression Policy enables or disables GZIP compression negotiation for that group of URLs between the User Agent and the backend server.

Examples

- Adds a Compression Policy, which enables compression negotiation on the URL set defined by `*datapower.com*`. Subsequently disables compression negotiation, thus restoring the default state.

```
# compression-policy *datapower.com* on
:
# compression-policy *datapower.com* off
#
```

ftp-policy

Creates a policy that associates a URL set with a set of default policies for the FTP protocol.

Syntax

ftp-policy *pattern* [**pasv-off** | **pasv-opt** | **pasv-req**] [**auth-off** | **auth-tls-opt** | **auth-tls-req**] [**ccc-off** | **ccc-opt** | **ccc-req**] [**enc-data-off** | **enc-data-opt** | **enc-data-req**] [**ascii** | **binary**] [**slash-stou-off** | **slash-stou-on**] *name* [**size-check-optional** | **size-check-disabled**]

Parameters

pattern Specifies a shell-style regular expression that defines the URL set that is subject to the FTP policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

pasv-off | pasv-opt | pasv-req
Indicates how to use passive mode with the FTP **PASV** command.

pasv-off
Do not request passive mode.

pasv-opt
Request, but to not require, passive mode.

pasv-req
Request and require passive mode.

auth-off | auth-tls-opt | auth-tls-req
Indicates how to use authentication and encryption of the command channel with the FTP **AUTH TLS** command.

auth-off
Do not request authentication and encryption.

auth-tls-opt
Request, but to not require, authentication and encryption.

auth-tls-req
Request and require authentication and encryption. Failure to negotiate TLS results in a failure of the related transaction.

ccc-off | ccc-opt | ccc-req
Indicates how to use command channel encryption after user authentication with the FTP **CCC** command.

ccc-off Do not request command channel encryption.

ccc-opt
Request, but do not require, command channel encryption.

ccc-req
Request and require command channel encryption. Use this option when the command connection crosses a NAT or firewall that needs to monitor commands.

enc-data-off | enc-data-opt | enc-data-req
Indicates how to use data connection encryption with the FTP **PROT P** command.

enc-data-off
Do not request data connection encryption.

enc-data-opt
Request, but do not require, data connection encryption.

enc-data-req
Request and require data connection encryption. Failure of the FTP server to encrypt the data connection results in a failure of the related transaction.

ascii | binary
Indicates how to transfer data.

ascii Transfers data in ASCII mode using the FTP **TYPE A** command. Use caution when transferring XML documents in this mode. Many XML documents are sensitive to the exact end-of-line convention.

binary Transfers data in image mode using the FTP **TYPE I** command.

slash-stou-off | **slash-stou-on**
Indicates how to use server-generated unique file names when the URL being written to ends in a slash (/).

slash-stou-off
Do not request unique file names when the URL being written to ends in a slash. Because the URL ends in a slash, the requested file name will be empty. Generally, this condition results in a failure.

slash-stou-on
Request server-generated unique file names. When storing the file, uses the FTP **STOU** command instead of the FTP **STOR** command.

name Enter the name of an FTP quoted command list object. This property is required. If you do not need this property, enter "".

size-check-optional | **size-check-disabled**
Indicates how to use a size check after a data transfer with the FTP **SIZE** command.

size-check-optional
If the command is available, use it to check the size of the file after transfer. The command compares the returned value to the number of bytes that were transferred. If not equal, the transfer is marked as failing.

size-check-disabled
Do not perform this check.

Guidelines

Creates a policy that associates a URL set that is defined by a match pattern with a set of default policies for the FTP protocol. These default policies are used by the FTP user agent (the FTP client). Most of these default policies can be overridden by query parameters in the URL.

Related Commands

ftp-quoted-command-list (Global)

---

identifier

Specifies the string to be included in the User Agent request-header field.

Syntax

identifier *string*

Parameters

string Specifies a text string that contains the contents of the User Agent request-header field.

Guidelines

The User Agent request header field contains information about the User Agent initiating the request, that is the appliance. By default the appliance does not include a User Agent request-header field.

max-redirects

Specifies the maximum number of HTTP redirect messages.

Syntax

max-redirects *messages*

Parameters

messages
Specifies the maximum number of redirect messages. Use an integer in the range of 0 to 128. The Default is 8.

Guidelines

The **max-redirects** command specifies the maximum number of HTTP redirect messages to receive before the appliance declares the URL unreachable.

proxy

Creates a proxy policy.

Syntax

proxy *pattern server port*

proxy *pattern* **none**

no proxy [*pattern*]

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this proxy policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

server Specifies the name or IP address of an HTTP server. With the port, designates the HTTP proxy that services the URL set defined by the match pattern.

port Specifies a port on the HTTP server. With the server name or IP address, designates the HTTP proxy that services the URL set defined by the match pattern.

none Specifies that the URL set that is defined by the match pattern is not forwarded to an HTTP proxy.

Guidelines

A proxy policy associates a URL set with a specific HTTP proxy. You can create multiple proxy policies. In this case, candidate URLs are evaluated against each policy in turn. Consequently, policy ordering is important.

Use the **no proxy** command with the match expression to remove that specific match. To remove all match patterns, use the **no proxy** command without arguments

Examples

- Creates two proxy policies. URLs matching the `http://*.internal.datapower.com` pattern are directed to port 8080 on `backoffice`. All other URLs are directed to port 8080 on `internet-gateway`.

```
# proxy http://*.internal.datapower.com backoffice 8080
# proxy * internet-gateway 8080
#
```

- Creates two proxy policies. All URLs are directed to port 8080 on `internet-gateway`. The second policy will never be reached since all candidate URLs match the first policy.

```
# proxy * internet-gateway 8080
# proxy http://*.internal.datapower.com backoffice 8080
#
```

- Creates three proxy policies. URLs matching the `http://*.internal.datapower.com` pattern are directed to port 8080 on `backoffice`. URLs matching the `http://*.finance.datapower.com` pattern are dropped. All other URLs are directed to port 8080 on `internet-gateway`.

```
# proxy http://*.internal.datapower.com backoffice 8080
# proxy http://*.finance.datapower.com none
# proxy * internet-gateway 8080
#
```

- Deletes the specified proxy policy.

```
# no proxy http://*.internal.datapower.com backoffice 8080
#
```

- Deletes all proxy policies.

```
# no proxy
#
```

pubkeyauth

Creates a public key authentication policy.

Syntax

pubkeyauth *pattern key*

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this authentication policy.

You can use wildcards to define a match pattern as follows:

* The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

key Specifies the Crypto Key object used in the authentication process. This key must reside on the appliance.

Guidelines

A public key authentication policy defines a URL set and specifies a private key required to access the URL set.

Public key authentication uses the public-private key pair as follows:

1. A public-private key pair is generated.
2. The public portion is distributed to target servers you wish to authenticate with.
3. At login, the server returns a challenge message encrypted with your public key.
4. You authenticate by decrypting with your private key and returning the plaintext message to the server.

Use the **no pubkeyauth** command to remove the authentication policy.

Examples

- Adds a Public Key Authentication Policy, which uses the private key, `bob`, to access the URL set defined by `*datapower.com*`.

```
# pubkeyauth *datapower.com* bob
#
```

- Removes the Public Key Authentication Policy.

```
# no pubkeyauth *datapower.com*
#
```

restrict-http-policy

Creates an HTTP version restriction policy.

Syntax

restrict-http-policy-policy *pattern* {**on** | **off** }

no restrict-http-policy-policy *pattern*

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this HTTP version restriction policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

on Enables version restrictions.

off Disables version restrictions. Alternatively, use the **no restrict-http-policy-policy** command.

Guidelines

An HTTP version restriction policy limits access to a specified URL set to HTTP Version 1.0.

Examples

- Adds an HTTP version restriction policy, which requires HTTP 1.0 access on the URL set defined by `*datapower.com*`; subsequently disables version restriction, thus enabling access by HTTP 1.0 and HTTP 1.0.

```
# restrict-http-policy *datapower.com* on
⋮
# restrict-http-policy *datapower.com* off
#
```

soapaction

Creates a `SOAPAction` header injection policy.

Syntax

soap-action *pattern value*

no soap-action *pattern*

Parameters

pattern Specifies a shell-style match pattern that defines the URL set subject to this `SOAPAction` header injection policy.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

value Specifies the value of the `SOAPAction` header.

Guidelines

A `SOAPAction` header injection policy adds a `SOAPAction` header and header value to a specified set of URLs.

URLs processed by the User Agent are evaluated against the defined URL set; matching URLs are altered as needed to include the associated `SOAPAction` header field and value.

Use the **no soap-action** command to remove the SOAPAction header injection policy.

Examples

- Injects the SOAPAction header field that contains a value of http://example.org/add into all URLs matching the *datapower.com* match expression.

```
# soap-action *datapower.com* http://example.org/add
#
```

- Removes the SOAPAction Header Injection Policy.

```
# no soap-action *datapower.com*
#
```

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *pattern name*

no ssl

Parameters

pattern Specifies a shell-style regular expression that defines the URL set supported by the SSL Proxy Profile.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

[1-5] Matches 1, 2, 3, 4, or 5

[xy] Matches x or y

name Identifies the SSL Proxy Profile assigned the User Agent.

Guidelines

An SSL Proxy Profile specifies the SSL operational mode (client) and identifies the cryptographic resources (key, certificates, and cipher lists) available to the User Agent.

The SSL Proxy Profile (either client or two-way) must already be created with the **sslproxy** command.

Use the **no ssl** command to remove the SSL Proxy Profile.

Related Commands

sslproxy

Examples

- Creates an SSL policy for use by the current User Agent. When fetching a URL conforming to the specified match pattern, the Use Agent uses the SSL-UA1 SSL profile.

```
# ssl https://*/testbase/* SSL-UA1
#
```

timeout

Specifies the User Agent idle timeout value.

Syntax

timeout *time*

Parameters

time Specifies the idle timeout. Use an integer in the range of 0 to 86400. The default is 300.

Guidelines

The timeout is the maximum idle period before an established connection is torn down.

Chapter 97. User configuration mode

This chapter provides an alphabetic listing of commands that are available in User configuration mode.

To enter this configuration mode, use the Global **user** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in User configuration mode.

access-level

Assigns an account type of an account.

Syntax

access-level {**privileged** | **user**}

Parameters

privileged
Assigns executive access to the account. A privileged account has virtually the same access levels as the `admin` account. It differs only in that a privileged account cannot delete the `admin`.

user Assigns restricted access to the account. A user account is limited to the common commands and most, but not all of the **show** commands.

Guidelines

By default, newly created accounts are assigned the `user` access level.

Related Commands

group, **password**

domain

Restricts access to specific application domains.

Syntax

domain [*name*]

Parameters

name Specifies the name of an existing Application Domain object.

Guidelines

The **domain** command restricts access to the list of specific domains. Without at least one application domain, the user can log in to any domain on the appliance. This command is valid in the following situations only:

- The user is not a member of a user group.

- The user is a member of a user group, but the user group does not define access policies.

In these cases, the **domain** command defines access through all interfaces (WebGUI, command line, XML Management interface).

With access policies in the user group, the **domain** command can limit access to specific application domains from the command line only, not the WebGUI or XML Management interface.

- To add the user as a member of a user group, use the **group** command.
- To apply access across all interfaces, enforce the RBM policy on the command line with the RBM Settings **apply-cli** command.

Related Commands

apply-cli (RBM Setting), **group**

Examples

- Limits access for the `gharrison` User object to the `engineering` domain.

```
# user gharrison
Modify User configuration
# domain engineering
#
```

---

group

Associates an account with a group.

Syntax

group *name*

Parameters

name Specifies the name of an existing User Group object.

Guidelines

Refer to the **name** command for information on creating User Groups.

Related Commands

access-level, **password**, **name**

---

password

Assigns a password to a new account, or changes the password of an existing account.

Syntax

password *password*

Parameters

password

Specifies the password for the account. A password can contain only printable characters and must be 5 to 20 characters in length.

Guidelines

You must assign a password to a newly created account.

Related Commands

access-level, **group**

---

snmp-cred

Adds SNMP V3 credentials to this account.

Syntax

snmp-cred *engine-ID authentication-protocol authentication-secret-type authentication-secret privacy-protocol privacy-secret-type privacy-secret*

Parameters

engine-ID

Specifies the engine ID of the SNMP V3 engine for which this account is being defined. A value of `0` is the shorthand representation of the engine ID of the local SNMP V3 engine on the DataPower appliance. For any other engine ID, the value is a hexadecimal string that represents the five- to 32-byte value.

authentication-protocol

Identifies which authentication protocol to use.

none The account has no authentication key.

md5 The account uses HMAC-MD5-96 as the authentication protocol.

sha (Default) The account uses HMAC-SHA-96 as the authentication protocol.

authentication-secret-type

Indicates whether the authentication secret is a password or a fully localized key. This parameter is required when the value for *authentication-protocol* is **md5** or **sha**.

password
(Default) The authentication secret is a password that will be converted to an intermediate key with a standardized algorithm, and then localized against the engine ID value.

key The authentication secret is a fully localized key. Specifying a fully localized key is useful when the key was initially created on another system.

authentication-secret

Specifies the secret, or key, for authentication for this account. This parameter is required when the value for *authentication-protocol* is **md5** or **sha**.

- If a password, specify a plaintext password that is at least eight characters long.
- If a key and HMAC-MD5 is the authentication protocol, specify the hexadecimal representation of a 16-byte key.
- If a key and HMAC-SHA-96 is the authentication protocol, specify the hexadecimal representation of a 20-byte key.

You can use colons (:) between each two hexadecimal characters.

privacy-protocol
Identifies which privacy (encryption) protocol to use.

none The account has no privacy key.

des (Default) The account uses CBC-DES as the privacy protocol.

aes The account uses CFB128-AES-128 as the privacy protocol.

privacy-secret-type
Indicates whether the privacy secret is a password or a fully localized key. This parameter is required when the value for *privacy-protocol* is **des** or **aes**.

password
The privacy secret is a password that will be converted to an intermediate key with a standardized algorithm, and then localized against the engine ID value.

key The privacy secret is a fully localized key. Specifying a fully localized key is useful when the key was initially created on another system.

privacy-secret
Specifies the secret, or key, for privacy (encryption) for this account. This parameter is required when the value for *privacy-protocol* is **des** or **aes**.

- If a password, specify a plaintext password that is at least eight characters long.
- If a key and HMAC-MD5 is the authentication protocol, specify the hexadecimal representation of a 16-byte key.
- If a key and HMAC-SHA-96 is the authentication protocol, specify the hexadecimal representation of a 20-byte key.

You can use colons (:) between each two hexadecimal characters.

Guidelines

The **snmp-cred** command adds SNMP V3 credentials for this account. Each account can have multiple SNMP V3 credentials, one for each SNMP V3 engine that is identified by an *engine-ID* value.

Note: The current implementation supports an SNMP V3 credential for the local engine ID only. Therefore, there can be only one SNMP V3 credential for each account.

The secret for authentication and for privacy can be defined either as a password (passphrase), which will be hashed and localized with the engine ID or can be defined as a localized hexadecimal key.

Examples

- Creates SNMP V3 credentials for this account on the appliance with HMAC-MD5-96 as the authentication algorithm, and DES-CBC as the privacy algorithm. The password `aBigSecret` will be converted to a localized authentication key, and the password `aDifferentSecret` will be converted to a localized encryption key.

```
snmp-cred 0 md5 password aBigSecret des password aDifferentSecret
```

- Creates SNMP V3 credentials for this account on the remote machine with the engine ID `000000000000000000000002`, with HMAC-MD5-96 as the authentication

algorithm, and with no privacy algorithm. The password is `maplesyrup`, which will be converted to a localized key for the specified engine ID (`000000000000000000000002`).

```
snmp-cred 000000000000000000000002 md5 password maplesyrup none password ""
```

- Creates SNMP V3 credentials for this account on the remote machine with the engine ID `000000000000000000000002`, with HMAC-MD5-96 as the authentication algorithm, and with no privacy algorithm. The fully localized key is `52:6f:5e:ed:9f:cc:e2:6f:89:64:c2:93:07:87:d8:2b`.

```
snmp-cred 000000000000000000000002 md5 key
 52:6f:5e:ed:9f:cc:e2:6f:89:64:c2:93:07:87:d8:2b none password ""
```

Chapter 98. User Group configuration mode

This chapter provides an alphabetic listing of commands that are available in User Group configuration mode.

To enter this configuration mode, use the Global **usergroup** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in User Group configuration mode.

access-policy

Assigns an access policy.

Syntax

access-policy *statement*

Parameters

statement
Specifies the policy statement to add. A policy statement takes the following form:

address`/`*domain*`/`*resource*`? [Name=`*name*`]&Access=`*permission*

address
An IP address. This policy will apply only to clients with IP addresses that match this value.

PCRE expressions are supported.

The special value * matches any IP address.

domain
The name of an application domain. This policy will apply only to resources within the identified domain.

The special value * matches any domain.

resource
The resource type to which this policy applies.

The special value * matches any resource type.

`Name=`*name*
Optionally identifies by name a particular instance of the specified resource type. PCRE expressions can be used (for example `foo*`).

`Access=`*permission*
The permission string assigns permissions. The string is cumulative and connected by plus (+) signs. For example `r+w+a+d`

r Read

w Write (modify)

a Add new object

d Delete existing object

x Execute

Guidelines

The **access-policy** command assigns one or more access policy statements to the user group. If there are more than one statement, the statements are cumulative. If more than one statement applies to the same resource, the most specific statement will apply. For example, given the following two statements any member of this user group can read all objects but has complete access privileges to Multi-Protocol Gateway services:

```
*/*/*?Access=r
*/*/services/multiprotocol-gateway?Access=r+w+a+d+x
```

It is not possible to remove a specific access policy from the command line. If you invoke the **no access-policy** command, all access policies are removed. To remove a specific access policy from a user group, use the WebGUI.

Examples

- Adds full access privileges to all resources and read only access for login and network resources to members of the `appdev` User Group.

```
# usergroup appdev
User group configuration mode
# access-policy "*/*/*?Access=r+w+a+d"
# access-policy "*/*/login/*?Access=r"
# access-policy "*/*/network/*?Access=r"
# exit
Usergroup update successful
#
```

---

add

Adds a command suite.

Syntax

add *name*

Parameters

name Specifies the name of the command group.

Guidelines

To display a list of available command groups, use the **add** command, with no arguments, at the CLI prompt.

Edit of an existing User Group with either the **add** or **delete** commands does not effect the access privileges of current user accounts assigned to the User Group. The updated access privileges defined by the edited User Group are assigned to user accounts subsequently added to the User Group.

To assign an edited User Group to user account, you must first delete the account with the **no username** command and then recreate the account with the **username** command.

Related Commands

delete

Examples

- Adds access to configuration mode, URL Map Mode, URL Refresh Mode, URL Rewrite Policy configuration mode, Matching Rule configuration mode, Stylesheet Policy configuration mode, and XSL Proxy configuration mode to members of the `stylesheet` User Group.

```
# usergroup stylesheets
User group configuration mode
# add configuration
# add urlmap
# add urlrefresh
# add matching
# add stylesheetpolicy
# add xslproxy
# exit
Usergroup update successful
#
```

---

delete

Removes a command suite.

Syntax

delete *name*

Parameters

name Specifies the name of the command group.

Guidelines

To display a list of available command groups, enter the **delete** command, with no arguments, at the command prompt.

Edit of an existing User Group with either the **add** or **delete** commands does not effect the access privileges of current user accounts assigned to the User Group. The updated access privileges defined by the edited User Group are assigned to user accounts subsequently added to the User Group.

To assign an edited User Group to user account, you must first delete the account with the **no username** command and then recreate the account with the **username** command.

Related Commands

add

Examples

- Removes access to configuration mode for the `stylesheets` User Group.

```
# usergroup stylesheets
User group configuration mode
# delete configuration
# exit
#
```

Chapter 99. VLAN configuration mode

This chapter provides an alphabetic listing of commands that are available in VLAN configuration mode.

To enter this configuration mode, use the global **vlan-sub-interface** command.

In this configuration mode, all of the commands that are in "Common commands" on page 2 and most, but not all, of the commands that are in Chapter 129, "Monitoring commands," on page 1053 are available.

arp

Enables or disables ARP.

Syntax

arp

no arp

Guidelines

The **arp** command enables or disables the Address Resolution Protocol (ARP) on all IP interfaces that are provided by the current VLAN port.

By default, ARP is enabled. Certain network topologies and load-balancing configurations might require that you disable ARP.

To disable, use the **no arp** command.

Related Commands

vlan-sub-interface, **show netarp**

Examples

- Disables ARP.

```
# no arp
#
```

- Enables ARP (restores default state).

```
# arp
#
```

dhcp

Enables or disables the DHCP client.

Syntax

dhcp

no dhcp

Guidelines

The **dhcp** command enables or disables the (Dynamic Host Configuration Protocol (DHCP) client. By default, DHCP is disabled.

When enabled, the DHCP client can obtain the following parameters from the DHCP server:
- Interface IP address
- Default Gateway IP address
- DNS IP address

To disable the DHCP client, use the **no dhcp** command.

Examples

- Enables the DHCP client.

```
# dhcp
#
```

- Disables the DHCP client (restores default state).

```
# no dhcp
#
```

identifier

Specifies the VLAN identifier to send and to receive traffic.

Syntax

identifier *identifier*

Parameters

identifier

Specifies the number of the VLAN identifier. Use an integer in the range of 1 through 4094. The default is 2.

Guidelines

The **identifier** command specifies the number of the VLAN identifier to send and to receive traffic. The identifier must be unique among all VLAN interfaces on the same Ethernet Interface. The VLAN identifier 1 is generally reserved by switches for non-tagged packets.

Examples

- Sets the VLAN Identifier to 42

```
# identifier 42
#
```

interface

Specifies the Ethernet interface to provide connectivity.

Syntax

interface {**eth0** | **eth1** | **eth2** | **mgt0**}

Guidelines

The **interface** command specifies the Ethernet interface that provides connectivity to the VLAN interface. Even if the Ethernet interface is not configured with an IP address, this command enables that Ethernet port.

Depending on model type, the appliance provides three or four Ethernet interfaces:

- A single dedicated management port (labelled either `MANAGEMENT` or `MGMT`)
- Two or three network ports (labelled either `ETHERNET` or `NETWORK`)

Use the **show interface** command to view the available Ethernet interfaces.

Use the **show vlan-interface** command to view the VLAN interfaces.

Related Commands

show interface

Examples

- Enable the VLAN interface on Ethernet 2.

```
# interface eth2
#
```

---

ip address

Assigns a primary network address.

Syntax

ip address *address*

no ip address

Parameters

address Specifies the network address (IP address and subnet). Specify the IP address in decimal format. Specify the subnet mask in CIDR format (`/27`) or its equivalent decimal format (`255.255.255.224`).

Guidelines

The **ip address** command assigns a primary network address (IP address and subnet mask) to the current VLAN interface and enables the interface.

To remove the primary network address, use the **no ip address** command.

Related Commands

ip secondary-address

Examples

- Assigns a primary network address to the current interface.

```
# ip address 192.168.7.6/27
#
```

- Removes the primary network address.

```
# no ip address
#
```

ip default-gateway

Specifies the default gateway.

Syntax

ip default-gateway *gateway*

no ip default-gateway

Parameters

gateway
Specifies the host name or IP address.

Guidelines

The **ip default-gateway** command specifies the default gateway that is reachable by the current interface. You can define the default gateway by IP address or host name. To use a host name, ensure that DNS services are enabled.

To delete the default gateway, use the **no ip default-gateway** command.

Examples

- Specifies `10.10.10.100` as the IP address of the default gateway.

```
# ip default-gateway 10.10.10.100
#
```

- Deletes the default gateway.

```
# no ip default-gateway
#
```

ip route

Adds a static route to the routing table.

Syntax

ip route *address gateway* [*metric*]

no ip route *address gateway*

Parameters

address Specifies the network address (IP address and subnet) of the target destination. Specify the IP address in decimal format. Specify the subnet mask in CIDR format (`/27`) or its equivalent decimal format (`255.255.255.224`).

gateway
Specifies the IP address of the network gateway (next-hop router).

metric Assigns a routing metric. The greater value indicates a more preferred route. Use an integer in the range of 0 to 255. The default is 0.

Guidelines

To delete a static route, use the **no ip route** command.

Examples

- Adds a static route with destination network 10.10.10.0, subnet mask /27 (equivalent to 255.255.255.224), and next-hop gateway 192.168.1.100 to the routing table.

```
# ip route 10.10.10.0/27 192.168.1.100
#
```

or

```
# ip route 10.10.10.0 255.255.255.224 192.168.1.100
#
```

- Deletes a static route with destination network 10.10.10.0 and subnet mask /27 from the routing table.

```
# no ip route 10.10.10.0/27 192.168.1.100
#
```

---

ip secondary-address

Adds or removes secondary network addresses.

Syntax

ip secondary-address *address*

no ip secondary-address [*address*]

Parameters

address Specifies the network address (IP address and subnet). Specify the IP address in decimal format. Specify the subnet mask in CIDR format (/27) or its equivalent decimal format (255.255.255.224).

Guidelines

The **ip secondary address** command adds a secondary network address (IP address and subnet mask) to the current VLAN interface. This address accepts incoming connections. This address is used only as a source IP address when responding to incoming requests (TCP or ICMP) to the secondary address.

To remove a secondary network address, use the **no ip address** command.

Related Commands

ip address

Examples

- Adds 192.168.7.6/27 as a secondary network address to the current interface.

```
# ip address 192.168.7.6/27
#
```

- Removes 192.168.7.6/27 as a secondary network address.

```
# no ip address 192.168.7.6/27
#
```

- Removes all secondary network addresses.

```
# no ip address
#
```

outbound-priority

Sets the priority of outbound packets.

Syntax

outbound-priority *priority*

Parameters

priority
Specifies the priority value. Use an integer in the range of 0 through 7. The default is 0.

Guidelines

The **outbound-priority** command sets the priority value to place in outgoing VLAN headers for packets that sent on this VLAN interface. This value is placed in the `user_control` field of the Tag Control Information (TCI).

The exact interpretation of the value depends on the VLAN switch configuration.

Examples

- Set the priority to 4

```
# outbound-priority 4
```

packet-capture

Initiates a packet-capture session on the current interface.

Syntax

Starts a package capture
packet-capture *filename duration kilobytes*

Immediately stops a package capture
no packet-capture *filename*

Parameters

filename
specifies the file to which packet-capture data is written.

duration
is an integer (within the range 5 through 3600) that specifies the maximum duration, in seconds, of the packet-capture session. A value of -1 indicates that the packet capture completes when the maximum file size is reached or until you invoke the **no packet-capture** command.

kilobytes
Specifies the maximum size, in kilobytes, of the packet-capture file. Use an integer in the range of 10 through 50000.

Guidelines

Packet-capture date is saved in a *pcap* format. Use a utility such as **tcpdump** or **ethereal** to interpret the packet-capture file.

Examples

- Initiates a packet-capture session on Ethernet 0. Packet-capture data is written to the file `Eth0Trace` in the general storage directory. The session terminates after 30 seconds or when `Eth0Trace` contains 2500 kilobytes of data (whichever occurs first).

```
# packet-capture store://Eth0Trace 1800 2500
Trace begun.
⋮
#
```

- Initiates and then terminates a packet-capture session.

```
# packet-capture store://Eth0Trace 1800 2500
Trace begun.
⋮
# no packet-capture store://Eth0Trace
#
```

---

standby

Implements a failover configuration

Syntax

Assign both interfaces to a group using a Virtual IP address (VIP).
standby *group* **ip** *VIP*

Assign a priority to a group.
standby *group* **priority** *priority*

Set the authentication parameters for the group.
standby *group* **auth** *auth-high auth-low*

Add additional IP addresses for a group.
standby *group* **ip-aux** *IP-address*[:*IP-address* ...]

Indicate preempt mode for the group.
standby *group* { **on** | **off** }

Delete a group from the current interface.
no standby *group*

Delete all groups from the current interface.
no standby

Parameters

group Identifies the standby group. Use an integer in the range of 1 to 255.

ip *VIP* Specifies the IP address that one member of this group listens on at any time. The value must be on the same IP subnet as the primary IP address of this VLAN.

priority *priority*
Specifies the priority of the interface. Use an integer in the range of 0 to 255. The default is 100.

The interface with the highest priority becomes the active interface. The interface with the lesser priority becomes the standby interface. For this reason, specify a priority of less than 100 for the standby member of the group.

on | **off**
Indicates whether to use preemption.

on Indicates that this interface forcibly claims the virtual IP address if the current owner of the virtual IP address has a lower priority. Higher priorities allow favoring one interface to be active at the cost of increasing the risk of broken TCP connections when the active interface changes.

off Indicates that the virtual IP address move only when the current owner is disconnected from the network.

auth *auth-high auth-low*
Specifies the first eight bytes of the authentication string in hexadecimal:

auth-high
Specifies the first four bytes. The default is `0x35554158`.

auth-low
Specifies the second four bytes. The default is `0x00000000`.

ip-aux *IP-address*[:*IP-address* ...]
Specifies additional IP addresses to install on the interface when it is the active interface. Separate multiple IP addresses with a colon.

Guidelines

The various forms of the **standby** command implement a standby configuration to ensure that an interface on another DataPower appliance is available if an active interface becomes unresponsive. In a standby configuration multiple interfaces coordinate the use of one virtual IP (VIP) address. Only one interface on the connected IP subnet can be active (listening) on that address at one time. The interfaces must all be in the same broadcast domain and must be able to receive IP packets that are sent to the multicast address `224.0.0.2` (all routers) from each other. The interfaces in a group should implement the same services.

Only the first form of the command is required to create a standby configuration. The other forms alter parameters that have default values. The following configuration parameters must be identical for all of the interfaces in one failover group:
- *group*
- **ip** *VIP*
- **auth** *auth-high auth-low*

By default, all interfaces in a group seek to be active at the same priority. The active interface changes only if that interface goes down or is disconnected. The priority parameter allows favoring some systems over others, with higher values being higher priority. The active interface changes only when the previous active interface goes down, unless preemption is enabled.

Only one interface, either Ethernet or VLAN, on a given physical Ethernet interface can have a failover configuration. Only one interface on a given system can have a failover configuration with a particular group VIP.

For detailed information about implementing a standby configuration, refer to the "Standby configurations" topic in the *IBM WebSphere DataPower SOA Appliances: Administrators Guide*.

To disable a failover configuration or to disable preemption, use the **no standby** command

Related Commands

interface, **ip address**

Examples

- Assigns `vlan-1` to standby group 2. Specifies a VIP of 10.10.66.66. Not specifying a priority (accepting the default of 100) ensures that the interface is the active member of the group. Places the interface in preempt mode meaning that it resumes the active role following a failure and subsequent restoration to service.

```
# vlan vlan-1
New VLAN Sub-Interface configuration
# ip address 10.10.66.1 255.0.0.0
# ip default-gateway 10.20.1.1
# standby 2 ip 10.10.66.66
# standby 2 preempt
# exit
```

- Assigns `vlan-2` to standby group 2. Specifies a VIP of 10.10.66.66. The priority value of 90 ensures that the interface is the standby member of the group. Because it is the standby member, it is not placed in preempt mode.

```
# vlan vlan-2
New VLAN Sub-Interface configuration
# 10.10.66.3 255.0.0.0
# ip default-gateway 10.30.1.1
# standby 2 ip 10.10.66.66
# standby 2 priority 90
# exit
```

- Assigns `vlan-3` to standby group 5 in the active role and specifies a VIP of 10.10.66.66. Not specifying a priority (accepting the default of 100) ensures that the interface is the active member of this group. Places the interface in preempt mode meaning that it resumes the active role following a failure and subsequent restoration to service.

```
# vlan vlan-3
New VLAN Sub-Interface configuration
# ip address 10.10.66.1 255.0.0.0
# ip default-gateway 10.20.1.1
# standby 5 ip 10.10.66.66
# standby 2 on
# exit
```

  Assigns `vlan-4` to standby group 7 in the standby role and specifies a VIP of 10.10.66.67. The priority value of 90 ensures that the interface is the standby member of the group. Because it is the standby member, it is not placed in preempt mode.

```
# vlan vlan-4
New VLAN Sub-Interface configuration
# ip address 10.10.66.2 255.0.0.0
# standby 7 ip 10.10.66.67
# standby 7 priority 90
# exit
```

- Disables preempt mode for standby group 2 on `vlan-3`.

```
# vlan vlan-3
Modify VLAN Sub-Interface configuration
# standby 2 off
# exit
```

- Deletes standby group 2 on `vlan-3`.

```
# vlan vlan-3
Modify VLAN Sub-Interface configuration
# no standby 2
# exit
```

- Deletes all standby groups on vlan-3.

```
# vlan vlan-3
Modify VLAN Sub-Interface configuration
# no standby
# exit
```

Chapter 100. Web Application Error Handling Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Application Error Handling Policy configuration mode.

To enter this mode, use the Global **webapp-error-handling** command. The global command creates the Error Handling Policy if the Policy does not exist. While in this mode, define the parameters for the Error Handling Policy object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are available in Error Handling Policy configuration mode.

error-monitor

Assigns or removes a Count Monitor.

Syntax

error-monitor *name*

no error-monitor

Parameters

name Specifies the name of an existing Count Monitor.

Guidelines

The Count Monitor can monitor transactions that are handled by this Error Handling Policy. The Count Monitor must be configured to count transactions that are generated by this Error Handling Policy (typically, the Count Monitor is configured to capture all events).

Use the Global **monitor-count** command to create a Count Monitor.

Related Commands

monitor-count

error-rule

Assigns or removes a Processing Rule.

Syntax

error-rule *name*

Parameters

name Specifies the name of an existing Processing Rule.

Guidelines

The Processing Rule runs when the Policy **type** is set to **error-rule**.

Use the Global **rule** command to create a new Processing Rule.

Related Commands

rule (Global), **type**

---

type

Establishes the mode of operation for this Error Handling Policy.

Syntax

type { **redirect** | **proxy** | **error-rule** | **standard** }

Parameters

redirect
The appliance redirects the client to the specified URL.

proxy The appliance will fetch the specified URL and then return its contents to the client.

error-rule
The appliance runs the specified error rule and return the result to the client.

standard
The appliance passes the error to the Application Security Policy selected for the Web Application Firewall. If the Application Security Policy includes an Error Map that will match the error, then that action is taken. This mode is useful , even if no Error Map matches the request, when you want to execute error handling rules for specific requests and want to enforce monitoring of all errors.

Related Commands

error-rule, **error-url**, **rule** (Global)

Chapter 101. Web Application Firewall configuration mode

This chapter provides an alphabetical listing of commands that are available in Web Application Firewall configuration mode.

To enter this configuration mode, use the Global **web-application-firewall** command. The Global command creates the Web Application Firewall if the Firewall does not exist. While in this mode, define the parameters for the Web Application Firewall object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Application Firewall configuration mode.

back-persistent-timeout

Sets the inter-transaction timeout value for firewall-to-server connections.

Syntax

back-persistent-timeout *time*

Parameters

time Specifies the maximum inter-transaction idle time. Use an integer in the range of 0 to 7200. The default is 180.

A value of 0 disables persistent connections.

Guidelines

Sets the inter-transaction timeout value, the maximum idle time allowed between the completion of a TCP transaction and the initiation of a new TCP transaction on the firewall-to-server connection. If the specified idle timeout is exceeded, the connection is torn down.

An idle TCP connection can remain in the idle state for as long as 20 seconds after the expiration of the inter-transaction timer.

Related Commands

back-timeout, **front-persistent-timeout**, **front-timeout**, **persistent-connections**

back-timeout

Sets the intra-transaction timeout value for firewall-to-server connections.

Syntax

back-timeout *time*

Parameters

time Specifies the maximum intra-transaction idle time. Use an integer in the range of 10 to 86400. The default is 120.

Guidelines

Sets the intra-transaction timeout value, the maximum idle time allowed within a transaction on the firewall-to-server connection. This timer, for example, monitors the interval between sending the client request and receiving the server response, and idle time within the data transfer process. If the specified idle time is exceeded, the connection is torn down.

Related Commands

back-persistent-timeout, **front-timeout**, **front-persistent-timeout**, **persistent-connections**

---

chunked-uploads

Controls the ability to send Content-Type Chunked Encoded documents to the backend server.

Syntax

chunked-uploads {**on** | **off**}

Parameters

on Enables chunked-encoded documents.

off (Default) Disables chunked-encoded documents. Alternatively, use the **no chunked-uploads** command.

Guidelines

The Gateway may send an HTTP 1.1 request to the back end server. In this case, the body of the document can be delimited by either Content-Length or chunked-encoded documents. All servers will understand how to interpret Content-Length, and many applications will fail to understand chunked, so Content-Length is generally used. However doing so interferes with the ability of the appliance to fully stream. If you must stream full documents towards the back side, this property should be turned on. However, you must know beforehand that the server you are communicating with is RFC 2616 compatible, unlike all other HTTP 1.1 features which can be negotiated down at runtime if necessary. You might also consider leaving this property turned off and turning it on a per-URL basis with the User Agent configuration.

---

error-policy

Assigns an Error Policy.

Syntax

error-policy *name*

no error-policy

Parameters

name Specifies the name of an existing Error Handling Policy.

Related Commands

security-policy, **webapp-error-handling** (Global)

Guidelines

An Error Policy determines the handling of errors encountered during processing. This is the default behavior for all requests and responses. It may be overridden by configurations set in the Security Policy.

By default, there is no Error Handling Policy assigned to the firewall. Use this command to assign a policy that acts as a default error handler for all transactions flowing through the service. This default behavior can be overridden by an Error Handling Policy set in the Security Policy.

Use the Global **webapp-error-handling** command to create a new Error Handling Policy.

follow-redirects

Controls attempts to resolve redirects.

Syntax

follow-redirects {**on** | **off**}

Parameters

on Enables the resolution of redirects.

off Disables the resolution of redirects. Alternatively, use the **no follow-redirects** command.

Guidelines

Some protocols generate redirects as part of the protocol; for example, HTTP response code 302. Use the **follow-redirects** command to specify if the DataPower service attempts to resolve redirects.

front-persistent-timeout

Sets the inter-transaction timeout value for firewall-to-client connections.

Syntax

front-persistent-timeout *time*

Parameters

time Specifies the maximum inter-transaction idle time. Use an integer in the range of 0 to 7200. The default is 180. A value of 0 disables persistent connections.

Related Commands

back-persistent-timeout, **back-timeout**, **front-timeout**, **persistent-connections**

Guidelines

Sets the inter-transaction timeout value, the maximum idle time allowed between the completion of a TCP transaction and the initiation of a new TCP transaction on the firewall-to-client connection. If the specified idle timeout is exceeded, the connection is torn down.

An idle TCP connection can remain in the idle state for as long as 20 seconds after the expiration of the persistence timer.

---

front-timeout

Sets the intra-transaction timeout value for firewall-to-client connections.

Syntax

front-timeout *time*

Parameters

time Specifies the maximum intra-transaction idle time. Use an integer in the range of 10 to 86400. The default is 120.

Guidelines

Sets the intra-transaction timeout value, the maximum idle time allowed within a transaction on the firewall-to-client connection. This timer, for example, monitors idle time within the data transfer process. If the specified idle time is exceeded, the connection is torn down.

Related Commands

back-persistent-timeout, **back-timeout**, **front-persistent-timeout**, **persistent-connections**

---

host-rewriting

Controls the rewriting of the `Host` header to reflect the final route.

Syntax

host-rewriting {**on** | **off**}

Parameters

on (Default) Indicates that the backend server receives a request that reflects the final route

off Indicates that the backend server receives a request that reflects the information as it arrived at the DataPower service.

Guidelines

Some protocols have distinct name-based elements, separate from the URL, to demultiplex. HTTP uses the `Host` header for this purposes.

Web servers that issue redirects might want to disable this feature. Web servers often depend on the `Host` header for the value of their redirect.

http-back-version

Selects the HTTP version to use on the server-side (backend) connection.

Syntax

http-back-version {**HTTP/1.0** | **HTTP/1.1**}

Parameters

HTTP/1.0
Uses HTTP 1.0.

HTTP/1.1
(Default) Uses HTTP 1.1.

http-client-ip-label

Sets the HTTP Client IP label (header name) in the HTTP header.

Syntax

http-client-ip-label *label*

Parameters

label Specifies the HTTP header label for the client IP address in HTTP communications. The default is `X-Client-IP`.

http-front-version

Selects the HTTP version to use on the client-side (frontend) connection.

Syntax

http-front-version {**HTTP/1.0** | **HTTP/1.1**}

Parameters

HTTP/1.0
Uses HTTP 1.0.

HTTP/1.1
(Default) Uses HTTP 1.1.

listen-on

Sets the addresses and ports on which the firewall listens.

Syntax

listen-on *address port use-SSL*

Parameters

address Specifies the local IP address on the appliance. Can be 0.0.0.0 to denote all local addresses, or can be a Host Alias.

port Specifies the TCP port number that this service listens on. No other service on the appliance can use this port.

use-SSL

Control SSL connections. Can be **on** or **off**. The defaults is **off**. When **on**, the SSL Proxy Profile that is specified with the **ssl-profile** command controls connections on this port.

Related Commands

ssl-profile

Guidelines

Issue this command as many times as needed to add the desired addresses and ports to this firewall. To delete these assignments, use the **no listen-on** command. This removes all assignments from the firewall.

Examples

- Sets the address of the firewall to 0.0.0.0 (all addresses assigned to the appliance) and the TCP port number to 3345. This assignment is then removed, leaving no address and port assigned. An assignment to only the 10.10.13.35 address and 3345 port.

```
# web-application-firewall portal-fw
Web Application Firewall configuration mode
# listen-on 0.0.0.0 3345
# no listen-on
# listen-on 10.10.13.35 3345
#
```

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal
(Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

remote-address

Specifies the address of the backend server.

Syntax

remote-address *address*

remote-address *load-balancer*

Parameters

address Specifies an IP address or host name of the server to route all traffic.

load-balancer
Specifies the name of an existing Load Balancer Group that identifies server address-port pairs of its members.

Related Commands

remote-port

remote-port

Establishes the TCP port number of remote (backend) application server.

Syntax

remote-port *port*

Parameters

port Specifies the TCP port to which all traffic is routed.

Related Commands

remote-address

request-security

Controls the enforcement of security on client requests.

Syntax

request-security {**on** | **off**}

Parameters

on Enables the enforcement of request security.

off Disables the enforcement of request security. Alternatively, use the **no request-security** command.

response-security

Controls the enforcement of security on server responses.

Syntax

response-security {**on** | **off**}

Parameters

on Enables the enforcement of response security.

off Disables the enforcement of response security. Alternatively, use the **no response-security** command.

security-policy

Assigns an Application Security Policy.

Syntax

security-policy *name*

Parameters

name Specifies the name of an existing Application Security Policy.

Guidelines

Specifies an Application Security Policy when configuring a Web Application Firewall. Use the Global **application-security-policy** command to create a policy.

Related Commands

application-security-policy (Global), **request-security**, **response-security**

---

ssl-profile

Assigns an SSL Proxy Profile.

Syntax

ssl-profile *name*

no ssl-profile

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

An SSL Proxy Profile specifies the SSL operational mode (client, server, or two-way) and identifies the cryptographic resources (key, certificates, and cipher lists) available to the SSL proxy.

Assignment of an SSL Proxy Profile to a Web Application Firewall is optional, unless the *useSSL* argument of the **listen-on** command is set to **on** for at least one address-port assignment. In the absence of an assigned SSL Proxy Profile, the firewall conducts all communications with both clients and servers over a nonsecure connection.

The SSL Proxy Profile must have previously created with the **sslproxy** command.

Use the **no ssl-profile** command to remove the SSL Proxy Profile assignment.

---

stream-output-to-back

Determines whether to begin sending output to the backend server before all processing is complete.

Syntax

stream-output-to-back {**buffer-until-verification** | **stream-until-infraction**}

Parameters

buffer-until-verification
(Default) Causes the Web Application Firewall to buffer submitted messages until all processing is verified complete. After verification, forwards messages to the appropriate backend URL.

stream-until-infraction
Causes the Web Application Firewall to begin sending the message to the backend URL before all processing is complete, potentially increasing the speed. If an infraction is encountered, the firewall reverts to buffered behavior. If the XML Manager that is selected for this firewall has streaming enabled, select **stream-until-infraction** to be certain that the firewall streams messages end-to-end.

Related Commands

stream-output-to-front

---

stream-output-to-front

Determines whether to begin sending output to the client before all processing is complete.

Syntax

stream-output-to-back {**buffer-until-verification** | **stream-until-infraction**}

Parameters

buffer-until-verification
(Default) Causes the Web Application Firewall to buffer submitted messages until all processing is verified complete. After verification, returns messages to the client.

stream-until-infraction
Causes the Web Application Firewall to begin sending the message to the client before all processing is complete, potentially increasing the speed. If an infraction is encountered, the firewall reverts to buffered behavior. If the XML Manager that is selected for this firewall has streaming enabled, select **stream-until-infraction** to be certain that the firewall streams messages end-to-end.

Related Commands

stream-output-to-back

---

uri-normalization

Controls the normalization of URIs before processing.

Syntax

uri-normalization {**on** | **off**}

Parameters

on (Default) Enables URI normalization.

off Disables URI normalization. Alternatively, use the **no uri-normalization** command.

Guidelines

Enables or disables the normalization of URIs before processing. If this property is enabled, the URI is rewritten to make sure the URI is RFC-compliant by escaping certain characters. Additionally, characters that are escaped that do not need to be are unescaped. This makes checking for attack sequences such as "`..`" more reliable.

---

xml-manager

Assigns an XML manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of the XML Manager. The default is `default`.

Guidelines

You do not need to change the default value. To use an XML Manager with user-specific characteristics, use the Global **xml-manager** command to create a new XML Manager. Use this command to associate it with the current Web Application Firewall.

Related Commands

security-policy, **xml-manager** (Global)

Chapter 102. Web Application Name Value Profile configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Application Name Value Profile configuration mode.

To enter this configuration mode, use the Global **webapp-gnvc** command. The Global command creates the Web Application Name Value Profile if the Policy does not exist. While in this mode, define the parameters for the Web Application Name Value Profile object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Application Name Value Profile configuration mode.

max-aggregate-size

Specifies the maximum size of the combined name-value pairs.

Syntax

max-aggregate-size *bytes*

Parameters

bytes Specifies the maximum size in bytes. The default is 128000.

max-attributes

Specifies the maximum number of attributes in name-nalue pairs to allow.

Syntax

max-aggregate-size *number*

Parameters

number
Specifies the maximum number of attributes. The default is 256.

max-name-size

Specifies the maximum number of characters in the `Name` attribute of name-value pairs to allow.

Syntax

max-name-size *characters*

Parameters

characters
Specifies the maximum number of characters in the `Name` attribute The default is 512.

Related Commands

max-value-size

---

max-value-size

Specifies the maximum number of characters in the `Value` attribute of name-value pairs to allow.

Syntax

max-value-size *characters*

Parameters

characters
Specifies the maximum number of characters in the `Value` attribute The default is 1024.

Related Commands

max-name-size

---

unvalidated-fixup-map

Assigns the value to a `Value` attribute that does not match an entry in the validation list.

Syntax

unvalidated-fixup-map *value*

Parameters

value Specifies an alphanumeric string. Use quotation marks around the string.

Guidelines

The value of the `Value` attribute of any name-value pair that does not match at least one entry in the validation list is replaced with this constant value when **unvalidated-fixup-policy** is **set**.

Related Commands

unvalidated-fixup-policy

---

unvalidated-fixup-policy

Determines the action taken for name-value pairs that do not match any of the entries in the validation list.

Syntax

unvalidated-fixup-policy {**error** | **passthru** | **set** | **strip**}

Parameters

error Generates an error. The Error Handling Policy or the Error Handling Map can then handle the error condition.

passthru
Passes the name-value pair through for further processing.

set Replaces the `Value` attribute with the string set by the **unvalidated-fixup-map** command.

strip Removes the name-value pair from the entity (HTTP header, HTTP body, or query string).

Related Commands

unvalidated-fixup-map

---

unvalidated-xss-check

Controls checking of name-value pairs that do not match an entry in the validation list for Cross Site Scripting signatures.

Syntax

unvalidated-xss-check {**on** | **off**}

Parameters

on Enables checking.

off Disables checking.

Guidelines

Cross-site scripting (XSS) signatures are generally attempts to obfuscate the real meaning of the value if the value were displayed directly in a browser. You want to validate any data that might get stored and displayed again later, such as the contents of a comment form. The check looks for escaped characters, characters with the high-bit set, and various forms of the term script, which is often used to engage JavaScript on a browser without the user knowing.

Related Commands

unvalidated-fixup-map

---

validation

Creates a validation lList.

Syntax

validation *name-PCRE value-PCRE policy* [*check-XSS*]

Parameters

name-PCRE
Specifies a PCRE that the submitted names are matched against. If they match the value must also match against the corresponding value constraint to be passed through.

value-PCRE
Specifies a PCRE that is applied to a value input to see if it is an expected input.

policy Specifies the action to take when a value does not match the expression. Values are as follows:

error (Default) The profile validation fails and an error is generated.

passthru
Passes the given name-value pair to the next step in processing.

set Replaces the given value is replaced with a default value. The Map Value input appears when this option is selected.

strip The name-value pair is removed from the entity and processing continues.

check-XSS
Can be **on** to enable Cross-Site Script checking, or can be **off** to disable checking.

Guidelines

The **validation** command creates the validation list.

The Name-Value Profile works by comparing each name of a name-value pair to the name expressions on the validation list. If no match is found, the "No Match Policy" is run. When a match is found, the corresponding value is compared to the corresponding value constraint in the validation list. If a match is found, the name-value pair passes. If no match is found, the "Failure Policy" is executed. Additionally, unmatched values can be checked for Cross-Site Scripting.

Cross-site scripting signatures are generally attempts to obfuscate the real meaning of the value if the value were displayed directly in a browser. You want to validate any data that might get stored and displayed again later, such as the contents of a comment form. The check looks for escaped characters, characters with the high-bit set, and various forms of the term script, which is often used to engage JavaScript™ on a browser without the user knowing.

Examples

- Matches any `Name` attribute that contains the string `hdr`. When a match is made, the value must start with `PRE` followed by any numeric character 0 through 8, followed by anything. If the validation fails, the name-value pair is stripped from the entity. Cross-Site Script checking is disabled.

  ```
  # validation hdr "^PRE[012345678]*" strip "" "off"
  ```

- Matches any `Name` attribute that contains the string `hdr`. When a match is made, the value must start with `PRE` followed by any numeric character 0 through 8, followed by anything. If the validation fails, the `Value` attribute is set to the string `BogieAlert`. Cross-Site Script checking is enabled.

  ```
  # validation hdr "^PRE[012345678]*" set "BogieAlert" "on"
  ```

Chapter 103. Web Application Request Profile configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Application Request Profile configuration mode.

to enter this configuration mode, Use the Global **webapp-request-profile** command. The global command creates the Web Application Request Profile if the Request Profile does not exist. While in this mode, define the parameters for the Web Application Request Profile object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Application Request Profile configuration mode.

aaa-policy

Assigns an AAA Policy.

Syntax

aaa-policy *name*

no aaa-policy *name*

Parameters

name Specifies the name of an existing AAA Policy.

Guidelines

The **aaa-policy** command assigns an AAA policy to the Web Application Request Profile. The AAA Policy applies to all requests. Use the Global **aaapolicy** command to create an AAA Policy.

Use the **no aaa-policy** command to remove the AAA Policy.

Related Commands

aaapolicy (Global)

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

no acl *name*

Parameters

name Specifies the name of an existing Access Control List.

Guidelines

The **acl** command assigns an Access Control List to the Web Application Request Profile. The Access Control List applies to all requests. Use the Global **acl** command to create an Access Control List.

Use the **no acl** command to remove the Access Control List. Without an Access Control List, no restrictions are enforced on clients that make requests.

Related Commands

acl (Global)

---

cookie-policy

Sets the Cookie processing policy.

Syntax

cookie-policy *policy type* [*key*] [*in-watermark*] [*GNVC*]

Parameters

policy Specifies the requirement of presenting a cookie as part of the request.

allow (Default) Allows requests whether they present a cookie or not.

deny Denies requests that present a cookie.

require
Requires requests to present a cookie.

If the request does not conform, setting to **deny** or **require** might cause an error.

type Specifies what to do with the cookie contents in the request.

none (Default) Does not encrypt or sign cookie contents.

encrypt
Encrypts cookie contents using the specified key.

sign Appends a digital signature to the cookie contents using the specified key.

key Specifies the secret passphrase to encrypt or sign cookie contents. If the key is the same on multiple appliances, each appliance decrypts or verifies the cookie contents with a key that is generated by another appliance without maintaining state.

in-watermark
Generally signed or encrypted cookie contains the client IP address to prevent the client from using this cookie from any other host. Some proxy environments might make this behavior undesirable.

on Adds the IP address to the cookie.

off (Default) Does not add the IP address to the cookie. The cookie is IP address independent.

GNVC Specifies the name of an existing Name-Value Profile to apply to cookie contents. If not present, no profile is applied.

Guidelines

The **cookie-policy** command sets the Cookie processing policy for this Request Profile. Requests that violate these limits cause an error. By default, cookies are allowed, but they are not encrypted or signed.

Use the Global **webapp-gnvc** command to create a Name-Value Profile.

Examples

- Requires requests to present cookies. The appliance signs the cookie contents with the secret phrase `mysecretkey`. The IP address is used in the cookie. The cookie contents are evaluated with the `portal-pairs` Name Value Profile.

```
# cookie-policy require sign mysecretkey on port-pairs
#
```

- Returns the Cookie Policy to the default state, which allow cookies but not requiring them and performing no other processing tasks.

```
# cookie-policy allow none
#
```

error-policy-override

Assigns an Error Policy.

Syntax

error-policy-override *name*

no error-policy-override

Parameters

name Specifies the name of an existing Error Handling Policy.

Related Commands

web-application-firewall (Global), **webapp-error-handling** (Global)

Guidelines

Establishes an Error Policy for the Request Profile. An Error Policy determines the handling of errors that are encountered during processing. This is the default behavior for all requests and responses that are handled by this Profile. The Error Policy might override the Error Policy configurations set in the Security Policy of the Web Application Firewall using this Request Policy.

By default, there is no Error Handling Policy assigned to the Profile. Use this command to assign a policy that acts as a default error handler for all transactions flowing through this Profile.

Use the Global **webapp-error-handling** command to create an Error Handling Policy.

Use the **no error-policy-override** command to set the Error Policy to none.

Examples

- Assigns the `req-1-errors` Error Handling Policy.

  ```
  # error-policy-override req-1-errors
  ```

- Sets the error handling policy to none, which effectively disables error handling.

  ```
  # no error-policy-override
  ```

---

multipart-form-data

Sets the policy for processing multipart requests.

Syntax

multipart-form-data *parts maximum-part-size maximum-size* {**on** | **off**}

Parameters

parts Specifies the maximum number of parts to allow. The default is 4.

maximum-part-size
Specifies the maximum size of any one part to allow. The default is 5000000.

maximum-size
Specifies the maximum size of all parts combined to allow. The default is 5000000.

on | **off**
If **on**, forces the individual form-data content types to be matched against the general list of request acceptable content-type expressions. The default is **off**.

Guidelines

The defaults are enforced automatically. Use the **request-content-type** command to create content types.

Related Commands

request-content-type

Examples

- Expands allowed parts to 12, makes sizes smaller and enforces content-type restriction.

  ```
  # multipart-form-data 12 50000 250000 on
  #
  ```

---

policy-type

Determines the operational mode.

Syntax

policy-type {**admission** | **pre-requisite**}

Parameters

admission
(Default) If a request passes the criteria set forth in this profile, the client's

request (the transaction request) is immediately forwarded to the back end service. No other matching profile is run.

pre-requisite
If a request passes the criteria set forth in this profile, any other profiles that match the request may now run. The request is not necessarily forwarded to the back end service. However, if there are no other matching profiles and the request passes this profile, the request will then be passed to the back end service.

Guidelines

The **policy-type** command determines how the current Profile operates. However, an Application Security Policy can run more than one Request Profile on any single transaction. When there is more than one Request Profile, the policy type determines how the current profile relates to other profiles in the Application Security Policy.

Related Commands

application-security-policy (Global)

---

ratelimiter-policy

Assigns or removes a Rate Limit Policy Applies to all requests.

Syntax

ratelimiter-policy *name*

no ratelimiter-policy *name*

Parameters

name Specifies the name of an existing Rate Limit Policy.

Guidelines

A Rate Limit policy is optional. A Rate Limit Policy restricts identities, as determined by an AAA Policy or the client IP address (when not using an AAA Policy), to a specific number of transactions per second or a specific number of concurrent transaction connections. Use the Global **simple-rate-limit** command to create a policy.

Use the **no ratelimiter-policy** command to remove the policy, which enforces no rate limiting on requests.

Related Commands

simple-rate-limit (Global)

Examples

- Assigns the `portal-rates` Rate Limit Policy to the current Web Application Request Profile.

```
# ratelimiter-policy portal-rates
#
```

- Removes the Rate Limit Policy from the current Web Application Request Profile.

```
# no ratelimiter-policy
#
```

request-body-max

Specifies the maximum request body size in bytes, if the HTTP method provides a body.

Syntax

request-body-max *bytes*

Parameters

bytes Specifies the maximum request body size in bytes. The default is 128000000.

Related Commands

request-body-min

request-body-min

Specifies the minimum request body size in bytes, if the HTTP method provides a body.

Syntax

request-body-min *bytes*

Parameters

bytes Specifies the minimum request body size in bytes. The default is 0.

Related Commands

request-body-max

request-body-profile

Assigns or removes the Name-Value Profile to process URL-encoded HTTP POST body content.

Syntax

request-body-profile *name*

no request-body-profile *name*

Parameters

name Specifies the name of an existing Name-Value Profile.

Guidelines

Use the Global **webapp-gnvc** command to create a Name-Value Profile. If no Name-Value Profile is specified, no processing occurs.

Use the **no request-body-profile** command to remove any profile assigned using this command.

Related Commands

webapp-gnvc (Global)

request-content-type

Sets the HTTP content types to allow.

Syntax

request-content-type *PCRE*

no request-content-type *PCRE*

no request-content-type

Parameters

PCRE Specifies a string representation of the Content type.

Guidelines

Use this command as many times as needed to create a list of HTTP Content types that this profile allows. If no Content types are specified, all Content types are allowed. You can use a PCRE expression to match one or more HTTP Content type expressions.

Use the **no** form with the *PCRE* to remove only the designated Content type from the list.

Use the **no**form without the *PCRE* argument to remove all Content types that are assigned.

Examples

- Adds `text/xml` and `text/html` to the Contents types list.

```
# request-content-type text/html
# request-content-type text/xml
#
```

- Removes `text/xml` from the Content types list.

```
# no request-content-type text/xml
#
```

request-header-profile

Sets the Name-Value Profile to process HTTP Header content.

Syntax

request-header-profile *name*

no request-header-profile *name*

Parameters

name Specifies the name of an existing Name-Value Profile.

Guidelines

If no Name-Value Profile is specified, no processing occurs. Use the Global **webapp-gnvc** command to create a new profile.

Use the **no request-header-profile** command to remove any Name-Value Profile that is assigned.

Related Commands

webapp-gnvc (Global)

Examples

- Assigns the `portal-hdr-nvp` Name-Value Profile.

```
# request-header-profile portal-hdr-nvp
#
```

- Removes all Name-Value Profile assignments.

```
# no request-header-profile
#
```

---

request-methods

Sets the HTTP methods to allow.

Syntax

request-methods *Method*[+*Method*]...

Parameters

Method[+*Method*]...

Specifies one or more of the allowed Methods, concatenated with the + symbol.

- **POST**
- **GET**
- **PUT**
- **HEAD**
- **OPTIONS**
- **TRACE**
- **DELETE**

Guidelines

Specifies all methods to allow when using this command. If the current settings allow four methods and you want to add a fifth, you must specify all five methods with the command.

Related Commands

request-version

Examples

- Adds the HTTP TRACE method to the default methods (GET, POST and HEAD) to allow.

```
# request-methods GET+POST+HEAD+TRACE
#
```

request-nonxml-policy

Determines how to handle non-XML content.

Syntax

request-nonxml-policy {**nothing** | **side** | **binary**}

Parameters

nothing
(Default) Performs no processing.

side The appliance executes the Non-XML Processing Rule specified. This rule cannot alter the content of the request (cannot access the INPUT and OUTPUT contexts). The Rule can perform such actions as authenticate and authorize, or send a copy of the request content to a third destination.

binary The appliance executes the Non-XML Processing Rule specified. The request payload is submitted as a non-parsed binary object. This rule can alter the content of the request. The Rule can perform such actions as authenticate and authorize, convert to XML, repackage with additional information retrieved from elsewhere and/or send a copy of the request content to a third destination. The result of this rule is then used as the request payload for further processing.

Related Commands

request-nonxml-rule, **request-xml-policy**, **request-xml-rule**

request-nonxml-rule

Specifies the Processing Rule to apply to non-XML content.

Syntax

request-nonxml-rule *name*

Parameters

name Specifies the name of an existing Processing Rule.

Guidelines

Specifies the Processing Rule to apply to non-XML content when the **request-nonxml-policy** is either **side** or **binary**.

Related Commands

request-nonxml-policy, **request-xml-policy**, **request-xml-rule**, **rule** (Global)

Examples

- Sets the policy for non-XML requests to run a side effect Processing Rule, which does not change the content of the request but does check authentication. The Processing Rule is then identified.

```
# request-nonxml-policy side
# request-nonxml-rule request-aaa
#
```

---

request-qs-policy

Determines how to handle HTTP Query Strings.

Syntax

request-qs-policy {**allow** | **deny** | **require**}

Parameters

allow (Default) Allows Query Strings in the request.

deny Rejects requests that contain Query Strings.

require
Requires request to contain Query Strings. Rejects requests without Query Strings.

Related Commands

request-qs-profile

---

request-qs-profile

Specifies the Name-Value Profile to apply to HTTP Query Strings.

Syntax

request-qs-profile *name*

no request-qs-profile *name*

Parameters

name Specifies the name of an existing Name Value Profile.

Guidelines

Use the **request-qs-profile** command to set the Name-Value Profile to apply to Query Strings when **request-qs-policy** is either **allow** or **require**. If no profile is specified, no processing is applied. Use the Global **webapp-gnvc** command to create a Name-Value Profile.

Use the **no request-qs-profile** command to remove any assigned Name-Value Profile.

Related Commands

request-qs-policy, **webapp-gnvc** (Global)

request-sql-policy

Enables or disables a filter for SQL Injection attack threats.

Syntax

request-sql-policy {**on** | **off**}

Parameters

on (Default) Filters for SQL Injection attack threats.

off Disables the filter.

Related Commands

request-uri-filter-dotdot, **request-uri-filter-exe**, **request-uri-filter-fragment**, **request-uri-filter-unicode**

request-ssl-policy

Determines how to handle handles SSL communications with the requesting client.

Syntax

request-ssl-policy {**allow** | **deny** | **require**}

Parameters

allow (Default) Allows SSL connections. If the SSL Proxy Profile specified at the Web Application Firewall level is configured to act as an SSL server, the connection request will be accepted; if it is not, the request will be refused.

deny Rejects requests that request SSL connections.

require
Requires the client to use SSL communications. The SSL Proxy Profile specified at the Web Application Firewall level must be configured to act as an SSL server.

Related Commands

ssl-profile (Web Application Firewall)

request-uri-filter-dotdot

Controls a filter for URLs that include the string `..` (dot dot) after URI normalization is performed.

Syntax

request-uri-filter-dotdot {**on** | **off**}

Parameters

on (Default) Filters all content for a `..` string.

off Disables the filter.

Related Commands

request-sql-policy, **request-uri-filter-exe**, **request-uri-filter-fragment**, **request-uri-filter-unicode**

---

request-uri-filter-exe

Controls a filter for URLs that include the string exe after URI normalization is performed.

Syntax

request-uri-filter-exe {**on** | **off**}

Parameters

on (Default) Filters for the exe string.

off Disables the filter.

Related Commands

request-sql-policy, **request-uri-filter-dotdot**, **request-uri-filter-fragment**, **request-uri-filter-unicode**

---

request-uri-filter-fragment

Determines how to handle URI fragments.

Syntax

request-uri-filter-fragment {**allow** | **reject** | **truncate**}

Parameters

allow (Default) Allows requests that contain URI fragments.

reject Rejects requests that contain URI fragments.

truncate
Processes requests after removing the URI fragment.

Guidelines

The **request-uri-filter-fragment** command determines how to handle URI fragments. A URI fragment is the protion of a URI after the hash (#) symbol.

Related Commands

request-sql-policy, **request-uri-filter-dotdot**, **request-uri-filter-exe**, **request-uri-filter-unicode**

---

request-uri-filter-unicode

Controls the filter for URLs that include Unicode after URI normalization is performed.

Syntax

request-uri-filter-unicode {**on** | **off**}

Parameters

on (Default) Filters for Unicode.

off Disables the filter.

Related Commands

request-sql-policy, **request-uri-filter-dotdot**, **request-uri-filter-exe**, **request-uri-filter-fragment**

---

request-uri-max

Sets the maximum size to allow for the entire URI.

Syntax

request-uri-max *characters*

Parameters

characters
Specifies the maximum number of characters. The default is 1024.

Related Commands

request-sql-policy, **request-uri-filter-dotdot**, **request-uri-filter-exe**, **request-uri-filter-fragment**, **request-uri-filter-unicode**

---

request-versions

Sets HTTP protocol versions to support.

Syntax

request-versions {**HTTP/1.0** | **HTTP/1.1**}

Parameters

HTTP/1.0
Specifies HTTP 1.0.

HTTP/1.1
Specifies HTTP 1.1

Guidelines

This protocol version automatically negotiates down to 1.0.

Related Commands

request-methods

---

request-xml-policy

Determines how to handle XML content.

Syntax

request-xml-policy {**nothing** | **xml** | **soap**}

Parameters

nothing
(Default) Performs no processing.

xml The appliance parses the response to validate that the response is well-formed XML. The XML Transformation Rule specified then runs on the response and the result is used as the response content.

soap The appliance parses the response to validate that the response adheres to the SOAP specifications. The XML Transformation Rule specified then runs on the response and the result is used as the response content.

Related Commands

request-nonxml-policy, **request-nonxml-rule**, **request-xml-rule**

Examples

- Sets the policy for XML requests to validate that the request is well-formed XML. A Processing Rule is then configured to run on the request.

```
# request-xml-policy xml
# request-xml-rule request-aaa
#
```

request-xml-rule

Specifies the Processing Rule to apply to XML content.

Syntax

request-xml-rule *name*

Parameters

name The name of an existing Processing Rule.

Guidelines

Specifies the Processing Rule to apply to XML content when **request-xml-policy** is either **xml** or **soap**. Without a Processing Rule, no rule is run. Even though no rule is run, the DataPower appliance might if the XML is well-formed or validate SOAP requests against the SOAP schemas.

Related Commands

request-nonxml-policy, **request-nonxml-rule**, **request-xml-policy**, **rule** (Global)

Examples

- Sets the policy for XML requests to validate that the request is well-formed XML. A Processing Rule is then configured to run on the request.

```
# request-xml-policy xml
# request-xml-rule request-aaa
#
```

session-policy

Assigns a Session Management Policy.

Syntax

session-policy *name*

no session-policy *name*

Parameters

name Specifies the name of an existing Session Management Policy.

Guidelines

Without a Session Management Policy, no Policy is applied. Use the Global **webapp-session-management** command to create a Session Management Policy.

Use the **no session-policy** to remove any Session Policy assignment .

Related Commands

webapp-session-management (Global)

Chapter 104. Web Application Response Profile configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Application Response Profile configuration mode.

To enter this configuration mode, use the Global **webapp-response-profile** command. The Global command creates the Web Application Response Profile if the Response Profile does not exist. While in this mode, define the parameters for the Web Application Response Profile object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Application Response Profile configuration mode.

error-policy-override

Assigns an Error Policy.

Syntax

error-policy-override *name*

no error-policy-override

Parameters

name Specifies the name of an existing Error Handling Policy.

Guidelines

Establishes an optional Error Policy for the Response Profile. An Error Policy determines the handling of errors that are encountered during processing. This is the default behavior for all responses and responses handled by this Response Profile. It can override the Error Policy configurations set in the Security Policy of the Web Application Firewall that uses this Response Profile.

By default, there is no Error Handling Policy assigned to the Response Profile. Use this command to assign a policy that acts as a default error handler for all transactions that flow through this Response Profile.

Use the Global **webapp-error-handling** command to create a new Error Handling Policy.

Use the**no error-policy-override** command to set the Error Policy to none.

Related Commands

web-application-firewall (Global), **webapp-error-handling** (Global)

Examples

- Sets the `rsp-1-errors` Error Handling Policy.

  ```
  # error-policy-override rsp-1-errors
  ```

- Sets the Error Handling Policy to none, which effectively disables the error handling.

  ```
  # no error-policy-override
  ```

---

policy-type

Determines the satisfaction policy.

Syntax

policy-type {**admission** | **pre-requisite**}

Parameters

admission
: If a response passes the criteria set forth in this profile, the client's response (the transaction response) is immediately forwarded to the back end service. No other matching profile is run. This is the default.

pre-requisite
: If a response passes the criteria set forth in this profile, any other profiles that match the response may now run. The response is not necessarily forwarded to the back end service. However, if there are no other matching profiles and the response passes this profile, the response will then be passed to the backend service.

Guidelines

Each transaction can match more than one response profile on the same transaction. When this happens, the satisfaction style (policy type) helps to determine how the results of the profiles are combined.

- When the policy type is **pre-requisite**, all matching profiles in the Application Security Policy are run. A failure of any of these profiles results in the failure of the transaction. Only when all matching profiles are successful will the transaction be accepted.

  A typical use of this style would be a broad match that enforces a basic item, such as maximum size, that is followed up with more specific matches with stronger criteria.
- When the policy type is **admission**, only this policy is run. If this profile is successful, the transaction is accepted.

A failed profile always results in the failure of the transaction.

Related Commands

application-security-policy (Global)

---

response-body-max

Determine the maximum response body size if the HTTP method provides a body.

Syntax

response-body-max *bytes*

Parameters

bytes Specifies the maximum size of the response body in bytes, if the HTTP method provides a body. The default is 128000000.

Related Commands

response-body-min

---

response-body-min

Determine the minimum response body size if the HTTP method provides a body.

Syntax

response-body-min *bytes*

Parameters

bytes Specifies the minimum size of the response body in bytes, if the HTTP method provides a body. The default is 0.

Related Commands

response-body-max

---

response-codes

Sets the HTTP methods to allow.

Syntax

response-codes *code*[+*code*]...

Parameters

code[+*code*]...

Specifies one or more codes to allow. To specify multiple codes, concatenate with the + symbol. The default values are shown in this list. Possible values:

- `HTTP-100` — 100 Continue (default)
- `HTTP-101` — 101 Switching Protocols
- `HTTP-200` — 200 OK (default)
- `HTTP-201` — 201 Created (default)
- `HTTP-202` — 202 Accepted (default)
- `HTTP-203` — 203 Informative
- `HTTP-204` — 204 No Content (default)
- `HTTP-205` — 205 Reset
- `HTTP-206` — 206 Partial Content (default)
- `HTTP-300` — 300 Multiple Choices
- `HTTP-301` — Moved (default)
- `HTTP-302` — Found (default)
- `HTTP-303` — See Other
- `HTTP-304` — Not Modified (default)
- `HTTP-305` — Use Proxy
- `HTTP-307` — Moved Temporarily (default)
- `HTTP-400` — Bad Request
- `HTTP-401` — Unauthorized

- HTTP-402 — Payment Required
- HTTP-403 — Forbidden
- HTTP-404 — Not Found
- HTTP-405 — Method Not Allowed
- HTTP-406 — Not Acceptable
- HTTP-407 — Proxy Authentication Required
- HTTP-408 — Request Timeout
- HTTP-409 — Conflict
- HTTP-410 — Gone
- HTTP-411 — Length required
- HTTP-412 — Precondition Failed
- HTTP-413 — Request Entity Too Large
- HTTP-500 — Server Error
- HTTP-503 — Service Unavailable

Guidelines

You must specify all methods allowed when using this command. Thus, if the current settings allow four methods and you want to add a fifth, you must specify all five methods with the command.

Related Commands

response-version

Examples

- Sets the allowed response codes to HTTP-100 and HTTP-200 only.

```
# response-codes HTTP-100+HTTP-200
#
```

---

response-content-type

Sets the HTTP Content types to allow.

Syntax

response-content-type *PCRE*

no response-content-type *PCRE*

Parameters

PCRE Specifies a string representation of the Content type to allow (such as text/xml). This is a PCRE expression that may match one or more HTTP Content type expressions.

Guidelines

Use this command as many times as desired to create a list of HTTP Content types allowed by this profile. If no Content types are specified, all Content types are allowed. Use the **no response-content-type** command with no *PCRE* parameter to remove all types assigned using this command. Use the **no response-content-type** command with the *PCRE* parameter to remove only the designated Content type from the list.

Examples

- Sets the allowed Content types totext/xml and text/html.

```
# response-content-type text/html
# response-content-type text/xml
#
```

- Removes`text/xml` from the allowed Content types.

```
# no response-content-type text/xml
#
```

response-header-profile

Sets the Name-Value Profile to process HTTP Header content.

Syntax

response-header-profile *name*

no response-header-profile *name*

Parameters

name Specifies the name of an existing Name-Value Profile.

Guidelines

Use the Global **webapp-gnvc** command to create a Name-Value Profile.

Use the **no response-header-profile** command to remove any profile assigned. If no Profile is specified, no processing occurs.

Related Commands

webapp-gnvc (Global)

Examples

- Specifies the `portal-hdr-nvp` Name-Value Profile to apply the HTTP header content.

```
# response-header-profile portal-hdr-nvp
#
```

- Removes the Name-Value Profile.

```
# no response-header-profile
#
```

response-nonxml-policy

Determines how to handle non-XML content.

Syntax

response-nonxml-policy {**nothing** | **side** | **binary**}

Parameters

nothing
(Default) No processing performed.

side The appliance executes the Non-XML Processing Rule specified. This rule cannot alter the content of the response (cannot access the INPUT and

OUTPUT multistep processing contexts). The Rule can perform such actions as authenticate and authorize, or send a copy of the response content to a third destination.

binary The appliance executes the Non-XML Processing Rule specified. The response payload is submitted as an unparsed binary object. This rule can alter the content of the response. The Rule can perform such actions as authenticate and authorize, convert to XML, repackage with additional information retrieved from elsewhere and/or send a copy of the response content to a third destination. The result of this rule is then used as the response payload for further processing.

Related Commands

request-nonxml-rule, **request-xml-policy**, **response-xml-rule**

---

response-nonxml-rule

Specifies the Processing Rule to apply to non-XML content.

Syntax

response-nonxml-rule *name*

Parameters

name Specifies the name of an existing Processing Rule.

Guidelines

The **response-nonxml-rule** specifies the Processing Rule to apply to non-XML content. Use this command when the **response-nonxml-policy** is either **side** or **binary**.

Related Commands

request-nonxml-policy, **request-xml-policy**, **response-xml-rule**, **rule** (Global)

---

response-versions

Sets HTTP protocol versions to support.

Syntax

response-versions {**HTTP/1.0** | **HTTP/1.1**}

Parameters

HTTP/1.0
Specifies HTTP 1.0.

HTTP/1.1
(Default) Specifies HTTP 1.1.

Guidelines

The protocol version automatically negotiates down to 1.0.

Related Commands

response-methods

response-xml-policy

Determines how to handle XML content.

Syntax

response-nonxml-policy {**nothing** | **xml** | **soap**}

Parameters

nothing
(Default) No processing performed.

xml The appliance parses the response to validate that the response is well-formed XML. The XML Transformation Rule specified then runs on the response and the result is used as the response content.

soap The appliance parses the response to validate that the response adheres to the SOAP specifications. The XML Transformation Rule specified then runs on the response and the result is used as the response content.

Related Commands

request-nonxml-policy, **request-nonxml-rule**, **response-xml-rule**

response-xml-rule

Specifies the Processing Rule to apply to XML content.

Syntax

response-nonxml-rule *name*

Parameters

name Specifies the name of an existing Processing Rule.

Guidelines

Specifies the Processing Rule to apply to XML content when the **response-xml-policy** is either **xml** or **soap**.

Without a Processing Rule, no Processing Rule is run. Even though no Processing Rule is run, the DataPower appliance might check to determine whether the XML is well-formed or might validate SOAP responses against the SOAP schemas.

Related Commands

request-nonxml-policy, **request-nonxml-rule**, **response-xml-policy**, **rule** (Global)

Chapter 105. Web Application Session Management Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Application Session Management Policy configuration mode.

To enter this configuration mode, use the Global **webapp-session-management** command. The Global command creates the Web Application Session Management Policy if the Policy does not exist. While in this mode, define the parameters for the Web Application Session Management Policy object.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Application Session Management Policy configuration mode.

allow-cookie-sharing

Enables or disables the use of cookies by more than one Internet address.

Syntax

allow-cookie-sharing {**on** | **off**}

Parameters

on Makes the session cookie address independent.

off (Default) Cookies cannot be shared by IP addresses.

Guidelines

The **allow-cookie-sharing** command enables or disables the use of cookies by more than one Internet address. Enabling cooking-sharing is useful when the remote client is behind a proxy. Normally the session cookie contains the client IP address, and this prevents them from using the session on any other host. Some proxy server environments might make this undesirable.

auto-renew

Enables or disables the automatic renewal of a session whenever the user takes action.

Syntax

auto-renew {**on** | **off**}

on (Default) Renews the session lifetime on each use of the session.

off The session lifetime is the total amount of time to allow before returning to the login section.

Guidelines

The **auto-renew** command enables or disables the automatic renewal of a session whenever the user takes action. The click of a mouse or submission of a form constitutes a use. When enables, the session lifetime measures idle time between uses.

Related Commands

lifetime

---

lifetime

Determines the length, in seconds, of a session.

Syntax

lifetime *seconds*

Parameters

seconds
Specifies the duration of a session in seconds. The default is 3600.

Related Commands

auto-renew

---

matching-policy

Sets the Matching Rule to determine the URLs of session start pages.

Syntax

matching-policy *name*

Parameters

name Specifies the name of an existing Match Rule to identify start pages.

Guidelines

The **matching-policy** command sets the Matching Rule to determine the URLs of session start pages. Start pages are pages that can be accessed without a session cookie. If the security policy is enforced on the start page, these pages will issue a session cookie.

Use the Global **match** command to create a new Match Rule.

Chapter 106. Web Management Service configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Management Service configuration mode. To enter this configuration mode, use the Global **web-mgmt** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Management Service configuration mode.

idle-timeout

Specifies the amount of idle time before closing a WebGUI session because of inactivity.

Syntax

idle-timeout *seconds*

Parameters

seconds
Specifies the timeout value of the idle session in seconds. Use an integer in the range of 0 through 65535.

local-address

Identifies the local address to monitor for requests.

Syntax

local-address *address*:*port*

Parameters

address:*port*
Specifies the IP address and port number monitored for incoming Web Management Service requests.

Guidelines

You can use the special IP address 0.0.0.0 to specify all local addresses.

Examples

- Specifies that port 8090 on all interfaces is monitored for incoming Web Management Service requests.

```
# web-mgmt
Web Management Service configuration mode
# local-address 0.0.0.0:8090
#
```

save-config-overwrite

Specifies system behavior after a running configuration is saved.

Syntax

save-config overwrite

Guidelines

By default the **Save Config** button and the **write mem** command write the current running configuration to `config:///autoconfig.cfg`, and designate that file as the startup configuration. To override the default behavior, place the **no** form of this command in a startup configuration script.

Related Commands

boot config, **write mem**

Examples

- Assuming that this file is the startup configuration, clicking the **Save Config** WebGUI button:
  - Writes the running configuration to `config:///autoconfig.cfg`
  - Retains the file designated by the **boot config** command as the startup configuration

```
# no save-config overwrite
#
```

- Assuming that this file is the startup configuration, clicking the **Save Config** WebGUI button:
  - Writes the running configuration to `config:///autoconfig.cfg`
  - Designates `config:///autoconfig.cfg` as the startup configuration

```
# save-config overwrite
#
```

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Chapter 107. Web Service Proxy configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Service Proxy configuration mode.

To enter this configuration mode, use the Global **wsgw** command. The Global command creates the named Web Service Proxy if the proxy does not already exist. In this mode, define properties for the new or existing Web Service Proxy.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Service Proxy configuration mode.

aaa-policy

Assigns an AAA (Authentication, Authorization, Audit) Policy to the Web Service Proxy.

Syntax

aaa-policy *aaaPolicyName*

Parameters

aaaPolicyName
Specifies the name of an existing AAA Policy.

Guidelines

The results of the AAA Policy are cached, so it is not evaluated again when used in later processing by the request or response rule.

With Reliable Messaging the focus is on protecting the Reliable Messaging control messages, such as `CreateSequence` and `TerminateSequence`, it is also run on incoming Reliable Messaging data messages, with a `Sequence` header. This prevents unauthorized clients from using system resources by issuing `CreateSequence` requests, or from disrupting existing Reliable Messaging sequences with `CloseSequence` or `TerminateSequence` messages, or from falsely acknowledging messages with `SequenceAcknowledgement` messages.

Related Commands

wsrm

attachment-byte-count

Specifies the maximum size for an attached document in bytes.

Syntax

attachment-byte-count *bytes*

Parameters

bytes Specifies the maximum number of bytes to allow in any attachment. The default is 2000000000.

Guidelines

A value of 0 specifies that size limitations are not enforced by this proxy.

Attachments that exceed this size will result in a failure of the entire transaction.

Related Commands

gateway-parser-limits, **request-attachments**, **response-attachments**

---

attribute-count

Specifies the maximum number of attributes to associate with a given XML element.

Syntax

attribute-count *count*

Parameters

count Specifies the maximum number of attributes associated with an XML element. The default is 128.

Guidelines

If proxy-specific parser limitations are enabled by the **gateway-parser-limits** command, the attribute count assigned by **attribute-count** overrides any parser limit that might be inherited from the XML manager assigned to the Web Service Proxy.

Related Commands

attachment-byte-count, **element-depth**, **gateway-parser-limits**, **max-message-size**, **max-node-size**

---

autocreate-sources

Enables or disables a default HTTP Front Side Protocol Handler.

Syntax

autocreate-sources {**on** | **off**}

Parameters

on Enables the auto creation of an HTTP protocol handler that supports HTTP traffic only.

off (Default) Disables the auto creation of an HTTP protocol handler.

Guidelines

If front side traffic is conveyed by standard HTTP protocol, use this command to enable a default traffic handler. Otherwise, use the **front-protocol** command to assign one or more protocol-specific traffic handlers to the Web Service Proxy.

Related Commands

front-protocol

---

back-attachment-format

Specifies the attachment format output to backend servers.

Syntax

back-attachment-format {**dime** | **dynamic** | **mime**}

Parameters

dime Specifies that server attachments are DIME-encapsulated documents.

dynamic
Specifies that server attachments are deduced from front end content.

mime Specifies that server attachments are MIME-encapsulated documents.

Related Commands

front-attachment-format

---

back-persistent-timeout

Sets the inter-transaction timeout for proxy-to-server connections.

Syntax

back-persistent-timeout *timerValue*

Parameters

timerValue
Specifies the maximum inter-transaction idle time in seconds. Use an integer in the range of 0 through 7200. The default is 180. A value of 0 disables persistent connections.

Guidelines

The **back-persistent-timeout** command sets the inter-transaction timeout value, the proxy-specific maximum idle time to allow between the completion of a TCP transaction and the initiation of a new TCP transaction on the Web Services Proxy-to-server connection. If the specified idle timeout is exceeded, the connection is torn down.

An idle TCP connection could remain in the idle state for as long as 20 seconds after the expiration of the inter-transaction timer.

Related Commands

back-timeout, **front-persistent-timeout**, **front-timeout**, **persistent-connections**

back-timeout

Sets the intra-transaction timeout value.

Syntax

back-timeout *timerValue*

Parameters

timerValue
Specifies the maximum intra-transaction idle time in seconds. Use an integer in the range of 10 to 86400. The default is 120.

Related Commands

back-persistent-timeout, **front-timeout**, **front-persistent-timeout**, **persistent-connections**

Guidelines

The **back-timeout** command sets the intra-transaction timeout value, the proxy-specific maximum idle time allowed within a transaction on the Web Services proxy-to-server connection. This timer, for example, monitors the interval between sending the client request and receiving the server response, and idle time within the data transfer process. If the specified idle time is exceeded, the connection is torn down

backend-url

Specifies the URL to which all traffic to the static backend server is routed.

Syntax

backend-url *url*

Parameters

url Specifies a URL that fully identifies where all traffic is routed by default. This property can take one of the following general forms:

- `http://host:port/URI`
- `https://host:port/URI`
- `dpmq://queueManager/URI?RequestQueue=qName;ReplyQueue=qName...`
- `dptibems://server?RequestQueue=qName&ReplyQueue=qName...`
- `dpwasjms://server?RequestQueue=qName&ReplyQueue=qName...`

To use a load balancer, specify the name of an existing Load Balancer Group instead of the address-port pair in the URL.

Guidelines

HTTP and HTTPS
The service uses the HTTP or HTTPS protocol to connect the host at the specified port. The URL includes the URI. The URL or URI might be rewritten by other configuration options. With HTTPS, the configured SSL Proxy handles the SSL security negotiation.

MQ (dpmq)

The service uses the MQ protocol to connect to the requested Queue Manager. The URL includes the URI and query parameters for the request and reply queues. Optionally, the URL includes query parameters for transactionality and user identity.

The complete URL is as follows:

`dpmq://`*`queueManager`*`/`
`URI``?RequestQueue=`*`QName`*`;ReplyQueue=`*`QName`*`;Sync=`*`value`*`;PMO=`*`value`*

queueManager
Specifies the name of an existing MQ Queue Manager object.

URI Specifies the name of a service. Enter a string, such as `/SomeBank/services/checking` to be included in the URL.

`RequestQueue=`*QName*
Specifies the name of a queue that is managed by the specified Queue Manager. The DataPower service places requests on this queue.

`ReplyQueue=`*QName*;
Specifies the name of a queue that is managed by the specified Queue Manager. The DataPower service polls this queue for responses.

`Sync=`*value*
Optionally indicates whether the DataPower service implements transactionality. Use the binary `true` or `false`. If `true`, the service implements transactionality. When implemented, the service does not consider a request to be successfully delivered until it receives a corresponding response. If not specified, the default is `false`.

`PMO=`*value*
Optionally sets the `MQPMO.Options` field on the **MQPUT** call. The specified value is a cumulative value in decimal format of all acceptable options. If not specified, the default is `MQPMO_NO_SYNCPOINT` (decimal `4`, hexadecimal `0x00000004`.

Table 17. MQPMO.Options available for MQPUT calls

Put-Message option	Hexadecimal value	Decimal value
MQPMO_SYNCPOINT	0x00000002	2
MQPMO_NO_SYNCPOINT	0x00000004	4
MQPMO_NEW_MSG_ID	0x00000040	64
MQPMO_NEW_CORREL_ID	0x00000080	128
MQPMO_LOGICAL_ORDER	0x00008000	32728
MQPMO_NO_CONTEXT	0x00004000	16384
MQPMO_DEFAULT_CONTEXT	0x00000020	32
MQPMO_PASS_IDENTITY_CONTEXT	0x00000100	256
MQPMO_PASS_ALL_CONTEXT	0x00000200	512
MQPMO_SET_IDENTITY_CONTEXT	0x00000400	1024
MQPMO_SET_ALL_CONTEXT	0x00000800	2048
MQPMO_ALTERNATE_USER_AUTHORITY	0x00001000	4096

Table 17. MQPMO.Options available for MQPUT calls (continued)

Put-Message option	Hexadecimal value	Decimal value
MQPMO_FAIL_IF_QUIESCING	0x00002000	8192
MQPMO_NONE	0x00000000	0

TIBCO EMS (dptibems and dptibemss)

The service uses the TIBCO EMS protocol to connect to the requested TIBCO server. The URL includes the server and query parameters for the request and reply queues. Optionally, the URL includes the `Selector` query parameter. With `dptibemss`, the configured SSL Proxy handles the SSL security negotiation.

The complete URL is as follows:

```
dptibems:///server?RequestQueue=qName&ReplyQueue=qName
&Selector=expression
```

server Specifies the name of an existing TIBCO Server object.

`RequestQueue=`*qName*
Specifies the name of a queue or topic that is managed by the TIBCO server. Enter the queue or topic for the request.

`ReplyQueue=`*qName*
Specifies the name of a queue or topic that is managed by the TIBCO server. Enter the queue or topic for the reply.

`Selector=`*expression*
Optionally specifies an SQL92 conditional expression to identify *messages of interest*.

WebSphere JMS (dpwasjms and dpwasjmss)

The service uses the WebSphere JMS protocol to connect to the requested WebSphere JMS server. The URL includes the server and query parameters for the request and reply queues. Optionally, the URL includes the following query parameters:

- Request topic spaces
- Reply topic spaces
- Selector
- Timeout

With `dpwasjmss`, the configured SSL Proxy handles the SSL security negotiation.

The complete URL is as follows:

```
dpwasjms:///server?RequestQueue=qName&ReplyQueue=qName
&RequestTopicSpace=topicSpace&ResponseTopicSpace=topicSpace
&Selector=expression&TimeOut=timeout
```

server Specifies the name of an existing WebSphere JMS object.

`RequestQueue=`*qName*
Specifies the name of a queue or topic that is managed by the WebSphere JMS server. If using queues, specify the queue for the request. If using topic spaces, specify this query parameter without a value.

`ReplyQueue=`*qName*
Specifies the name of a queue or topic that is managed by the

WebSphere JMS server. If using queues, specify the queue for the reply. If using topic spaces, specify this query parameter without a value.

`RequestTopicSpace=`*topicSpace*
Optionally specifies a non-default request topic namespace.

`ResponseTopicSpace=`*topicSpace*
Optionally specifies a non-default reply topic namespace.

`Selector=`*expression*
Optionally specifies an SQL92 conditional expression to identify *messages of interest*..

`Timeout=`*timeout*
Optionally specifies a timeout value.

Related Commands

propagate-uri, **type**

Examples

- Sets the static backend URL to `http://10.10.10.2:3000/services`.

```
# backend-url http://10.10.10.2:3000/services
#
```

- Sets the static backend URL to `https://10.10.10.2:3000/services`. To support the SSL connection with the backend server, assigns the `clientssl` SSL Proxy Profile to provide the credentials for the secure connection.

```
# backend-url https://10.10.10.2:3000/services
# ssl clientssl
#
```

- Sets the static backend URL to `dpmq://BackEndQM/services?RequestQueue=Put_Q_Service;ReplyQueue=Get_Q_Service;UserName=aname;TimeOut=3000;Size=50000`.

```
# backend-url dpmq://BackEndQM/services?RequestQueue=
Put_Q_Service;ReplyQueue=Get_Q_Service;UserName=aname;TimeOut=3000;
Size=50000
#
```

backside-port-rewrite

Specifies a URL Rewrite Policy to modify the WSDL-port address in the source WSDL when the service is loaded.

Syntax

backside-port-rewrite *name*

Parameters

name Specifies the name of an existing URL Rewrite Policy to rewrite the backend WSDL-port address.

Guidelines

If no URL Rewrite Policy is specified, the default local address is the IP address of the appliance and the relative URI and the original port number are from the WSDL-port address that are specified in the source WSDL.

Related Commands

frontside-port-rewrite

Examples

- Identifies the URL Rewrite Policy `amendBackPort` as used to rewrite the WSDL-port address.

```
# backside-port-rewrite amendBackPort
#
```

chunked-uploads

Enables or disables the transmission of Content-Type Chunked Encoded documents to the backend server.

Syntax

Enables chunked encoding
chunked-uploads

Disables chunked encoding
no chunked-uploads

Guidelines

The Web Service Proxy can send an HTTP 1.1 request to the backend server. In this case, the body of the document can be delimited by either Content-Length or chunked encoding. All servers will understand how to interpret Content-Length, and many applications will fail to understand chunked. Therefore, Content-Length is generally used. However, doing so interferes with the ability of the DataPower appliance to fully stream.

If you must stream full documents toward the back side, this property should be turned on. Unlike all other HTTP 1.1 features that can be negotiated down at runtime, if necessary, you must know beforehand that the server with which you are communicating is compatible with RFC 2616. You might consider leaving this property disabled, the default state, and enabling it on a per-URL basis with the User Agent configuration.

client-principal

Specifies the name of the Kerberos client principal to decrypt encrypted requests.

Syntax

client-principal *principal*

Parameters

principal
Specifies the full name of a client principal.

Guidelines

The **client-principal** command the full name of a Kerberos client principal to decrypt encrypted requests. This command is meaningful when the service needs to decrypt automatically encrypted requests. Use when the encryption uses a Kerberos session key or uses a key that was derived from the session key.

This principal must be in the keytab identified by the **kerberos-keytab** command

Related Commands

kerberos-keytab

compression

Enables or disables `gzip` (GNU zip) compression negotiation between the Web Service Proxy and the backend server.

Syntax

Enables `gzip` negotiation
compression

Disables `gzip` negotiation
no compression

Guidelines

If enabled, the Web Service Proxy uses gzip to compress HTTP transmissions to the server only if the server indicates the ability to process compressed documents in the `Accept-Encoding` HTTP header field.

The proxy signals compression usage in the `Transfer-Encoding` HTTP header field.

GNU zip is described in RFC 1952, *GZIP File Format Specification, Version 4.3*.

Use the **no compression** command to disable compression negotiation between the Web Service Proxy and the application or Web server. Compression negotiation is disabled by default.

decrypt-key

Specifies the key to decrypt encrypted operations.

Syntax

decrypt-key *name*

Parameters

name Specifies the name of an existing Key object.

Guidelines

The **decrypt-key** command specifies the key to decrypt encrypted payloads, if any. The resulting decrypted node set will be passed to the processing rules.

default-param-namespace

Specifies the default XML namespace for stylesheet parameters that are defined without an explicit namespace.

Syntax

default-param-namespace *namespace*

Parameters

namespace
Identifies the default namespace. The default is the `http://www.datapower.com/param/config` namespace.

Related Commands

parameter

Examples

- Assigns the `http://www.megaCorp.com/XML/NS/external` as the default parameter namespace to the current proxy.

```
# default-param-namespace http://www.megaCorp.com/XML/NS/external
#
```

---

element-depth

Defines the maximum depth of element nesting in an XML document.

Syntax

element-depth *depth*

Parameters

depth Specifies the proxy-specific maximum depth of element nesting. The default is 512.

Guidelines

If proxy-specific parser limitations are enabled by the **gateway-parser-limits** command, the element depth assigned by the element depth overrides any parser limit that might be inherited from the XML manager assigned to the Web Service Proxy.

Related Commands

attachment-byte-count, **attribute-count**, **gateway-parser-limits**, **max-message-size**, **max-node-size**

---

endpoint-rewrite-policy

Assigns an Endpoint Rewrite Policy to the Web Service Proxy.

Syntax

endpoint-rewrite-policy *policyName*

Parameters

policyName
Specifies the name of an existing Endpoint Rewrite Policy.

Guidelines

The Endpoint Rewrite Policy determines the local, remote, and published endpoints that are used by the Web Service Proxy. If absence of an existing policy,

the proxy uses local and remote endpoint rules in conjunction with the **autocreate-sources** command to generate endpoints.

Use the **wsm-endpointrewrite** command to create a policy.

Related Commands

autocreate-sources, **wsm-endpointrewrite** (Global)

external-references

Defines the handling mode for input documents that contain external references.

Syntax

external-references {**allow** | **forbid** | **ignore**}

Parameters

allow Specifies that external references are allowed and resolved.

forbid (Default) Specifies that external references cause the XML parser to abort.

ignore Specifies that external references are ignored. External entities are replaced with an empty string.

follow-redirects

Enables or disables redirection on the current Web Service Proxy.

Syntax

Enables redirection
follow-redirects

Disables redirection
no follow-redirects

Guidelines

Some protocols generate redirects as part of the protocol, for example HTTP response code 302. If this property is enabled, the proxy will try and transparently resolve those redirects.

forbid-external-references (deprecated)

Comments

This command is deprecated. Use the **external-references** command.

front-attachment-format

Specifies the attachment format received from front end clients.

Syntax

front-attachment-format {**dime** | **dynamic** | **mime**}

Parameters

dime Specifies that client attachments are DIME-encapsulated documents.

dynamic
Specifies that the format of client attachments is deduced from document content.

mime Specifies that client attachments are MIME-encapsulated documents.

Related Commands

back-attachment-format

---

front-persistent-timeout

Sets the inter-transaction timeout for proxy-to-client connections.

Syntax

front-persistent-timeout *timerValue*

Parameters

timerValue
Specifies the maximum inter-transaction idle time in seconds. Use an integer in the range of 0 through 7200. The default is 180. A time value of 0 disables persistent connections.

Guidelines

The **front-persistent-timeout** command sets the inter-transaction timeout value. This value is the maximum idle time to allow between the completion of a TCP transaction and the initiation of a new TCP transaction on the proxy-to-client connection. If the specified idle timeout is exceeded, the connection is torn down.

An idle TCP connection can remain in the idle state for as long as 20 seconds after the expiration of the persistence timer.

Related Commands

back-persistent-timeout, **back-timeout**, **front-timeout**, **persistent-connections**

---

front-protocol

Assigns a specified protocol handler to the Web Service Proxy.

Syntax

front-protocol *name*

no front-protocol [*name*]

Parameters

name Identifies the client-side protocol handler.

Guidelines

Issue this command as many times as needed to add the desired front side protocol handlers to the current proxy.

A protocol handler provides a protocol-specific conduit that links external clients to the proxy.

Use the **no front-protocol** command to remove the assignment of one or all client-side protocol handlers from the Web Service Proxy.

Related Commands

source-http, **source-https**, **source-mq**

---

front-timeout

Sets the intra-transaction timeout value.

Syntax

front-timeout *timerValue*

Parameters

timerValue
Specifies the maximum intra-transaction idle time in seconds. Use an integer in the range of 10 to 86400. The default is 120.

Guidelines

Sets the intra-transaction timeout value, the maximum idle time allowed within a transaction on the proxy-to-client connection. This timer, for example, monitors idle time within the data transfer process. If the specified idle time is exceeded, the connection is torn down.

Related Commands

back-persistent-timeout, **back-timeout**, **front-persistent-timeout**, **persistent-connections**

---

frontside-port-rewrite

Specifies a URL Rewrite Policy used to modify the WSDL-port address specified in the source WSDL when the service is loaded.

Syntax

frontside-port-rewrite *name*

Parameters

name Identifies the URL Rewrite Policy used to rewrite the frontend WSDL-port address.

Guidelines

If no URL Rewrite Policy is specified, the default local address is the IP address of the appliance and the relative URI and the original port number from the WSDL-port address specified in the source WSDL.

Related Commands

backside-port-rewrite

Examples

- Uses the `amendClientPort` URL Rewrite Policy to rewrite the WSDL-port address.

```
# frontside-port-rewrite amendClientPort
#
```

fwcred

Assigns a Firewall Credentials list to the Web Service Proxy.

Syntax

fwcred [*fwCredName*]

no fwcred [*fwCredName*]

Parameters

fwCredName
Specifies the name of an existing Firewall Credentials List.

Guidelines

A Firewall Credentials list specifies which keys and certificates are available to support Web Service Proxy processing. In the absence of a Firewall Credentials List, all locally-stored key and certificates are available.

Use the **no fwcred** command to remove the assignment of a Firewall Credentials to a proxy. In the absence of a Firewall Credentials List, all assigned lists are removed.

Related Commands

fwcred (Crypto)

Examples

- Assigns the `standard-creds` Firewall Credentials List to the current proxy.

```
# fwcred standard-creds
#
```

- Removes the `standard-creds` Firewall Credentials List from the current proxy.

```
# no fwcred standard-creds
#
```

- Removes all Firewall Credentials List objects from the current proxy.

```
# no fwcred
#
```

gateway-parser-limits

Enables or disables proxy-specific parser limitations.

Syntax

Enables parser limitations
gateway-parser-limits

Disables proxy-specific parser limitations
no gateway-parser-limits

Guidelines

Parser limitations guard against denial-of-service attacks that use malicious XML documents that seek to exhaust system resources.

With proxy-specific parser limitations enabled, the values specified by the **attribute-count** and **element-depth** commands (Web Service Proxy) are used to evaluate incoming XML documents.

With proxy-specific parser limitations disabled (the default condition), parser limitations, if any, are derived from the XML Manager assigned to the Web Service Proxy.

Related Commands

attribute-count, **element-depth**

---

host-rewriting

Enables or disables host rewriting.

Syntax

host-rewriting {**on** | **off**}

Parameters

on (Default) The backend server receives a request that reflects the final route. The final route is determined by the DataPower service. The final route is not the one in the original, client submission.

off The backend server receives a request that reflects the information as it arrived at the DataPower appliance. Alternatively, use the **no host-rewriting** command.

Guidelines

Some protocols have distinct name-based elements that are separate from the URL to demultiplex. HTTP uses the `Host` header for this purposes.

Web servers that issue redirects might want to disable this feature. These web servers often depend on the `Host` header for the value of their redirect.

Related Commands

urlrewrite-policy, **propagate-uri**

---

http-client-ip-label

Identifies the HTTP header that contains the IP address of the calling client.

Syntax

http-client-ip-label *header*

no http-client-ip-label

Parameters

header Identifies the HTTP header that contains the IP address. The default is `X-Client-IP`.

Guidelines

The **http-client-ip-label** command identifies the HTTP header that contains the IP address of the calling client. When defined, the IP address of the calling client is read from this HTTP header. This IP address will then be used for monitoring and logging.

Use the **no http-client-ip-label** command to disable the reading of the HTTP header to identify the IP address of the calling client.

Examples

- Disables the reading of the HTTP header to identify the IP address of the calling client. Subsequently, enables this function to read the IP address from the `X-Forwarded-For` HTTP header for monitoring and logging.

```
# no http-client-ip-label
⋮
# http-client-ip-label X-Forwarded-For
#
```

http-server-version

Selects the HTTP version to use on the server-side (backend) connection of the current Web Service Proxy.

Syntax

http-server-version {**HTTP/1.0** | **HTTP/1.1** }

Parameters

HTTP/1.0
Indicates version 1.0 of the protocol.

HTTP/1.1
(Default) Indicates version 1.1 of the protocol.

include-content-type-encoding

Enables or disables the inclusion of character set encoding data in content-type headers that are generated by the Web Service Proxy.

Syntax

Enables the inclusion of character set encoding data
include-content-type-encoding

Disables the inclusion of character set encoding data

no include-content-type-encoding

Guidelines

Assume a UTF-8 encoded XML document. When enabled, the content-type header contains `text/xml; charset=UTF-8`. When disabled, the content-type header contains `text/xml`.

Use the **no include-content-type-encoding** command to disable the inclusion of character set encoding data. The default, is to disable the inclusion of character set encoding data.

inject

Injects a nonstandard HTTP header field into the packet stream between a Web Service Proxy and an HTTP client or server.

Syntax

inject {**front** | **back**} *field value*

no inject {**front** | **back**} *field*

Parameters

front Indicates that the packet stream is between a proxy and the HTTP client.

back Indicates that the packet stream is between a proxy and the HTTP server.

field Specifies the name of a nonstandard HTTP header field. This argument is case-sensitive.

value Specifies the value for the nonstandard HTTP header field. The value can be a character string or an integer. This argument is case-sensitive.

Guidelines

Use the **no inject** command to remove a previously-injected nonstandard HTTP header field.

Related Commands

suppress

kerberos-keytab

Specifies the keytab that contains the principals.

Syntax

kerberos-keytab *name*

Parameters

name Specifies the name of an existing Kerberos Keytab object.

Guidelines

The **kerberos-keytab** command specifies the keytab that contains the principals. The Web Service Proxy uses these principals to decrypt automatically encrypted

requests and responses. This keytab must contain the principals that are used by the **client-principal** command or by the **server-principal** command.

Related Commands

client-principal, **server-principal**

load-balancer-hash-header

Determines the value to use for hash calculations to load balance traffic to backend servers.

Syntax

Uses the value of an HTTP header

load-balancer-hash-header *header*

Uses the client IP address

no load-balancer-hash-header

Parameters

header Specifies the name of the HTTP header.

Guidelines

The **load-balancer-hash-header** command identifies the HTTP header to use for calculating the hash for load balancing traffic to the backend servers.

- When defined, the hash algorithm uses the value of the identified HTTP header.
- When not defined, the hash algorithm uses the IP address of the client.

This command is relevant only when the value defined by the **algorithm** command in Load Balancer configuration mode is **hash**.

Use the **no load-balancer-hash-header** command to disable the use of an HTTP header as the hash algorithm to use for load balancing.

Related Commands

algorithm

Examples

- Disables the use of an HTTP header for load balancing (uses the IP address to calculate the hash). Subsequently, enables load balancing traffic to the backend servers using a hash algorithm identified by the `X-Forwarded-For` HTTP header.

```
# no load-balancer-hash-header
⋮
# load-balancer-hash-header X-Forwarded-For
#
```

loop-detection

Enables or disables the loop detection algorithm.

Syntax

Enables the loop detection algorithm
loop-detection

Disables the loop detection algorithm
no loop-detection

Guidelines

Some protocols provide a loop detection mechanism that can be used to detect network loops. This is a good policy, but it can cause each proxy to be publicly recorded in the transmitted message, which might be considered an information leak.

By default, the loop detection algorithm is disabled.

---

max-message-size

Specifies the maximum size of an XML document to accept.

Syntax

max-message-size [*kilobytes*]

Parameters

kilobytes
Specifies the maximum number of kilobytes to scan before the document is considered malicious and dropped. Use an integer in the range of 0 through 2097151. The default is 0. A value of 0 specifies unlimited size.

Guidelines

The specified kilobyte count includes the contents or any external documents that are referenced by the incoming XML.

Related Commands

attachment-byte-count, **attribute-count**, **element-depth**, **max-node-size**

---

max-node-size

Specifies the maximum size of an XML node to accept.

Syntax

max-node-size [*bytes*]

Parameters

bytes Specifies the proxy-specific maximum number of bytes to allow in a single parsed XML node before the source XML document is considered malicious and dropped. The default is 0. This value indicates that there is no size limit.

Related Commands

attachment-byte-count, **attribute-count**, **element-depth**, **max-message-size**

mime-back-headers

Enables or disables the parsing of MIME headers in multipart messages that are sent over HTTP to and from backend servers.

Syntax

Enables MIME header parsing
mime-back-headers

Disables MIME header parsing
no mime-back-headers

Guidelines

The body of a message can sometimes contain MIME headers before any preamble and before the first MIME boundary in the body of the message. These MIME headers might contain important information that is not available in the protocol headers, such as the string that identifies the MIME boundary. If this command is enabled (the default state), the DataPower service processes these MIME headers.

When enabled and there are no MIME headers in the message, the DataPower service will try to parse the message by using the protocol header information, if available.

When disabled and MIME headers is in the body of the message, the MIME headers are considered part of the preamble. The MIME headers are not used to parse the message. If a protocol header (such as HTTP) indicates MIME boundaries, the DataPower service can parse and process individual attachments. If such information is not available, no attachments can be parsed from the body of the message.

Related Commands

mime-front-headers, **request-attachments**, **response-attachments**

mime-front-headers

Enables or disables the parsing of MIME headers in multipart messages sent over HTTP to and from clients.

Syntax

Enables MIME header parsing
mime-front-headers

Disables MIME header parsing
no mime-front-headers

Guidelines

The body of a message can sometimes contain MIME headers before any preamble and before the first MIME boundary in the body of the message. These MIME headers might contain important information that is not available in the protocol headers, such as the string identifying the MIME boundary. If this command is enabled (the default state), the DataPower service processes these MIME headers.

When enabled and there are no MIME headers in the message, the DataPower services continues will try to parse the message using the protocol header information, if available.

When disabled and MIME headers are in the body of the message, these MIME headers are considered part of the preamble. The MIME headers are not used to parse the message. If a protocol header (such as HTTP) indicates MIME boundaries, the DataPower service can parse and process individual attachments. If such information is not available, no attachments can be parsed from the body of the message.

Related Commands

mime-back-headers, **request-attachments**, **response-attachments**

monitor-count

Assigns or removes a Count Monitor.

Syntax

Assigns a Count Monitor
monitor-count *name*

Removes a Count Monitor
no monitor-count [*name*]

Parameters

name Specifies the name of a Count Monitor.

Guidelines

Use this command to add or to remove one or more Count Monitors.

Count Monitors watch for defined messaging events and increment counters each time event occurs. When a certain threshold is reached, the monitor can either write a notification to a log or block the service for a configured amount of time.

Related Commands

monitor-duration (Global), **monitor-service** (Global)

Examples

- Assigns the `wsgw-counter` Count Monitor to the current proxy.

```
# monitor-count wsgw-counter
#
```

- Removes the `wsgw-counter` Count from the current proxy.

```
# no monitor-count wsgw-counter
#
```

- Removes all Count Monitors from the current proxy.

```
# no monitor-count
#
```

monitor-duration

Assigns or removes a Duration Monitor.

Syntax

Assigns a Duration monitor
monitor-duration *name*

Removes a Duration monitor
no monitor-duration [*name*]

Parameters

name Specifies the name of a Duration Monitor.

Guidelines

The **monitor-duration** command adds or removes one or more Duration monitors.

Duration monitors watch for events that meet or exceed a configured duration. When a duration is met or exceeded, the monitor can write a notification to a log or block the service for a configured amount of time.

Related Commands

monitor-count (Global), **monitor-service** (Global)

Examples

- Assigns the `wsgw-duration` Duration monitor to the current proxy.

```
# monitor-duration wsgw-duration
#
```

- Removes the `wsgw-duration` Duration monitor from the current proxy.

```
# no monitor-duration wsgw-duration
#
```

- Removes all Duration monitors from the current proxy.

```
# no monitor-duration
#
```

monitor-processing-policy

Sets the behavior when a service has multiple monitors.

Syntax

monitor-processing-policy {**terminate-at-first-throttle** | **terminate-at-first-match**}

Parameters

terminate-at-first-throttle
(Default) Monitors will execute in the order in which they are listed. After any monitor either shapes (buffers to delay) or rejects a message, no further monitors will execute.

terminate-at-first-match
Monitors will execute in the order in which they are listed. After any monitor matches a message and takes any action at all, no further monitor will execute.

monitor-service

Assigns or removes a Service Level Monitor.

Syntax

Assigns a Service Level Monitor
monitor-service *name*

Removes a Service Level Monitor
no monitor-service [*name*]

Parameters

name Specifies the name of a Service Level Monitor.

Guidelines

Service Level Monitors watch Web Services endpoints. A Service Level Monitor collects statistics, establishes monitors (Count Monitors and Duration Monitors), and can take action when thresholds are met or exceeded.

Use this command to add or to remove one or more Service Level Monitors.

Use the **no monitor-service** command to remove the Service Level Monitor assignment.

Related Commands

monitor-count, **monitor-duration**

Examples

- Assigns the `wsgw-service` Service Level Monitor to the current proxy.

```
# monitor-service wsgw-service
#
```

- Removes the `wsgw-service` Service from the current proxy.

```
# no monitor-service wsgw-service
#
```

- Removes all Service Level Monitors from the current proxy.

```
# no monitor-service
#
```

---

operation-conformance

Sets the operational conformance policy.

Syntax

operation-conformance *policy wsdlComponentType* [*wsdlComponentValue* | *subscription*]

Parameters

policy Specifies the name of an existing Conformance Policy.

wsdlComponentType
Specifies the type of the WSDL component to match. Use one of the following values:

"" Disables all WSDL-based matching criteria. Disabling the matching criteria effectively creates a document-based Processing Policy.

all (Default) Matches all inputs, which Includes or excludes all WSDL component types (operation, port, service) to and from the match criteria.

operation
Matches when the identified operation is requested in the current transaction.

Matches `wsdl:binding/operation/@name` when formatted as `{bindingNamespace}name`, or matches `wsdl:service/wsdl:port` when formatted as `{serviceNamespace}port-name/operation-name`.

port Matches when the operation requested in the current transaction is included in the identified WSDL port.

Matches `wsdl:service/wsdl:port/@name` when formatted as `{serviceNamespace}port-name`.

service
Matches when the operation requested in the current transaction is included in the identified WSDL service.

Matches `wsdl:service/@name` when formatted as `{serviceNamespace}name`.

subscription
Matches an identified subscription key.

wsdl Matches when the operation requested in the current transaction is defined in the identified WSDL file.

wsdlComponentValue
Identifies the value of the WSDL-defined component. The value to specify depends on the identified WSDL component type.

- If **all**, specify double quotation marks (""). This combination eliminates the WSDL component from consideration.
- If **operation**, specifies the name of the WSDL operation. Use the wildcard character (*) to specify all operations.
- If **port**, specifies the name of the WSDL port. Use the wildcard character (*) to specify all ports.
- If **service**, specifies the name of the WSDL service. Use the wildcard character (*) to specify all services.
- If **subscription**, specify double quotation marks (""). Any specified value is ignored.
- If **wsdl**, specifies either a URL or the “local name” mnemonic that is assigned to the WSDL file.

subscription
Specifies the name of an existing Subscription object. The property is meaningful only when the value of the component type is **subscription**.

Guidelines

To create a new Conformance Policy, use the Global **conformancepolicy** command.

Related Commands

conformancepolicy

operation-policy-opt-out

Sets the WS-Policy subjects to ignore.

Syntax

operation-policy-opt-out *subjects wsdlComponentType* [*wsdlComponentValue* | *subscription*]

Parameters

subjects

Identifies which policy subjects should ignore any defined WS-Policy. Use any combination of the following keywords. To specify multiple keywords, enclose the entire selection in double quotation marks ("") characters and separate each keyword with the plus sign (+) character.

Service
Ignores the policy defined for the service policy subject.

Endpoint
Ignores the policy defined for the endpoint policy subject.

Operation
Ignores the policy defined for the operation policy subject.

MessageIn
Ignores the policy defined for the message policy subject for input messages.

MessageOut
Ignores the policy defined for the message policy subject for output messages.

wsdlComponentType

Specifies the type of the WSDL component to match. Use one of the following values:

"" Disables all WSDL-based matching criteria. Disabling the matching criteria effectively creates a document-based Processing Policy.

all (Default) Matches all inputs, which Includes or excludes all WSDL component types (operation, port, service) to and from the match criteria.

operation
Matches when the identified operation is requested in the current transaction.

Matches `wsdl:binding/operation/@name` when formatted as `{bindingNamespace}name`, or matches `wsdl:service/wsdl:port` when formatted as `{serviceNamespace}port-name/operation-name`.

port Matches when the operation requested in the current transaction is included in the identified WSDL port.

Matches `wsdl:service/wsdl:port/@name` when formatted as `{serviceNamespace}port-name`.

service
Matches when the operation requested in the current transaction is included in the identified WSDL service.

Matches `wsdl:service/@name` when formatted as `{serviceNamespace}name`.

subscription
Matches an identified subscription key.

wsdl Matches when the operation requested in the current transaction is defined in the identified WSDL file.

wsdlComponentValue
Identifies the value of the WSDL-defined component. The value to specify depends on the identified WSDL component type.

- If **all**, specify double quotation marks (""). This combination eliminates the WSDL component from consideration.
- If **operation**, specifies the name of the WSDL operation. Use the wildcard character (*) to specify all operations.
- If **port**, specifies the name of the WSDL port. Use the wildcard character (*) to specify all ports.
- If **service**, specifies the name of the WSDL service. Use the wildcard character (*) to specify all services.
- If **subscription**, specify double quotation marks (""). Any specified value is ignored.
- If **wsdl**, specifies either a URL or the "local name" mnemonic that is assigned to the WSDL file.

subscription
Specifies the name of an existing Subscription object. The property is meaningful only when the value of the component type is **subscription**.

---

operation-priority

Defines the priority for a specific web services operation.

Syntax

operation-priority *wsdlComponentType wsdlComponentValue subscription priority*

Parameters

wsdlComponentType
Specifies the type of the WSDL component to match. Use one of the following values:

"" Disables all WSDL-based matching criteria. Disabling the matching criteria effectively creates a document-based Processing Policy.

all (Default) Matches all inputs, which Includes or excludes all WSDL component types (operation, port, service) to and from the match criteria.

operation
Matches when the identified operation is requested in the current transaction.

Matches `wsdl:binding/operation/@name` when formatted as `{bindingNamespace}name`, or matches `wsdl:service/wsdl:port` when formatted as `{serviceNamespace}port-name/operation-name`.

port Matches when the operation requested in the current transaction is included in the identified WSDL port.

Matches `wsdl:service/wsdl:port/@name` when formatted as `{serviceNamespace}port-name`.

service
Matches when the operation requested in the current transaction is included in the identified WSDL service.

Matches `wsdl:service/@name` when formatted as `{serviceNamespace}name`.

subscription
Matches an identified subscription key.

wsdl Matches when the operation requested in the current transaction is defined in the identified WSDL file.

wsdlComponentValue
Identifies the value of the WSDL-defined component. The value to specify depends on the identified WSDL component type.

- If **all**, specify double quotation marks (""). This combination eliminates the WSDL component from consideration.
- If **operation**, specifies the name of the WSDL operation. Use the wildcard character (*) to specify all operations.
- If **port**, specifies the name of the WSDL port. Use the wildcard character (*) to specify all ports.
- If **service**, specifies the name of the WSDL service. Use the wildcard character (*) to specify all services.
- If **subscription**, specify double quotation marks (""). Any specified value is ignored.
- If **wsdl**, specifies either a URL or the "local name" mnemonic that is assigned to the WSDL file.

subscription
Specifies the name of an existing Subscription object. The property is meaningful only when the value of the component type is **subscription**.

priority
Assigns a priority for scheduling or for resource allocation. Use one of the following values:

low Receives below normal priority.

normal
(Default) Receives normal priority.

high Receives above normal priority.

Guidelines

The priority set by the **priority** command is overridden by this property.

Related Commands

priority

parameter

Assigns or removes a stylesheet parameter.

Syntax

parameter *name value*

no parameter [*name*]

Parameters

name Specifies the name of the parameter.

value Specifies the value for the parameter.

Guidelines

The following namespace declaration must be included in a style sheet to enable that style sheet to access parameter-value pairs that are defined by the **parameter** command.

```
xmlns:dpconfig="http://www.datapower.com/param/config
```

Related Commands

default-param-namespace

Examples

- Makes the `recipient` parameter with a value of `ALICE` and the `type` parameter with a value of `content` available to the current proxy. The default parameter namespace is used.

  ```
  # parameter recipient ALICE
  # parameter type content
  #
  ```

- Makes `foobar` parameter with a value of `value` available to the current proxy. `{http://www.example.com}` designates the parameter namespace.

  ```
  # parameter {http://www.example.com}foobar value
  #
  ```

- Makes the `foobar` parameter with a value of `value` available to the current proxy. `{}` designates no namespace.

  ```
  # parameter {}foobar value
  #
  ```

- Deletes the `recipient` parameter from the current proxy.

  ```
  # no parameter recipient
  #
  ```

- Deletes all parameters from the current proxy.

  ```
  # no parameter
  #
  ```

---

persistent-connections

Enables or disables HTTP 1.1 persistent connections on the proxy-to-server connection.

Syntax

Enables persistent connections
persistent-connections

Disables persistent connections

no persistent-connections

Guidelines

With persistent connections enabled, the default state for both HTTP 1.0 and HTTP 1.1, the DataPower service negotiates with the remote HTTP peer and establishes a persistent connection, if agreeable to the peer.

With persistent connections disabled, the DataPower service refuses to negotiate the establishment of persistent connections.

Use the **no persistent-connections** command to disable HTTP persistent connections.

Related Commands

back-persistent-timeout, **back-timeout**, **front-persistent-timeout**, **front-timeout**

---

policy-parameters

Defines the set of policy parameters for use by attached policy.

Syntax

policy-parameters *parameterSet wsdlComponentType* [*wsdlComponentValue* | *subscription*]

Parameters

parameterSet

Specifies the name of an existing Policy Parameters object.

wsdlComponentType

Specifies the type of the WSDL component to match. Use one of the following values:

"" Disables all WSDL-based matching criteria. Disabling the matching criteria effectively creates a document-based Processing Policy.

all (Default) Matches all inputs, which Includes or excludes all WSDL component types (operation, port, service) to and from the match criteria.

operation

Matches when the identified operation is requested in the current transaction.

Matches `wsdl:binding/operation/@name` when formatted as `{bindingNamespace}name`, or matches `wsdl:service/wsdl:port` when formatted as `{serviceNamespace}port-name/operation-name`.

port Matches when the operation requested in the current transaction is included in the identified WSDL port.

Matches `wsdl:service/wsdl:port/@name` when formatted as `{serviceNamespace}port-name`.

service

Matches when the operation requested in the current transaction is included in the identified WSDL service.

Matches `wsdl:service/@name` when formatted as `{serviceNamespace}name`.

subscription
Matches an identified subscription key.

wsdl Matches when the operation requested in the current transaction is defined in the identified WSDL file.

wsdlComponentValue
Identifies the value of the WSDL-defined component. The value to specify depends on the identified WSDL component type.

- If **all**, specify double quotation marks (""). This combination eliminates the WSDL component from consideration.
- If **operation**, specifies the name of the WSDL operation. Use the wildcard character (*) to specify all operations.
- If **port**, specifies the name of the WSDL port. Use the wildcard character (*) to specify all ports.
- If **service**, specifies the name of the WSDL service. Use the wildcard character (*) to specify all services.
- If **subscription**, specify double quotation marks (""). Any specified value is ignored.
- If **wsdl**, specifies either a URL or the "local name" mnemonic that is assigned to the WSDL file.

subscription
Specifies the name of an existing Subscription object. The property is meaningful only when the value of the component type is **subscription**.

Guidelines

To create a new Policy Parameters object, use the Global **policy-parameters** command.

Related Commands

policy-parameters

---

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal
(Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

Guidelines

The priority set by the **operation-priority** command overrides this setting.

Related Commands

operation-priority

process-http-errors

Indicates whether to processing errors from the backend server.

Syntax

process-http-errors {**on** | **off**}

Parameters

on (default) Ignores the error condition, and processes the response rule.

off Notices the error condition, and processes the error rule.

Guidelines

The **process-http-errors** command indicates whether to process errors from the backend server.

Depending on the protocol, the backend service might return a response code that indicates an error condition. For HTTP messages, the response from the backend server might include a response body that contains XML that provides more details about the error. For MQ messages, the response from the backend MQ server does not provide a response message.

propagate-uri

Enables or disables propagation of the local portion of the URI to the target server.

Syntax

Enables URI propagation
propagate-uri

Disables URI propagation
no propagate-uri

Guidelines

The **propagate-uri** command enables or disables the propagation of the client URL to the backend server.

Enabling URI propagation is meaningful in the following situations only:

- When the service is configured to use a static backend.
- When the service is configured to use a dynamic backend and dynamic routing is set with a route with style sheet (`route-action`) action in the processing policy. In this case, use the `dp:set-target()` extension element to define that target backend server.

  For the other dynamic routing options that are available with the `route-action` and `route-set` actions, the URI is absolute.

When enabled, the service rewrites the URI of the backend URL to the URI in the client request. If URI propagation is enabled and the client submits `http://host/service` and the backend URL is `http://server/listener`, the URL is rewritten to `http://server/service`.

Notes:

1. When enabled, any Matching Rule must match the rewritten URL.
2. Any action in the Processing Policy can change the URI that is sent to the backend server. The rewritten URI could override the intended effect of this setting.

Related Commands

urlrewrite-policy

---

query-param-namespace

Specifies a default namespace for parameters in a URL query string.

Syntax

query-param-namespace *namespace*

Parameters

namespace
Identifies the default namespace for query parameters. The default is the `http://www.datapower.com/param/query` namespace.

Related Commands

default-param-namespace, **parameter**

---

reliable-messaging

Controls reliable messaging properties.

Syntax

reliable-messaging *options deliveryAssuranceType wsdlComponentType* [*wsdlComponentValue* | *subscription*]

Parameters

options Identifies the options for reliable messaging. Use any combination of the following keywords. To specify multiple keywords, enclose the entire selection in double quotation marks ("") characters and separate each keyword with the plus sign (+) character.

Optional
Reliable Messaging is optional.

SequenceTransportSecurity
Reliable Messaging `Sequence` must be bound to underlying transport-level protocol.

InOrder
Reliable Messaging messages must be delivered in the same sequence as sent by the source

deliveryAssuranceType

Identifies the assurance type. Use the following keyword:

exactly-once
Messages must be delivered exactly one time.

wsdlComponentType

Specifies the type of the WSDL component to match. Use one of the following values:

"" Disables all WSDL-based matching criteria. Disabling the matching criteria effectively creates a document-based Processing Policy.

all (Default) Matches all inputs, which Includes or excludes all WSDL component types (operation, port, service) to and from the match criteria.

operation
Matches when the identified operation is requested in the current transaction.

Matches `wsdl:binding/operation/@name` when formatted as `{bindingNamespace}name`, or matches `wsdl:service/wsdl:port` when formatted as `{serviceNamespace}port-name/operation-name`.

port Matches when the operation requested in the current transaction is included in the identified WSDL port.

Matches `wsdl:service/wsdl:port/@name` when formatted as `{serviceNamespace}port-name`.

service
Matches when the operation requested in the current transaction is included in the identified WSDL service.

Matches `wsdl:service/@name` when formatted as `{serviceNamespace}name`.

subscription
Matches an identified subscription key.

wsdl Matches when the operation requested in the current transaction is defined in the identified WSDL file.

wsdlComponentValue

Identifies the value of the WSDL-defined component. The value to specify depends on the identified WSDL component type.

- If **all**, specify double quotation marks (""). This combination eliminates the WSDL component from consideration.
- If **operation**, specifies the name of the WSDL operation. Use the wildcard character (*) to specify all operations.
- If **port**, specifies the name of the WSDL port. Use the wildcard character (*) to specify all ports.
- If **service**, specifies the name of the WSDL service. Use the wildcard character (*) to specify all services.
- If **subscription**, specify double quotation marks (""). Any specified value is ignored.
- If **wsdl**, specifies either a URL or the “local name” mnemonic that is assigned to the WSDL file.

subscription
Specifies the name of an existing Subscription object. The property is meaningful only when the value of the component type is **subscription**.

remote-retry

Enables or disables the recovery policy for failed network connections.

Syntax

remote-retry off

remote-retry on *retry-interval reporting-interval total-retries*

Parameters

on Enables the recovery policy.

off (Default) Disables the recovery policy.

retry-interval
Specifies the number of seconds to wait after a failed connection attempt before performing another connection attempt. The minimum and default is 1.

reporting-interval
Specifies the number of seconds after a failed attempt to log a message at the `error` level instead of the default `debug` level. The minimum and default is 1.

total-retries
Specifies the total number of connection attempts to perform after the initial failed attempt. The minimum and default is 1.

Guidelines

The **remote-retry** command allows you to define a policy for failed network connection that occur while attempting to retrieve a WSDL file from a remote server. If you define this policy, specify the following information:

- The number of connection attempts to perform after the initial failed attempt
- The number of seconds to wait after a failed attempt to perform another connection attempt
- The number of seconds after a failed attempts to log a message at the `error` level instead of the `debug` level

If the retry interval is ten seconds and the reporting interval is four seconds, the system logs a message at the `error` level every four seconds until the next connection attempt. During the ten seconds between connection attempts, the system logged the `error` message twice. If this connection attempt fails, the system logs the message at the `debug` level and then four seconds later logs the message at the `error` level.

Note: If identical event detection for the Log Target is enabled and its suppression period is greater than the reporting interval, logging of the failure message at the `error` level is suppressed.

While the Web Service Proxy attempts to retrieve the WSDL file, the WSDL status is `Processing`. If the Web Service Proxy is unable to retrieve the file, the WSDL

status is `Error`. When the remote server becomes responsive, disable the Web Service Proxy and then re-enable it. Between each change in the administrative state, save the change to the running configuration.

Related Commands

admin-state, **event-detection** (Log Target), **exit**, **show ws-wsdl-status**, **suppression-period** (Log Target), **test tcp-connection** (initial login)

---

request-attachments

Specifies the processing mode for SOAP attachments in client requests.

Syntax

request-attachments *mode*

Parameters

mode Specifies one of the following keywords to indicate the processing mode for SOAP attachments:

allow Allows messages that contain attachments, and processes *needed* attachments. Needed attachments are buffered, but attachments that are not needed might be streamed directly to output.

Attachments are buffered when an action in the processing rule requests any of the following:

- Needed attachments
- All attachments in the package before the needed attachment
- All attachments in the package for a needed manifest
- All attachments in the package if the package does not contain the needed attachment

reject Rejects messages that contain attachments.

strip (Default) Removes attachments from the message before processing.

streaming
Allows messages that contain attachments in streaming mode, but provides limited processing. Messages in the form of a *SOAP message package*, which is a SOAP with Attachments message, are supported. Processing can be applied individually to each attachment. The appliance does not create a manifest of all attachments. Attachments must be accessed and processed in the order that they appear in the package.

unprocessed
Allows messages that contain attachments, but does not process attachments. This option permits the forwarding of messages that contain large attachments. The root part of the message, which typically contains a SOAP message, is subject to filter and transform actions. No processing of parts other than the root part is possible. Accompanying documents can be passed intact.

Guidelines

The **request-attachment** command specifies the processing mode for attachments in client requests (as defined in RFC 2387). This type of request is a compound

object that consists of several interrelated body parts and is the mechanism that is used to support the bundling of attachments in a *SOAP message package*, which is commonly referred to as a SOAP with Attachments message.

Meaningful only, if the value of the **request-type** command is **soap**.

Related Commands

request-type

---

request-type

Characterizes the client-originated traffic stream.

Syntax

request-type {**preprocessed** | **xml** | **soap** | **unprocessed**}

Parameters

preprocessed
Characterizes the traffic as non-XML that is not transformed by the proxy. The proxy can operate on other aspects of the message, such as determining the route or performing authentication and authorization.

xml Characterizes the traffic as raw (unencapsulated) XML.

soap (Default) Characterizes the traffic as SOAP.

unprocessed
Characterizes the traffic as non-XML traffic that is not transformed by the proxy.

Related Commands

response-type, **soap-schema-url**

---

response-attachments

Specifies the processing mode for SOAP attachments in server responses.

Syntax

response-attachments *mode*

Parameters

mode Specifies one of the following keywords to indicate the processing mode for SOAP attachments:

allow Allows messages that contain attachments, and processes *needed* attachments. Needed attachments are buffered, but attachments that are not needed might be streamed directly to output.

Attachments are buffered when an action in the processing rule requests any of the following:

- Needed attachments
- All attachments in the package before the needed attachment
- All attachments in the package for a needed manifest

- All attachments in the package if the package does not contain the needed attachment

reject Rejects messages that contain attachments.

strip (Default) Removes attachments from the message before processing.

streaming
Allows messages that contain attachments in streaming mode, but provides limited processing. Messages in the form of a *SOAP message package*, which is a SOAP with Attachments message, are supported. Processing can be applied individually to each attachment. The appliance does not create a manifest of all attachments. Attachments must be accessed and processed in the order that they appear in the package.

unprocessed
Allows messages that contain attachments, but does not process attachments. This option permits the forwarding of messages that contain large attachments. The root part of the message, which typically contains a SOAP message, is subject to filter and transform actions. No processing of parts other than the root part is possible. Accompanying documents can be passed intact.

Guidelines

The **response-attachment** command specifies the processing mode for attachments in server responses (as defined in RFC 2387). This type of request is a compound object that consists of several interrelated body parts and is the mechanism that is used to support the bundling of attachments in a *SOAP message package*, which is commonly referred to as a SOAP with Attachments message.

Meaningful only when the value of the **response-type** command is **soap**.

Related Commands

response-type

---

response-type

Characterizes the server-originated traffic stream.

Syntax

response-type {**preprocessed** | **xml** | **soap** | **unprocessed**}

Parameters

preprocessed
Characterizes the traffic as non-XML traffic that is not transformed by the proxy. The proxy can operate on other aspects of the message, such as determining the route or performing authentication and authorization.

xml Characterizes the traffic as raw (unencapsulated) XML.

soap Characterizes the traffic as SOAP.

unprocessed
Characterizes the traffic as non-XML traffic that is not transformed by the proxy.

Related Commands

request-type, **soap-schema-url**

---

root-part-not-first-action

Sets the action to take when the MIME message root part is not first.

Syntax

root-part-not-first-action {**abort** | **buffer** | **process-in-order**}

Parameters

abort Stops the transaction and return an error.

buffer Buffers attachments before the root part into memory. Then processes the root part, buffered attachments, and subsequent attachments.

process-in-order
(Default) Processes the attachments and root part in the order that they appear in the original message. All parts are still processed in streaming mode even though only attachments after the root will be streamed from the network.

Guidelines

When streaming MIME messages, specifies the action to take when the root part is not the first part of the message. If the root part must be first (for example to do conformance checking) and the action is set to **process-in-order**, the attachments up to the root will be buffered.

This command is meaningful only when the value of either the **request-attachments** or **response-attachments** command is **streaming.**

Related Commands

request-attachments, **response-attachments**

---

server-principal

Specifies the name of the Kerberos server principal to decrypt encrypted responses.

Syntax

server-principal *principal*

Parameters

principal
Specifies the full name of a client principal.

Guidelines

The **server-principal** command the full name of a Kerberos server principal to decrypt encrypted responses. This command is meaningful when the service needs to decrypt automatically encrypted requests. Use when the encryption uses a Kerberos session key or uses a key that was derived from the session key.

This principal must be in the keytab identified by the **kerberos-keytab** command

Related Commands

kerberos-keytab

soap-action-policy

Sets the SOAP Action Header Policy.

Syntax

soap-action-policy {**lax** | **strict** | **off**}

Parameters

lax (Default) An empty header or a header that contains the empty string from the client is considered a match. The client might quo te the SOAP header. For example, `SOAPAction:\"\"` is treated as a match.

strict The client must provide exactly the header specified in the WSDL. The client might quote the SOAP header.

off The SOAP header is ignored when issued by clients and is never compared to the WSDL.

soap-schema-url

Assigns a schema to validate incoming SOAP messages.

Syntax

soap-schema-url *schemaURL*

Parameters

schemaURL

Specifies the URL of the schema file to validate that SOAP messages conform to the SOAP schema. The default is the `schemas/soap-envelope.xsd` schema in the store: directory.

Guidelines

When a Web Service Proxy is in SOAP mode, either on the request or response side, it validates the incoming messages against a W3C Schema that defines a conforming SOAP message.

It is possible to customize which schema is used on a per-proxy basis by using this command. Different schemas can be used to accommodate nonstandard configurations or other special cases.

Related Commands

request-type response-type

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

no ssl *name*

Parameters

name Specifies the name of the SSL Proxy Profile.

Guidelines

The **ssl** command assigns or removes an SSL Proxy Profile to the current Web Service Proxy, thus enabling a secure communications line between the Web Service Proxy and the remote servers or clients.

An SSL Proxy Profile specifies the SSL operational mode (client, server, or two-way) and identifies the cryptographic resources (key, certificates, and cipher lists) available to the SSL proxy.

Use the **no ssl** command to remove the SSL Proxy Profile assignment. The proxy-server exchanges are accomplished over a nonsecure connection.

Related Commands

urlrewrite-policy, **xml-manager**

---

stream-output-to-back

Specifies server-facing streaming behavior.

Syntax

stream-output-to-back {**buffer-until-verification** | **stream-until-infraction**}

Parameters

buffer-until-verification
(Default) Specifies that the DataPower service buffer client request messages until all processing is verified as complete. After verification, transmits the request to the server.

stream-until-infraction
Specifies that the DataPower service begins sending client request messages to the server before all processing is complete, potentially increasing the speed. If an infraction is encountered, the DataPower service reverts to buffered behavior.

Guidelines

If the XML Manager that is assigned to this DataPower service has streaming enabled, select **stream-until-infraction** to be certain that the DataPower service streams messages end-to-end.

Related Commands

stream-output-to-front

stream-output-to-front

Specifies client-facing streaming behavior.

Syntax

stream-output-to-back {**buffer-until-verification** | **stream-until-infraction**}

Parameters

buffer-until-verification
(Default) Specifies that the DataPower service buffer server response messages until all processing is verified as complete. After verification, transmits the response to the client.

stream-until-infraction
Specifies that the DataPower service begins sending server response messages to the client before all processing is complete, potentially increasing the speed. If an infraction is encountered, the DataPower service reverts to buffered behavior.

Guidelines

If the XML Manager that is assigned to this DataPower service has streaming enabled, select **stream-until-infraction** to be certain that the DataPower service streams messages end-to-end.

Related Commands

stream-output-to-back

stylepolicy

Assigns a Processing Policy.

Syntax

stylepolicy *wsProcessingPolicyName*

Parameters

wsProcessingPolicyName
Specifies the name of a Processing Policy.

Guidelines

You do not need to specify a Processing Policy to configuring a Web Service Proxy. If absence, the Web Service Proxy uses processing instructions (if any) that are in the XML document. Such processing is independent of any processing that is performed by the Web Service Proxy.

Related Commands

xml-manager

suppress

Suppresses (deletes) HTTP header fields from the traffic stream between a Web Service Proxy and an HTTP client or server.

Syntax

suppress {**front** | **back**} *field*

no suppress {**front** | **back**} *field*

Parameters

front Indicates the traffic stream between a proxy and the HTTP client.

back Indicates the traffic stream between a proxy and the HTTP server.

field Specifies the name of an HTTP header field.

Guidelines

The name of a header field must be entered exactly as defined in sections 4.5, 5.3, 6.2, and 7.1 of RFC 2616.

Use the **no suppress** command to restore the standard HTTP header field to the packet stream.

Related Commands

host-rewriting inject

Examples

- Deletes the HTTP `Authorization` header from the traffic stream to the HTTP server.

```
# suppress back Authorization
#
```

- Restores the HTTP `Authorization` header to the traffic stream to the HTTP server.

```
# no suppress back Authorization
#
```

---

type

Specifies the type of Web Service Proxy.

Syntax

type {**dynamic-backend** | **static-backend**}

Parameters

dynamic-backend
Sets the proxy type to dynamic-backend. The address of the target server is programmatically extracted from the client request by using extension elements.

static-backend
(Default) Sets the proxy type to static-backend.

Guidelines

If the type is **static-backend**, use the **backend-url** command to identify the supported server.

Related Commands

backend-url

uddi-subscription

Adds or removes a UDDI subscription.

Syntax

uddi-subscription *uddiSubscriptionName*

no uddi-subscription

Parameters

uddiSubscriptionName
Specifies the name of an existing UDDI Subscription object.

Guidelines

Adds a UDDI Subscription to the current Web Service Proxy. A UDDI Subscription object in turn refers to a subscription to a UDDI Registry to obtain information (typically the information contained in a WSDL) about a web service that the current Web Service Proxy will virtualize.

You can add more than one UDDI Subscription to the current Proxy by repeating this command.

Use the **no uddi-subscription** command to remove the assignment of a UDDI Subscription from the current proxy.

Related Commands

uddi-subscription (Global)

Examples

- Adds the `ActivityEndpoint1` and `ActivityEndpoint2` UDDI Subscription objects to the current proxy.

```
# uddi-subscription ActivityEndpoint1
# uddi-subscription ActivityEndpoint2
#
```

- Removes the assignment of all UDDI Subscriptions from the current proxy.

```
# no uddi-subscription
#
```

urlrewrite-policy

Assigns or removes a URL Rewrite Policy.

Syntax

urlrewrite-policy *name*

no urlrewrite-policy [*name*]

Parameters

name Specifies the name of the URL Rewrite Policy.

Guidelines

A URL Rewrite Policy is not required to configure a Web Service Proxy.

Use the **no urlrewrite-policy** command to remove the assignment of a specific URL Rewrite Policy. Without a URL Rewrite Policy, removes all assigned URL Rewrite Policy objects from the proxy.

Related Commands

propagate-uri

Examples

- Assigns the `Rw1` URL Rewrite Policy to the current proxy.

```
# urlrewrite-policy Rw-1
#
```

- Removes the assignment of the `Rw1` URL Rewrite Policy from the current proxy.

```
# no urlrewrite-policy Rw-1
#
```

- Removes the assignment of all URL Rewrite Policy objects from the current proxy.

```
# no urlrewrite-policy
#
```

---

user-policy

Assigns a user-policy.

Syntax

user-policy *target-namespace WSDL-file WSDL-service WSDL-portType WSDL-binding WSDL-operation* [*behavior*]

no user-policy

Parameters

target-namespace
Specifies namespace criteria for policy selection. The target namespace is found in the WSDL definitions element. Enter the target namespace, or enter `*` to match any namespace.

WSDL-file
Identifies a specific WSDL file. Use the value of the *local-name* parameter as identified by the **wsdl** command, or enter `*` to identify all associated WSDL files.

WSDL-service
Specifies WSDL service criteria for policy selection. Specify a particular service with the `wsdl:definitions/wsdl:service/@name` form, or enter `*` for any `service`.

WSDL-portType
Specifies WSDL port criteria for policy selection. Specify a particular port with the `wsdl:definitions/wsdl:portType/@name` form, or enter `*` for any `portType`.

WSDL-binding
Specifies WSDL binding criteria for policy selection. Specify a particular binding with the `wsdl:definitions/wsdl:binding/@name` form, or enter `*` for any `binding`.

WSDL-operation
Specifies WSDL operation criteria for policy selection. Specify a particular operation with the `wsdl:definitions/wsdl:operation/@name` form, or enter `*` for any `operation`.

behavior
Identifies the availability and behavior options the user policy. Use any combination of the following keywords. To specify multiple keywords, enclose the entire selection in double quotation marks ("") characters and separate each keyword with the plus sign (+) character.

Enable
Enables requests for the operations and services included by this component level. A WSDL component, for example, includes all ports, services, and operation that are defined in the WSDL file.

Publish
Includes (selected) the component in any WSDL file that is published to external directories or returned in the WSDL file that is produced by the Web Service Proxy in response to requests by external clients. It is possible to enable an operation but not publish it until some other time. Also, it is possible to discontinue publishing an operation after a sunset period.

VerifyFaults
Validates fault messages against the schema that is contained in the corresponding WSDL file. Not all WSDL files contain schema information for faults. For this reason, the Web Service Proxy can be configured to allow fault messages to pass when no fault schema information is available. When selected and the WSDL file contains fault schema information, fault messages are checked against that schema and rejected if they do not validate.

VerifyHeaders
Validates SOAP headers. A WSDL file can contain schema information about SOAP headers.

NoRequestValidation
Does not validate request messages against the schema that is contained in the corresponding WSDL file.

NoResponseValidation
Does not validate response messages against the schema that is contained in the corresponding WSDL file.

SuppressFaultsElementsForRPCWrappers
Allows RPC operation wrapper for fault messages. This setting applies to the full selected WSDL files.

NoWSA
Ignores all Web Services Addressing (WS-Addressing) configuration settings.

NoWSRM
Ignores all Web Services Reliable Messaging (WS-Reliable Messaging) configuration settings.

Guidelines

Each WSDL Operation of the web service can have a user policy defined for that component.

Components are specified by the combination of target namespace, WSDL file, service, portType, binding, and operation.

Use the **no user-policy** command to delete a user-policy.

---

wsa-back-protocol

Specifies the Front Side Protocol Handler to receive asynchronous server responses and forward them to the original client.

Syntax

wsa-back-protocol *frontSideProtocolHandler*

Parameters

frontSideProtocolHandler
Specifies the name of an existing Front Side Protocol Handler.

Guidelines

The **wsa-back-protocol** command is relevant when the DataPower service provides asynchronous service (the **wsa-genstyle** command is **async**). In these topologies, this command specifies the Front Side Protocol Handler to receive the asynchronous response and forward that response to the original client.

This Front Side Protocol Handler can be overridden by the `var://context/__WSA_REQUEST/replyto` variable.

Related Commands

wsa-genstyle

---

wsa-default-faultto

Force the inclusion of the `FaultTo` element in Web Services Addressing (WS-Addressing) messages.

Syntax

wsa-default-faultto *faultURL*

Parameters

faultURL
Specifies the value of the `FaultTo` element.

Guidelines

The **wsa-default-faultto** command is relevant when the DataPower service provides service for WS-Addressing clients (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command ensures that all messages contain the WS-Addressing `FaultTo` element. This element identifies the recipient endpoint of fault messages.

Because the WS-Addressing specifications do not require the inclusion of the `FaultTo` element, the DataPower service might receive messages that do not contain a `FaultTo` element or that contain the element with no value.

When this happens, the DataPower service modifies the message to include a `FaultTo` element. This element contains the value specified by the *faultURL* argument.

If a default recipient endpoint of fault messages is not explicitly identified by this command, the DataPower service provides the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

Related Commands

wsa-mode

Examples

- Specifies `http://www.datapower.com/cs/internal/hdesk/` as the default recipient of `FaultTo` messages.

```
# wsa-default-faultto http://www.datapower.com/cs/internal/hdesk/
#
```

---

wsa-default-replyto

Force the inclusion of the `ReplyTo` element in Web Services Addressing (WS-Addressing) messages.

Syntax

wsa-default-replyto *replyURL*

Parameters

replyURL
Specifies the value of the `ReplyTo` element.

Guidelines

The **wsa-default-replyto** command is relevant when the DataPower service provides service for WS-Addressing clients (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command ensures that all messages contain the WS-Addressing `ReplyTo` element. This element identifies the recipient endpoint of a response message.

Because the WS-Addressing specifications do not require the inclusion of the `ReplyTo` element, the DataPower service might receive messages that do not contain a `ReplyTo` element or that contain the element without a value.

When this happens, the DataPower service modifies the message to include a `ReplyTo` element that contains the value specified by the *replyURL* argument.

If a default recipient endpoint of response messages is not explicitly identified by this command, the DataPower service provides the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

Related Commands

wsa-mode

Examples

- Specifies `http://www.customer.com/PO/inventoryReq/` as the default message recipient.

```
# wsa-default-replyto http://www.customer.com/PO/inventoryReq/
#
```

---

wsa-faultto-rewrite

Assigns or removes a URL Rewrite Policy that rewrites the contents of the Web Services Addressing (WS-Addressing) `FaultTo` element.

Syntax

Assign a URL Rewrite Policy
wsa-faultto-rewrite *urlRewritePolicy*

Removes a URL Rewrite Policy
no wsa-faultto-write

Parameters

urlRewritePolicy
Specifies the name of the URL Rewrite Policy.

Guidelines

The **wsa-faultto-write** command is relevant when the DataPower service provides service for WS-Addressing clients (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command modifies the contents of an incoming `FaultTo` element. This element identifies the recipient endpoint of fault messages.

Related Commands

absolute-rewrite (URL Rewrite Policy), **urlrewrite** (Global), **wsa-mode**, **wsa-replyto-rewrite**, **wsa-to-rewrite**

Examples

- Assigns the `wsaErrorHandler` URL Rewrite Policy to modify the contents of the `FaultTo` element.

```
# wsa-faultto-rewrite wsaErrorHandler
#
```

- Removes the assigned URL Rewrite Policy.

```
# no wsa-faultto-rewrite
#
```

wsa-force

Forces the inclusion of Web Services Addressing (WS-Addressing) headers into incoming, traditionally-addressed messages.

Syntax

wsa-force {**on** | **off**}

Parameters

on (Default) Forces the inclusion of WS-Addressing headers.

off Retains the traditional addressing headers.

Guidelines

The **was-force** command is relevant when the DataPower service provides service to users of WS-Addressing and users or traditionally-addressed messages (the **wsa-mode** command is **wsa2wsa**, **wsa2sync**, or **sync2wsa**). In these topologies, the DataPower service generally handles a mix of messages that use the WS-Addressing format and the traditional format.

Use this command to ensure that all messages use WS-Addressing. By default, **wsa-force** is disabled. When disabled, the DataPower service supports of mix of addressing styles.

When enabled, the DataPower service converts traditionally-addressed messages to the WS-Addressing format by adding the `reply-to` and `fault-to` headers to the traditionally-addressed message.

The `reply-to` header will contain the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

The `fault-to` header will contain the following default value:

```
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
```

These default values can be overridden with the **wsa-default-replyto** and **wsa-default-faultto** commands.

Related Commands

wsa-default-faultto, **wsa-default-replyto**, **wsa-strip-headers**

Examples

- Adds WS-Addressing headers to traditionally-addressed messages.

```
# wsa-force on
#
```

- Leaves traditionally-addressed message headers untouched.

```
# wsa-force off
#
```

or

```
# no wsa-force
#
```

wsa-genstyle

Specifies the request-response transmission model between the DataPower service and the target server.

Syntax

wsa-genstyle { **async** | **oob** | **sync** }

Parameters

async Identifies an asynchronous exchange pattern in which the server response is received over a different channel than the one used by the DataPower service to convey the client request.

oob Identifies an out-of-band exchange pattern in which the routing of the response to the original client is handled by the target server and does not pass through the DataPower service.

sync (Default) Identifies a synchronous exchange pattern in which the server response is received over the same channel used by the DataPower service to convey the client request.

Guidelines

If the request-response transmission model is **async**, use the **wsa-back-protocol** command to identify the Front Side Protocol Handler to convey asynchronous server responses to the original requesting clients. For the asynchronous model, use the **wsa-timeout** command to specify the maximum time allowed for a server response.

If the request-response transmission model is **oob**, ensure that the Web Server Proxy preserves explicit (non-anonymous), client-originated values for the `ReplyTo` and `FaultTo` elements and passes these values intact to the server.

Related Commands

wsa-back-protocol, **wsa-http-async-response-code**, **wsa-mode**, **wsa-timeout**

wsa-http-async-response-code

Specifies the HTTP response code to send to a client appliance before transmitting the actual asynchronous server response.

Syntax

wsa-http-async-response-code *responseCodeValue*

Parameters

responseCodeValue
Specifies the HTTP response code to close the original client channel. Use a value in the range of 200 through 599. The default is 204.

Guidelines

If the server response to an HTTP client request is asynchronous, the DataPower service must close the original HTTP channel with a valid response code. After the channel is closed, the DataPower service forwards the server-generated response or fault message to the client over a new channel.

Related Commands

wsa-genstyle

Examples

- Specifies an HTTP Response Code of 210 to close an open HTTP client channel.

```
# wsa-http-response-code 210
#
```

---

wsa-mode

Specifies the Web Services Addressing (WS-Addressing) support.

Syntax

wsa-mode {**sync2sync** | **sync2wsa** | **wsa2sync** | **wsa2wsa**}

Parameters

sync2sync
(Default) Disables WS-Addressing support. Both hosts (clients and servers) that access the DataPower service will use traditional addressing.

sync2wsa
Specifies that the DataPower service is mediating between hosts that employ traditional addressing and servers that support WS-Addressing.

wsa2sync
Specifies that the DataPower service is mediating between hosts that support WS-Addressing and servers that employ traditional addressing.

wsa2wsa
Specifies that the DataPower service is mediating between hosts and servers that support WS-Addressing.

Guidelines

The **wsa-mode** command specifies the WS-Addressing support that the DataPower service provides. The level of support is determined by the WS-Addressing capabilities of the associated clients and servers. Support for any particular level of WS-Addressing does not preclude simultaneous support for traditional addressing formats.

- When operating in **sync2wsa** mode, the DataPower service, under user control, can:
  - Insert WS-Addressing headers into the traditionally addressed original client request. The default behavior is to retain the original addressing format.
  - Strip the WS-Addressing headers from any server-generated response before forwarding the response to the original client. The default behavior is to strip the WS-Addressing headers.
  - Process synchronous or asynchronous server responses of either the `ReplyTo` (a standard response to a client request) or `FaultTo` (reporting an error condition) variety.

  A synchronous response is received over the same connection that carried the client request to the server.

  An asynchronous response is received over a different connection that carried the client request to the server, and requires the DataPower service to maintain state information associating the received response with an outstanding request.

- When operating in **wsa2sync** mode, the DataPower service, under user control, can:
  - Insert WS-Addressing headers into the traditionally addressed server response. The default behavior is to retain the original addressing format.
  - Strip the WS-Addressing headers from any client-generated request before forwarding the request to the target server. The default behavior is to strip the WS-Addressing headers.
  - Rewrite the contents of, or supply default values, for client-generated `ReplyTo` and `FaultTo` elements to specify the destinations of these response types.
  - Rewrite the contents of the client-generated `To` element to specify where the client request is routed.
  - Process synchronous or asynchronous server responses of either the `ReplyTo` (a standard response to a client request) or `FaultToA` (reporting an error condition) variety.

  A synchronous response is received over the same connection that carried the client request to the server.

  An asynchronous response is received over a different connection that carried the client request to the server, and requires the DataPower service to maintain state information associating the received response with an outstanding request.
- When operating in **wsa2wsa** mode, the DataPower service under user control, can:
  - Insert WS-Addressing headers into the traditionally addressed server response. The default behavior is to retain the original addressing format.
  - Strip the WS-Addressing headers from any client-generated request before forwarding the request to the target server. The default behavior is to strip the WS-Addressing headers.
  - Rewrite the contents of, or supply default values, for client-generated `ReplyTo` and `FaultTo` elements to specify the destinations of these response types.
  - Rewrite the contents of the client-generated `To` element to specify where the client request is routed.
  - Support three response modes for either the `ReplyTo` (a standard response to a client request) or `FaultTo` (reporting an error condition) variety.

  A synchronous response is received over the same connection that carried the client request to the server.

  An asynchronous response is received over a different connection that carried the client request to the server, and requires the DataPower service to maintain state information associating the received response with an outstanding request.

  An out-of-band response is handled by the target server and does not pass through the DataPower service. An out-of-band response requires explicit (non-anonymous) client-originated `ReplyTo` and `FaultTo` element values that are preserved by the DataPower service and passed to the server.

Related Commands

wsa-back-protocol, **wsa-force**, **wsa-genstyle**, **wsa-timeout**, **wsa-strip-headers**

Examples

- Specifies `sync2wsa` mode, indicating that the DataPower service is mediating between hosts that employ traditional addressing and servers that support WS-Addressing.

```
# wsa-mode sync2wsa
#
```

wsa-replyto-rewrite

Identifies the URL Rewrite Policy to rewrite the contents of the Web Services Addressing (WS-Addressing) `ReplyTo` element.

Syntax

wsa-replyto-rewrite *urlRewritePolicy*

no wsa-replyto-rewrite

Parameters

urlRewritePolicy
Specifies the name of the URL Rewrite Policy.

Guidelines

The **wsa-replyto-rewrite** command is relevant when the DataPower service provides service for WS-Addressing client users (the **wsa-mode** command is **wsa2sync** or **wsa2wsa**). In these topologies, this command modifies the contents of an incoming `ReplyTo` element. This element identifies the recipient endpoint of response messages.

Related Commands

absolute-rewrite, **urlrewrite**, **wsa-mode**, **wsa-faultto-rewrite**, **wsa-to-rewrite**

Examples

- Identifies `wsaResponseHandler` as the URL Rewrite Policy used to modify the contents of the `ReplyTo` element.

```
# wsa-replyto-rewrite wsaResponseHandler
#
```

- Removes the assignment of `wsaResponseHandler` as the URL Rewrite Policy used to modify the contents of the `ReplyTo` element.

```
# no wsa-replyto-rewrite
#
```

wsa-strip-headers

Removes all Web Services Addressing (WS-Addressing) headers from an incoming message before forwarding the message to the recipient.

Syntax

wsa-strip-headers {**on** | **off**}

Parameters

on (Default) Enables the deletion of WS-Addressing headers from an incoming message.

off Disables the deletion of WS-Addressing headers from an incoming message.

Guidelines

This command is relevant when the DataPower service is positioned between users of WS-Addressing and a nonusers; that is when the WS-Addressing mode, as specified by the **wsa-mode** command, is **wsa2sync** or **sync2wsa**.

Note: WS-Reliable Messaging requires the termination of WS-Addressing sequences. Changing the default value can break interoperability.

Related Commands

wsa-force, **wsa-mode**

Examples

- Changes the default state. Retains all WS-Addressing headers contained in incoming messages.

```
# wsa-strip-headers off
#
```

or

```
# no wsa-strip-headers
#
```

- Restores the default state. Deletes all WS-Addressing headers contained in incoming messages.

```
# wsa-strip-headers on
#
```

or

```
# wsa-strip-headers
#
```

---

wsa-timeout

Specifies the asynchronous timeout value.

Syntax

wsa-timeout *timerValue*

Parameters

timerValue
Specifies the maximum wait period in seconds. Use an integer in the range of 1 through 4000000. The default is 120.

Guidelines

The **wsa-timeout** command specifies the maximum period of time to wait for an asynchronous response, before abandoning the transaction.

This timeout value can be overridden by the `var://service/wsa/timeout` variable.

Related Commands

wsa-mode

Examples

- Specifies a maximum pause of 1 minute while waiting for an asynchronous response.

```
# wsa-timeout 60
#
```

wsa-to-rewrite

Assigns or removes a URL Rewrite Policy that rewrites the contents of the Web Services Addressing (WS-Addressing) `To` element.

Syntax

wsa-to-rewrite *urlRewritePolicy*

no wsa-to-rewrite

Parameters

urlRewritePolicy
Specifies the name of an existing URL Rewrite Policy.

Guidelines

The **wsa-to-rewrite** command modifies the contents of an incoming `To` element that identifies the message destination. This command is relevant when the DataPower service provides service for clients that support WS-Addressing formats. In these cases, the WS-Addressing mode, as specified by the **wsa-mode** command, is **wsa2sync** or **wsa2wsa**.

Related Commands

wsa-mode

wsdl

Assigns or removes a source WSDL file.

Syntax

wsdl *source-location local-name* [*policy-attachment*]

no wsdl *source-location*

Parameters

source-location
Specifies the exact location (URL) of the WSDL file. The file can be stored on the appliance or on a remote server (for example, `local:/// searchservice.wsdl`).

local-name
Specifies a mnemonic for the WSDL file. The mnemonic can be the file name (for example, `searchservice.wsdl`) or an alias (for example, `search`). The value for the parameter should be the name of the *WSDL-file* parameter as identified by the **user-policy** command.

policy-attachment
Specifies the name of an existing Policy Attachment object. Policy

attachments associate WS-Policy with the different types of policy subjects (service, endpoint, operation, message) in the WSDL file.

Guidelines

The **wsdl** command associates a WSDL file with the Web Service Proxy. The WSDL file defines the Web services that the Web Service Proxy supports.

Use the **no wsdl** command to remove a WSDL file from the Web Service Proxy.

Related Commands

user-policy

Examples

- Associates the `accountQuery.wsdl` WSDL file in the `local:` directory, defines this source WSDL file with the `getBalance` alias, and associates the `WSSec-UNT` Policy Attachment object.

```
# wsdl local:///accountQuery.wsdl getBalance WSSec-UNT
#
```

- Associates the `accountQuery.wsdl` WSDL file in the `local:` directory, defines this source WSDL file with the `getBalance` alias, but does not associates a Policy Attachment object.

```
# wsdl local:///accountQuery.wsdl getBalance
#
```

- Removes the association of the `accountQuery.wsdl` WSDL file in the `local:` directory.

```
# no wsdl local:///accountQuery.wsdl
#
```

---

wsdl-cache-policy

Establishes a WSDL caching policy file with the current Web Service Proxy.

Syntax

wsdl-cache-policy *wsdlLocation ttlValue*

Parameters

wsdlLocation
Specifies the location of one or more WSDL files.

ttlValue
Specifies the number of seconds before the proxy refreshes the WSDL files.

Guidelines

The proxy can automatically refresh one or more of the WSDL files on which the proxy is based. WSDL files that match the specified location (*wsdlLocation*) are automatically refreshed after the time-to-live (TTL) value is reached.

A refresh can cause the proxy to re-configure itself in accordance with the new state of the WSDL files.

Examples

- Creates a caching policy where the WSDL files at `http://server/banking/*` are cached every 24 hours.

```
# wsdl wsdl-cache-policy http://server/banking/* 86400
#
```

wsrr-subscription

Obtains web services through a WSRR subscription.

Syntax

wsrr-subscription *wsrrSubscriptionName*

no wsrr-subscription

Parameters

wsrrSubscriptionName
Specifies the name of an existing WSSR subscription object.

Guidelines

Adds a WSRR Subscription to the current Web Service Proxy. A WSRR Subscription object in turn refers to a subscription in a WSRR Registry to obtain information (typically the information contained in a WSDL) about a web service that the current Web Service Proxy will virtualize.

Configuration data can be contained in multiple WSDL files or concepts. Use this command as often as necessary to identify all required configuration resources.

Use the **no wsrr-subscription** command to remove the assignment of all WSRR subscriptions.

Related Commands

uddi-subscription

wsrm

Enable or disables Web Services Reliable Messaging.

Syntax

wsrm {**on** | **off**}

Parameters

on Enables Reliable Messaging.

off (Default) Disables Reliable Messaging.

Related Commands

wsrm-aaapolicy, **wsrm-destination-accept-create-sequence**, **wsrm-destination-accept-offers**, **wsrm-destination-inorder**, **wsrm-destination-maximum-inorder-queue-length**, **wsrm-destination-maximum-sequences**, **wsrm-request-force**, **wsrm-response-force**, **wsrm-sequence-expiration**, **wsrm-source-back-acks-to**, **wsrm-source-exponential-backoff**, **wsrm-source-front-acks-to**,

wsrm-source-inactivity-close-interval, **wsrm-source-make-offer**, **wsrm-source-maximum-queue-length**, **wsrm-source-maximum-sequences**, **wsrm-source-request-ack-count**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**, **wsrm-source-sequence-ssl**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

wsrm-aaapolicy

Assigns an AAA Policy.

Syntax

wsrm-aaapolicy *name*

Parameters

name Specifies the name of an existing AAA Policy.

Guidelines

Use the **wsrm-aaapolicy** command to assign an AAA Policy to perform authentication of incoming Reliable Messaging messages. This AAA Policy can be the same one that is used in later processing by the request or response rule. The results are cached, so it is not evaluated again.

While this is focused on protecting the Reliable Messaging control messages, such as `CreateSequence` and `TerminateSequence`, it is also run on incoming Reliable Messaging data messages, with a `Sequence` header. This prevents unauthorized clients from using system resources by issuing `CreateSequence` requests, or from disrupting existing Reliable Messaging sequences with `CloseSequence` or `TerminateSequence` messages, or from falsely acknowledging messages with `SequenceAcknowledgement` messages.

To create an AAA Policy, use the Global **aaapolicy** command.

Related Commands

aaapolicy (global), **wsrm**

wsrm-destination-accept-create-sequence

Indicates whether to accept incoming `CreateSequence` SOAP requests and create a Reliable Messaging destination when one is received.

Syntax

wsrm-destination-accept-create-sequence {**on** | **off**}

Parameters

on (Default) Enables this feature. If enabled, both the client and the server can use Reliable Messaging to send messages to this DataPower service.

off Disables this feature. If disabled, the client cannot use Reliable Messaging to communicate with this DataPower service. If disabled, the only way that a Reliable Messaging destination can be created on this DataPower service is when the Reliable Messaging source is configured to make offers. In this case an `Offer` and `Accept` can create a Reliable Messaging destination for the server to send Reliable Messaging messages to the client.

Related Commands

wsrm

---

wsrm-destination-accept-offers

Indicates whether to accept offers for two-way Reliable Messaging in `CreateSequence` SOAP requests.

Syntax

wsrm-destination-accept-offers { **on** | **off**}

Parameters

on Accepts two-way requests.

off (Default) Does not accept two-way requests.

Guidelines

The **wsrm-destination-accept-offers** command indicates whether to accept offers for two-way Reliable Messaging in `CreateSequence` SOAP requests. If the request includes an offer, the creation of a Reliable Messaging destination creates a Reliable Messaging source to send responses to the client.

Related Commands

wsrm, **wsrm-source-exponential-backoff**, **wsrm-source-inactivity-close-interval**, **wsrm-source-maximum-queue-length**, **wsrm-source-request-ack-count**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

---

wsrm-destination-inorder

Indicates whether to enable `InOrder` delivery assurance for Reliable Messaging destinations

Syntax

wsrm-destination-inorder {**on** | **off**}

Parameters

on Enables `InOrder` and `ExactlyOnce` delivery assurance.

off (Default) Enables `ExactlyOnce` delivery assurance only.

Guidelines

The **wsrm-destination-inorder** command indicates whether to enable `InOrder` delivery assurance for Reliable Messaging destinations in addition to the standard `ExactlyOnce` delivery assurance. No messages will be passed from the receive queue for further processing unless their sequence number as assigned by the client is one greater than the last one that was processed. `InOrder` delivery assurance increases memory and resource utilization by the Reliable Messaging destination.

Related Commands

wsrm, **wsrm-destination-maximum-inorder-queue-length**

wsrm-destination-maximum-inorder-queue-length

Specifies the maximum number of messages held in the queue.

Syntax

wsrm-destination-maximum-inorder-queue-length *numberOfMessages*

Parameters

numberOfMessages
Specifies the maximum number of messages beyond the gap. Use an integer in the range of 1 through 256. The default is 10.

Guidelines

The **wsrm-destination-maximum-inorder-queue-length** command specifies the maximum number of messages held in the Reliable Messaging queue beyond a gap in the received sequence numbers.

This property controls memory utilization.

Related Commands

wsrm, **wsrm-destination-inorder**

wsrm-destination-maximum-sequences

Sets a limit on the maximum number of simultaneously active sequences to Reliable Messaging destinations.

Syntax

wsrm-destination-maximum-sequences *maximumSequences*

Parameters

maximumSequences
Specifies the maximum number of simultaneous active sequences. The default is 400.

Guidelines

The **wsrm-destination-maximum-sequences** command sets a limit on the maximum number of simultaneously active sequences to Reliable Messaging destinations of this DataPower service. Attempts by clients to create sequences in excess of this limit result in a SOAP Faults. This property controls memory resource utilization.

Related Commands

wsrm

wsrm-request-force

Indicates whether to require Reliable Messaging for all SOAP messages that request rules process.

Syntax

wsrm-request-force {**on** | **off**}

Parameters

on Requires Reliable Messaging for all requests.

off (Default) Does not require Reliable Messaging for all requests.

Guidelines

The xxx command indicates whether to require the use of Reliable Messaging for all SOAP messages that request rules process. The client must establish a sequence with a `CreateSequence` SOAP call and must include a `Sequence` in each SOAP header. Any SOAP message without a `Sequence` results in a SOAP fault.

Related Commands

wsrm

---

wsrm-response-force

Indicates whether to require Reliable Messaging for all SOAP messages that response rules process.

Syntax

wsrm-response-force {**on** | **off**}

Parameters

on Requires Reliable Messaging for all responses.

off (Default) Does not require Reliable Messaging for all responses.

Guidelines

The **wsrm-response-force** command indicates whether to require the use of Reliable Messaging for all SOAP messages that response rules process. Any SOAP message without a `Sequence` results in a SOAP fault.

Note: When WS-Addressing is in use, SOAP messages without a WS-Addressing `RelatesTo` SOAP Header are processed by the request rule, not the response rule, even if the message come from the backend server.

Related Commands

wsrm

---

wsrm-sequence-expiration

Sets the target expiration interval in seconds for all Reliable Messaging sequences.

Syntax

wsrm-sequence-expiration *lifetime*

Parameters

lifetime Specifies the lifetime in seconds. The default is 3600.

Guidelines

If an incoming `CreateSequence` SOAP message has an `Expires`lifetime that is longer than this value, the value in the `SequenceResponse` SOAP message is reduced to this value. The same process applies to the `Expires`lifetime in any accepted `Offer` in an incoming `CreateSequence` and for the requested `Expires` value in any `CreateSequence` SOAP call that is made to the client or server from a Reliable Messaging source. This implementation never requests or accepts a non-expiring sequence (a value of `PT0S` that represents zero seconds).

Related Commands

wsrm

---

wsrm-source-back-acks-to

Specifies the name of the Front Side Protocol Handler to receive responses from the server.

Syntax

wsrm-source-back-acks-to *handler*

Parameters

handler
Specifies the name of an existing Front Side Protocol Handler.

Guidelines

The **wsrm-source-back-acks-to** command identifies the Front Side Protocol Handler to receive the asynchronous Reliable Messaging `SequenceAcknowledgement` SOAP responses from the server. The Front Side Protocol Handler must be associated with the same DataPower service where the corresponding Reliable Messaging sequence is occurring.

This property controls whether the backside Reliable Messaging source uses a unique URL to receive asynchronous `Acks` from the server Reliable Messaging destination, or whether `Acks` are sent synchronously in future responses to the backside server.

- With a specified Front Side Protocol Handler and the response process causes a `CreateSequence` SOAP message to be sent, the `AcksTo` element of the `CreateSequence` SOAP message will be set to the URL that is specified in back `AcksTo`.
- Without a Front Side Protocol Handler, the `AcksTo` element has the value `http://www.w3.org/2005/08/addressing/anonymous`, which indicates synchronous `Acks`.

Related Commands

wsrm

wsrm-source-exponential-backoff

Indicates whether to use the exponential back off.

Syntax

wsrm-source-exponential-backoff {**on** | **off**}

Parameters

on (Default) Uses the exponential back off to increase the interval between retransmissions. The value of the **wsrm-source-retransmission-interval** command sets with the initial timeout.

off Does not use the exponential back off to increase the interval between retransmissions.

Guidelines

The **wsrm-source-exponential-backoff** command indicates whether to use the exponential back off to increase the interval between retransmissions on unacknowledged messages by a Reliable Messaging source.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**, **wsrm-source-retransmission-interval**

wsrm-source-front-acks-to

Identifies the Front Side Protocol Handler to receive response for the client.

Syntax

wsrm-source-front-acks-to *handler*

Parameters

handler
Specifies the name of an existing Front Side Protocol Handler.

Guidelines

The **wsrm-source-front-acks-to** command identifies the Front Side Protocol Handler to receive the asynchronous Reliable Messaging `SequenceAcknowledgement` SOAP responses from the client. The Front Side Protocol Handler must be associated with the same DataPower service where the corresponding Reliable Messaging sequence is occurring.

This property controls whether a front-side Reliable Messaging source uses a unique URL to receive asynchronous `Acks` from the client Reliable Messaging destination or whether `Acks` are sent synchronously in future requests to the front-side client.

- With a specified Front Side Protocol Handler and the client includes an `Offer` in a `CreateSequence` SOAP message sent due to response processing, there will be a non-anonymous URL specified in the `AcksTo` element of the `Accept` element of the `CreateSequenceResponse` SOAP reply.

- With a specified Front Side Protocol Handler and the front-side sends a `CreateSequence` SOAP message to establish a reliable back channel, there will be a non-anonymous URL specified in the `AcksTo` element of the `CreateSequence` SOAP request.
- Without a Front Side Protocol Handler, the `AcksTo` elements has the value `http://www.w3.org/2005/08/addressing/anonymous`, which indicates synchronous `Acks`.

Related Commands

wsrm

---

wsrm-source-inactivity-close-interval

Specifies the duration to wait before closing the sequence.

Syntax

wsrm-source-inactivity-close-interval *duration*

Parameters

duration

Specifies the duration to wait in seconds. Use an integer in the range of 1 through 3600. The default is 3.

Guidelines

The **wsrm-source-inactivity-close-interval** command specifies the duration in second that a Reliable Messaging source waits for an another message to be sent before closing the sequence by sending a `CloseSequence` SOAP message.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

---

wsrm-source-make-offer

Indicates whether to include an offer for two-way.

Syntax

wsrm-source-make-offer {**on** | **off**}

Parameters

on Include an offer.

off (Default) Does not include an offer.

Guidelines

The **wsrm-source-make-offer** command indicates whether to include an offer for two-way Reliable Messaging in `CreateSequence` SOAP requests that are made as the result of request processing. Including an offer can result in the creation of a Reliable Messaging destination for the server to send responses on when the

DataPower service creates a Reliable Messaging source to send requests to the server. If the server does not accept the offer, DataPower server does not create a Reliable Messaging destination.

Related Commands

wsrm, **wsrm-source-request-create-sequence**

wsrm-source-maximum-queue-length

Specifies the maximum number of messages held in the queue.

Syntax

wsrm-source-maximum-queue-length *numberOfMessages*

Parameters

numberOfMessages
Specifies the size of the queue in number of messages. Use an integer in the range of 1 through 256. The default is 30.

Guidelines

The **wsrm-source-maximum-queue-length** command specifies the maximum number of messages held in the Reliable Messaging queue while waiting for `Ack` messages. This property controls memory utilization.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

wsrm-source-maximum-sequences

Sets the limit of simultaneous active sequences.

Syntax

wsrm-source-maximum-sequences *limit*

Parameters

limit Specifies the number of simultaneous active sequence. Use an integer in the range of 1 through 2048. The default is 400.

Guidelines

The **wsrm-source-maximum-sequences** command sets a limit on the maximum number of simultaneously active sequences from Reliable Messaging sources of this DataPower server. Each remote Reliable Messaging destination endpoint reference (URL) requires one sequence. Transactions that request the creation of sequences in excess of this limit result in a SOAP Fault. This property controls memory resource utilization.

Related Commands

wsrm

wsrm-source-request-ack-count

Specifies the number of messages to send before requesting acknowledgement.

Syntax

wsrm-source-request-ack-count *numberOfMessages*

Parameters

numberOfMessages
Use an integer in the range of 1 through 256. The default is 1.

Guidelines

The **wsrm-source-request-ack-count** command specifies the number of messages that the a Reliable Messaging source sends before including the `AckRequested` SOAP header to request an acknowledgement.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

wsrm-source-request-create-sequence

Indicates whether to create a source from the backend to the server.

Syntax

wsrm-source-request-create-sequence {**on** | **off**}

Parameters

on Creates a Reliable Messaging source.

off (Default) Does not create a Reliable Messaging source.

Guidelines

The **wsrm-source-request-create-sequence** command indicates whether to create a Reliable Messaging source from the backend to the server when there is SOAP data to sent to the server and when there is no Reliable Messaging source that was created by a `MakeOffer` from the server. The Reliable Messaging source is created by sending a `CreateSequence` SOAP request to the server address.

Related Commands

wsrm, **wsrm-source-exponential-backoff**, **wsrm-source-inactivity-close-interval**, **wsrm-source-make-offer**, **wsrm-source-maximum-queue-length**, **wsrm-source-request-ack-count**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

wsrm-source-response-create-sequence

Indicates whether to create a source from the front side to the client.

Syntax

wsrm-source-response-create-sequence {**on** | **off**}

Parameters

on Creates a Reliable Messaging source.

off (Default) Does not create a Reliable Messaging source.

Guidelines

When the WS-Addressing mode as defined by the **wsa-mode** command is **wsa2sync** or **wsa2wsa**, the **wsrm-source-response-create-sequence** command indicates whether to create a Reliable Messaging source from the front side to the client when there is SOAP data to send to the client and there is no Reliable Messaging source that was created by a `MakeOffer` from the client by sending a `CreateSequence` SOAP request to the WS-Addressing `ReplyTo` address.

Related Commands

wsa-mode, **wsrm**, **wsrm-source-exponential-backoff**, **wsrm-source-inactivity-close-interval**, **wsrm-source-maximum-queue-length**, **wsrm-source-request-ack-count**, **wsrm-source-retransmission-interval**, **wsrm-source-retransmit-count**

---

wsrm-source-retransmission-interval

Specifies the duration that a source waits.

Syntax

wsrm-source-retransmission-interval *interval*

Parameters

interval

Specifies the duration in milliseconds. Use an integer in the range of 10 through 60000. The default is 2000.

Guidelines

The **wsrm-source-retransmission-interval** command specifies the duration in milliseconds that a Reliable Messaging source waits for an `Ack` before retransmitting the message. This property also applies to the retransmission of the `CreateSequence` SOAP message.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-exponential-backoff**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

---

wsrm-source-retransmit-count

Specifies the number of times to retransmit a message.

Syntax

wsrm-source-retransmit-count *count*

Parameters

count Specifies the number of retransmissions. Use and integer in the range of 1 through 256. The default is 4.

Guidelines

The **wsrm-source-retransmit-count** command specifies the number of times a Reliable Messaging source retransmits a message before declaring a failure.

This command also controls the retransmission of `CreateSequence` requests.

Related Commands

wsrm, **wsrm-destination-accept-offers**, **wsrm-source-request-create-sequence**, **wsrm-source-response-create-sequence**

---

wsrm-source-sequence-ssl

Indicates whether to use an SSL session binding to protect sequence lifecycle messages.

Syntax

wsrm-source-sequence-ssl {**on** | **off**}

Parameters

on Uses an SSL session binding.

off (Default) Does not use an SSL session binding.

Guidelines

All Reliable Messaging control messages and sequence messages are bound to the original SSL/TLS session that is created by the Reliable Messaging source to transmit the `CreateSequence` control message. Sequence messages that are received by the Reliable Messaging destination with the correct identifier but on a different SSL/TLS session are rejected.

The lifetime of a SSL/TLS protected sequence is bound by the lifetime of the SSL/TLS session this is used to protect that sequence.

Related Commands

wsrm

---

xml-manager

Assigns an XML manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of the XML manager.

Guidelines

The **xml-manager** command assign an XML manager to the Web Service Proxy. An XML manager obtains and controls resources required by the Web Service Proxy. In the absence of an explicit limit, the DataPower appliance assigns the `default` XML Manager to support Web Service Proxy operations.

Related Commands

stylesheet-policy, **xml-manager** (Global)

Chapter 108. Web Services Management Agent configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Services Monitor Agent configuration mode. To enter this configuration mode, use the Global **wsm-agent** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Services Monitor configuration mode.

buffer-mode

Specifies a buffering strategy.

Syntax

buffer-mode {**buffer** | **discard**}

Parameters

buffer Buffers transaction information for the current domain in the expectation that a manager will connect. Buffering reduces the loss of transaction accounting information, but at the cost of more memory consumed.

discard
Discards transaction information for the current domain.

Guidelines

Buffering Mode controls the behavior of the Web Services Management Agent when there are no registered consumers of transaction events.

Records are accumulated until the configured size limits (**max-records** and **max-memory**) are reached. After maximum configured values are reached, new records will be dropped.

capture-mode

Specifies a message capture mode.

Syntax

capture-mode {**all-messages** | **faults** | **none**}

Parameters

all-messages
Captures all messages.

faults (Default) Captures fault messages.

none Does not capture messages.

Guidelines

capture-mode identifies messages that are captured and forwarded to a Web Services Manager for further analysis.

Not all Web Service Management protocols accommodate full message capture. Use the **all-messages** option only if the spooler can forward full messages. Use of this option incurs a performance penalty that can be seen when performing load testing.

---

max-memory

Specifies the maximum buffer usage in kilobytes.

Syntax

max-memory *kilobytes*

Parameters

kilobytes
Specifies the maximum buffer usage in Kilobytes.

Guidelines

Buffering controls the behavior of the Web Services Management Agent when there are no registered consumers of transaction events.

Buffering reduces the loss of transaction accounting information, but at the cost of more memory consumed.

Records are accumulated until the configured size limits are reached. After maximum values are reached, new records will be dropped.

---

max-records

Specifies the maximum number of buffered transactions.

Syntax

max-records *count*

Parameters

count Specifies the maximum number of buffered transactions.

Guidelines

Buffering controls the behavior of the Web Services Management Agent when there are no registered consumers of transaction events.

Buffering reduces the loss of transaction accounting information, but at the cost of more memory consumed.

Records are accumulated until the configured size limits are reached. After maximum values are reached, new records will be dropped.

Chapter 109. Web Services Monitor configuration mode

This chapter provides an alphabetic listing of commands that are available in Web Services Monitor configuration mode. To enter this configuration mode, use the Global **service-monitor** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in Web Services Monitor configuration mode.

endpoint-name

Specifies the WSDL-defined endpoint to monitor.

Syntax

endpoint-name *endpoint*

Parameters

endpoint
Identifies the target WSDL endpoint.

Guidelines

Use the endpoint name exactly as defined in the WSDL file.

endpoint-url

Specifies the URL of the WSDL endpoint to monitor.

Syntax

endpoint-url *URL*

Parameters

URL Specifies the URL of the target endpoint.

Guidelines

Use the URL exactly as defined in the WSDL file.

frontend-url

Specifies the URL that the client uses to access the WSDL endpoint that is monitored.

Syntax

frontend-url *URL*

Parameters

URL Specifies the URL sent by the client to access the monitored endpoint.

You can use wildcards to define a match pattern as follows:

***** The string wildcard matches 0 or more occurrences of any character.

? The single character wildcard matches one occurrence of any single character.

[] The delimiters bracket a character or numeric range:

`[1-5]` Matches 1, 2, 3, 4, or 5

`[xy]` Matches x or y

Guidelines

The **frontend-url** specifies the URL that the client uses to access the WSDL the is monitored by the current Web Service Monitor. This URL might or might not be identical to the one specified by the **endpoint-url** command.

---

operation

Specifies the administrative operations to monitor.

Syntax

operation *type target threshold value action*

Parameters

type Specifies the operations to monitor. Specify the following keyword:

all Monitors all WSDL-defined operations.

target Specifies whether to monitor errors or transactions. Specify one of the following keywords:

front Monitors error counts

rate Monitors transaction counts

threshold

Specifies the threshold level. Use one of the following keywords:

- **low**
- **high**

For example, as transaction rates rise, the first limit might be reached at 100 transactions per second, at which some action can be taken. The second limit might be reached at 300 transactions per second, at which some other action can be taken.

value Specifies the point (expressed in instances per second) at which the associated trigger (**low** or **high**) is fired.

action Specifies the administrative response to the firing of a trigger. Use one of the following keywords:

log Generates a log message

throttle

Queues as out-of-limits traffic

Examples

- Specifies monitor operations, generates log entries in response to more than 30 transactions per second, and throttles excessive transactions (greater than 50 per second).

```
# service-monitor WSMonitor-2
Web Services Monitor configuration mode
# operation all rate low 30 log
# operation all rate high 50 throttle
#
```

transport

Specifies the transport type that the monitored endpoint uses.

Syntax

transport *type*

Parameters

type Identifies the transport type. Use one of the following values:
- **HTTP-GET**
- **HTTP-SOAP**
- **SOAP-document**
- **SOAP-RPC**

wsdl

Specifies the location of the WSDL file.

Syntax

wsdl *URL*

Parameters

URL Specifies the location of the target WSDL file.

Guidelines

The WSDL file can reside on the local system or elsewhere on the network.

Chapter 110. WebSphere JMS configuration mode

This chapter provides an alphabetic listing of commands that are available in WebSphere JMS configuration mode. To enter this configuration mode, use the Global **wasjms-server** command.

While in this configuration mode, you define the parameters needed to locate and access a version 6.*x* WebSphere Application Server (WAS) JMS provider, running IBM JFAP (JetStream Formats and Protocols). A JMS provider enables messaging based on the Java Messaging Service (JMS). It provides J2EE connection factories to create connections for JMS destinations.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in WebSphere JMS configuration mode.

auto-retry

Enables or disables automatic critical error-recovery procedure.

Syntax

auto-retry {**on** | **off**}

Parameters

on (Default) Enables error recovery.

off Disables error recovery.

Guidelines

The **auto-retry** command enables or disables automatic critical error-recovery procedure that attempts to reestablish a connection that has been broken in response to an error condition.

Related Commands

retry-interval

default-message-type

Specifies the default JMS message type.

Syntax

default-message-type {**byte** | **text**}

Parameters

byte (Default) Specifies that the message payload is accessed as a Java byte array

text Specifies that the message payload is accessed as a Java string value

Guidelines

The **default-message-type** command specifies the default JMS message type. This message type is provided by the WebSphere JMS object only if the message type cannot be determined from the JMS message headers

enable-logging

Enables or disables expanded JMS logging facility.

Syntax

enable-logging {**on** | **off**}

Parameters

on Enables expanded JMS-specific logging.

off (Default) Disables expanded JMS-specific logging.

endpoint

Identifies a JMS non-default bootstrap server endpoint.

Syntax

endpoint *host port protocol*

Parameters

host Specifies the host name or IP address of a bootstrap server.

port Specifies the port that the bootstrap server monitors for incoming bootstrap requests.

protocol
Identifies the predefined transport chain that is provided by the bootstrap server. Specify one of the following values:

HTTP Specifies the predefined `BootstrapTunneledMessaging` transport chain (tunnels JFAP using HTTP wrappers).

HTTPS
Specifies the predefined `BootstrapTunneledSecureMessaging` transport chain (tunnels JFAP using HTTPS wrappers).

SSL Specifies the predefined `BootstrapSecureMessaging` transport chain (JFAP-SSL-TCP/IP).

TCP (Default) Specifies the predefined `BootstrapBasicMessaging` transport chain (JFAP-TCP/IP).

Guidelines

A *service integration bus* (SIB) supports applications that use message-based and service-oriented architectures. A bus is a group of interconnected servers and clusters that were added as members. Applications connect to a bus at one of the *messaging engines* that is associated with its bus members.

A messaging engine is a component that runs inside a server. The messaging engine manages messaging resources for a bus member. Applications connect to a messaging engine when accessing a SIB.

Applications (such as the WebSphere JMS object) that run outside the WebSphere Application Server (WAS) environment cannot directly locate a suitable messaging engine to connect to the target bus. In such cases the remote clients or servers must access the bus through a *bootstrap server* that is a member of the target bus. A bootstrap server is an application server that runs the SIB process. The bootstrap server does not need to run any message engines. Rather the bootstrap server selects a messaging engine that is running in an application server that supports the bootstrap protocol that is requested by the remote appliance.

To connect to a messaging engine:

1. The remote application first connects to a bootstrap server
2. The bootstrap server selects a messaging engine
3. The bootstrap server tells the client application to connect to that message engine to gain bus access.

A bootstrap server uses a host name or IP address, in conjunction with a port number and a bootstrap *transport chain*. The transport chain identifies the protocol stack that is offered by the bootstrap server to define an *endpoint* address.

The protocol stack that is used to access the bootstrap server does not need to be the same protocol stack that is used for actual message transfer via the bus.

You can use the **endpoint** command multiple times to identify more than one non-default bootstrap server.

Related Commands

messaging-bus, **target-transport-chain**

Examples

- Accesses the `jetStream-1` JMS object named. Defines a bootstrap server at the `192.168.34.87` IP address that monitors port 4500 for secure (SSL-enabled) connections.

```
# wasjms-server jetStream-1
Modify WebSphere JMS configuration
# endpoint 192.168.34.87 4500 SSL
#
```

maximum-message-size

Specifies the maximum message size.

Syntax

maximum-message-size *bytes*

Parameters

bytes Specifies the maximum message size in bytes Use an integer in the range of 0 through 1073741824. The default is 1048576 (1 MB). Use the special value of 0 to disable the enforcement of a maximum message size.

Related Commands

memory-threshold

memory-threshold

Specifies the maximum memory allocation for pending messages.

Syntax

memory-threshold *bytes*

Parameters

bytes Specifies the maximum memory to allocate in bytes. Use an integer in the range of 1048576 through 1073741824. This default is 268435456.

Related Commands

maximum-message-size

messaging-bus

Specifies the Service Integration Bus (SIB) to access the remote server.

Syntax

messaging-bus *name*

Parameters

name Specifies the name of the bus.

Related Commands

endpoint, **target-transport-chain**

Guidelines

A service integration bus (SIB) supports applications that use message-based and service-oriented architectures. A bus is a group of interconnected servers and clusters that were added as members of the bus. Applications connect to a bus at one of the messaging engines associated with its bus members.

If you have access to the WebSphere Administrative Console, you can view bus information, that includes bus members and messaging engines, queues and topics, and the bus-specific default topic space through the **Service integration** → **Buses** menu.

password

Specifies the password to access the server.

Syntax

password *password*

Parameters

password
Specifies the password to access the remote server.

Guidelines

The **password** command specifies the password to use in conjunction with the value provided by the **username** command to access the remote server.

Related Commands

username

---

retry-interval

Specifies the interval between attempts to reestablish a connection.

Syntax

retry-interval *seconds*

Parameters

seconds

Specifies the interval, in seconds, between attempts to reestablish a downed connection. The default is 1.

Related Commands

auto-retry

---

sessions-per-connection

Specifies the maximum number of concurrent multiplexed sessions that a single connection can support.

Syntax

sessions-per-connection *sessions*

Parameters

sessions

Specifies the maximum concurrent sessions to support. Use an integer greater than 4. The default is 20.

Guidelines

Session requests in excess of the value of the **sessions-per-connection** command trigger the establishment of a new connection to the server. A new connection cannot be established unless the number of current connections is less than the value of the **total-connection-limit** command.

Assume default values (20 sessions per connection and 5 total connections) and 3 active fully-subscribed connections, a new session request generates the establishment of a 4th connection.

Related Commands

total-connection-limit

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Identifies the existing SSL Proxy Profile.

Guidelines

In the absence of an explicitly assigned SSL Proxy Profile, the firmware establishes a nonsecure connection to the server. If you specify an SSL Proxy Profile, the cipher specification for the proxy is replaced by an IBM default cipher specification (SSL_RSS_WITH_NULL_MD5).

You can use the **ssl-cipher** and **ssl-fips** commands to tailor SSL requirements.

Related Commands

ssl-cipher, **ssl-fips**

---

ssl-cipher

Identifies the IBM cipher specification that the SSL Proxy Profile uses to establish a secure connection.

Syntax

ssl-ciphers *cipher*

Parameters

cipher Identifies the cipher specification and takes one of the following values:

- SSL_RSA_WITH_NULL_MD5 (Default)
- SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
- SSL_RSA_EXPORT_WITH_RC4_40_MD5
- SSL_RSA_WITH_RC4_128_MD5
- SSL_RSA_WITH_NULL_SHA
- SSL_RSA_EXPORT1024_WITH_RC4_56_SHA
- SSL_RSA_WITH_RC4_128_SHA
- SSL_RSA_WITH_DES_CBC_SHA
- SSL_RSA_EXPORT1024_WITH_DES_CBC_SHA
- SSL_RSA_FIPS_WITH_DES_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_DES_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA

Guidelines

In the absence of an explicitly assigned SSL Proxy Profile, the firmware establishes a nonsecure connection to the WebSphere Application Server.

If you specify an SSL Proxy, the cipher suite that is associated with the proxy is replaced by an IBM default cipher specification (SSL_RSS_WITH_NULL_MD5).

You can use the **ssl-cipher** and **ssl-fips** commands to tailor SSL requirements.

Related Commands

ssl, **ssl-fips**

---

ssl-fips

Requires a FIPS-compliant cipher specification.

Syntax

ssl-fips {**on** | **off**}

Parameters

on Requires FIPS-compliant cipher.

off (Default) Makes the use of a FIPS-compliant cipher optional.

Related Commands

ssl, **ssl-cipher**

---

target-transport-chain

Identifies the predefined transport chain to use for message exchanges.

Syntax

target-transport-chain *transport-chain*

Parameters

transport-chain
Identifies the predefined transport chain provided by the WebSphere Application Server and takes one of the following values:

InboundBasicMessaging
(Default) Specifies the predefined `InboundBasicMessaging` transport chain (JFAP-TCP/IP)

InboundHTTPMessaging
Specifies the predefined `InboundHTTPMessaging` transport chain (tunnels JFAP using HTTP wrappers)

InboundHTTPSMessaging
Specifies the predefined `InboundHTTPSMessaging` transport chain (tunnels JFAP using HTTPS wrappers)

InboundSecureMessaging
Specifies the predefined `InboundSecureMessaging` transport chain (JFAP-SSL-TCP/IP)

Guidelines

If you have access to the WebSphere Administrative Console, you can view transport chain information through the **Application Servers** → **serverName** → **Transport Chain** menu.

The transport chain used for message exchange need not match the chain used for bootstrap access.

Related Commands

messaging-bus, **target-transport-chain**

total-connection-limit

Specifies the maximum number of open connections to the server.

Syntax

total-connection-limit *connections*

Parameters

connections
Specifies the maximum number of open connections that can be established to the server. The minimum is 1. The default is 5.

Related Commands

sessions-per-connection

transactional

Indicates whether to use transaction-based processing.

Syntax

transactional {**on** | **off**}

Parameters

on Enables transaction-based processing.

off Disables transaction-based processing.

Guidelines

The **transactional** command enables or disables transactional processing. When transactional processing is enabled, messages are acknowledged only after the transaction succeeds.

username

Specifies the account name to access the server.

Syntax

username *account*

Parameters

account
Specifies the account to use in conjunction with the value of the **password** command to access the server.

Related Commands

password

Chapter 111. WebSphere JMS Front Side Handler configuration mode

This chapter provides an alphabetic listing of commands that are available in WebSphere JMS Front Side Handler configuration mode. To enter this configuration mode, use the Global **source-wasjms** command.

While in these modes, you can configure properties that define a client-side traffic handler.

All of the commands listed in "Common commands" on page 2 and most, but not all, of the commands listed in Chapter 129, "Monitoring commands," on page 1053 are available in these configuration modes.

get-queue

Specifies the queue that contains client-originated request messages.

Syntax

get-queue *name*

Parameters

name Specifies the name of the queue that contains WAS JMS request messages.

Related Commands

put-queue, **selector**

Guidelines

Identification of a GET queue is required.

The WebSphere JMS Front Side Handler monitors the GET queue for incoming client requests. Upon message receipt, the handler forwards the extracted message to the a local WebSphere JMS object that will gateway the message to a remote WebSphere JMS default message provider.

Within a WAS JMS environment, message destinations are characterized as either *queues* or *topics*.

- A JMS queue is a destination for point-to-point messaging.
- A JMS topic is a destination for publish-subscribe messaging.

If desired, use the `queue:` or `topic:` prefix to distinguish between point-to-point and publish-subscribe messaging models.

Examples

- Identifies the `hiPriority` queue as the GET queue that contains client-originated request messages. Identifies the `queue:loPriority` queue as the GET queue that contains client-originated point-to-point messages. Identifies the `topic:schedules` queue as the GET queue that contains client-originated publish-subscribe messages.

```
# get-queue hiPriority
# get-queue queue:loPriority
# get-queue topic:schedules
#
```

put-queue

Specifies the queue that contains server-originated WAS JMS reply messages.

Syntax

put-queue *name*

Parameters

name Specifies the name of the queue that contains WAS JMS response messages.

Related Commands

get-queue

Guidelines

WAS JMS response messages are originated by a remote WAS JMS default message provider and put into this queue by a local WebSphere JMS object.

Identification of a PUT queue is optional. A PUT queue should be configured if server replies are expected. If reply messages are not expected, a PUT queue need not be configured. In the absence of a PUT queue, any received replies are dropped.

Within a WAS JMS environment, message destinations are characterized as either queues or topics.

- A JMS queue is a destination for point-to-point messaging.
- A JMS topic is a destination for publish-subscribe messaging.

If desired, use the `queue:` or `topic:` prefix to distinguish between point-to-point and publish-subscribe messaging models.

Examples

- Identifies the `wsJmsResponse` queue as the PUT queue that contains server-originated response messages. Identifies the `queue:loPriority` queue as the PUT queue that contains server-originated point-to-point messages. Identifies the `topic:schedules` queue as the PUT queue that contains server-originated publish-subscribe messages.

```
# put-queue wsJmsResponse
# put-queue queue:loPriority
# get-queue topic:schedules
#
```

reply-topic-space

Specifies a nondefault topic space.

Syntax

reply-topic-space *name*

Parameters

name Specifies the name of the nondefault topic space.

Guidelines

A topic space is a hierarchy of topics used for publish/subscribe messaging. Topics with the same name can exist in multiple topic spaces, but there can be only one topic space with a given name in a service integration bus.

For example, consider a topic hierarchy split into the following topic spaces:

`library`
: Indicates topics for document management

`sales` Indicates topics for marketing and sales tracking

`engineering`
: Indicates topics for engineering and technology

The topic `volumes` can appear in all three topic spaces, and have a very different meaning in each.

Use this command to disambiguate a topic if the response destination is a topic whose name appears in multiple topic spaces.

Related Commands

request-topic-space

Examples

- Identifies `sales` as the topic space.

```
# reply-topic-space sales
#
```

---

request-topic-space

Specifies a nondefault topic space.

Syntax

request-topic-space *name*

Parameters

name Specifies the name of the nondefault topic space.

Related Commands

reply-topic-space

Guidelines

A topic space is a hierarchy of topics used for publish/subscribe messaging. Topics with the same name can exist in multiple topic spaces, but there can be only one topic space with a given name in a service integration bus.

For example, consider a topic hierarchy split into the following topic spaces:

`library`
Indicates topics for document management

`sales` Indicates topics for marketing and sales tracking

`engineering`
Indicates topics for engineering and technology

The topic `volumes` can appear in all three topic spaces, and have a very different meaning in each.

Use this command to disambiguate a topic if the request destination is a topic whose name appears in multiple topic spaces.

Examples

- Identifies `engineering` as the topic space.

```
# request-topic-space engineering
#
```

selector

Specifies the SQL-like expression to select messages from the get queue.

Syntax

selector *expression*

Parameters

expression
Defines the SQL-like expression to filter messages from the get queue. If the expression contains spaces, enclose in double quotation marks.

Guidelines

The message selector is a conditional expression based on a subset of SQL92 conditional expression syntax. The conditional expression enables the WebSphere JMS Front Side Handler to identify *messages of interest*.

The conditional expression does not operate on the body of the JMS message. The expression examines JMS message headers and properties. The properties are proprietary, user-created headers that might appear between the required headers and the message body.

The following JMS headers are required:

`Destination`
Contains the destination (queue) to which the message is being sent

`DeliveryMode`
Contains the delivery mode (`PERSISTENT` or `NON_PERSISTENT`)

`Expiration`
Contains a message TTL or a value of 0 indicating an unlimited TTL

`Priority`
Contains the message priority expressed as a digit from 0 (lowest priority) to 9 (highest priority)

`MessageID`
Contains a unique message identifier starting with the prefix `ID:`, or a null value, effectively disabling message ID

`Timestamp`
Contains the time the message was handed off for transmission, not the time it was actually sent

`CorrelationID`
Contains a means of associating one message (for example, a response) with another message (for example, the original request)

`ReplyTo`
Contains the destination (queue) to which a reply to this message should be sent

`Type` Contains a message identifier provided by the application

`Redelivered`
Contains a Boolean indicating that the message has been delivered in the past, but not yet acknowledged

Configuration of a message selector is optional. If a message selector is not specified, all incoming client requests are transferred by the WebSphere JMS Front Side Handler to the WebSphere JMS object for processing.

If a message selector is specified, only those client requests that match the criteria specified by the SQL expression are transferred to the WebSphere JMS object for processing. All other messages are dropped from the GET queue.

Related Commands

get-queue

Examples

- Indicates that only client requests that have a `DeliveryMode` of `PERSISTENT` are forwarded to the WebSphere JMS object for processing. All other messages are dropped from the GET queue.

```
# selector "DeliveryMode LIKE PERSISTENT"
#
```

---

server

Assigns the WebSphere JMS object.

Syntax

server *name*

Parameters

name Specifies the name of an existing WebSphere JMS object.

Examples

- Identifies the `webSphereJMS` WebSphere JMS object as supported by this protocol handler.

```
# server webSphereJMS
#
```

Chapter 112. WebSphere MQ Gateway configuration mode (deprecated)

The WebSphere MQ Gateway configuration mode and the commands that are available in this configuration mode are deprecated. To provide the same behavior, create a Multi-Protocol Gateway service or create a Web Service Proxy service.

This configuration mode provides the following commands:

- **client**
- **direction**
- **queue-manager**
- **server**
- **url**

For details about the commands in this deprecated configuration mode, refer to the online help.

Chapter 113. WebSphere MQ Host configuration mode (deprecated)

The WebSphere MQ Host configuration mode and the commands that are available in this configuration mode are deprecated. To provide the same behavior, create a Multi-Protocol Gateway service or create a Web Service Proxy service.

This configuration mode provides the following commands:

- **attachments**
- **back-attachment-format**
- **ccsi**
- **content-type**
- **count-monitor**
- **credentials**
- **default-namespace**
- **duration-monitor**
- **firewall-extensions**
- **front-attachment-format**
- **get-queue**
- **message-type**
- **parameter**
- **put-queue**
- **queue-manager**
- **rule**
- **soap-schema-url**
- **xml-manager**

For details about the commands in this deprecated configuration mode, refer to the online help.

Chapter 114. WebSphere MQ Proxy configuration mode (deprecated)

The WebSphere MQ Proxy configuration mode and the commands that are available in this configuration mode are deprecated. To provide the same behavior, create a Multi-Protocol Gateway service or create a Web Service Proxy service.

This configuration mode provides the following commands:

- **back-attachment-format**
- **back-queue-manager**
- **concurrent**
- **content-type**
- **count-monitor**
- **credentials**
- **default-namespace**
- **duration-monitor**
- **error-rule**
- **firewall-extensions**
- **front-attachment-format**
- **front-queue-manager**
- **parameter**
- **request-attachments**
- **request-get-queue**
- **request-put-queue**
- **request-rule**
- **request-type**
- **response-attachments**
- **response-get-queue**
- **response-put-queue**
- **response-rule**
- **response-type**
- **root-part-not-first-action**
- **soap-schema-url**
- **timeout**
- **xml-manager**

For details about the commands in this deprecated configuration mode, refer to the online help.

Chapter 115. WS-Proxy Endpoint Rewrite configuration mode

This chapter provides an alphabetic listing of commands that are available in WS-Proxy Endpoint Rewrite configuration mode. While in this configuration mode, you can define an Endpoint Rewrite policy that a Web Service Proxy service uses.

To enter this configuration mode, use the Global **wsm-endpointrewrite** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this configuration mode.

backend-rule

Adds, edits, or deletes a remote endpoint rewrite rule.

Syntax

backend-rule *pattern protocol host port uri binding-protocol*

Parameters

pattern Specifies a PCRE to specify web services port to rewrite endpoint for. There is no default value for this argument.

protocol
Specifies the part of the URL from web service binding that specifies the protocol. This defaults to the protocol found in the WSDL file.

host Specifies the part of the URL from web service binding that specifies the host name or IP address. If not specified, the value from the WSDL will be used.

To use a load balancer, specify the name of an existing Load Balancer Group.

port Specifies the part of the URL from web service binding that specifies the port. A value of `0` uses the port value from the WSDL. The default is `0`.

uri Specifies the part of the URL from web service binding that specifies the remote path. If no string is configured, the value from the WSDL will be used.

binding-protocol
Specifies the WSDL binding protocol to use in the rewritten web service.

default
(Default) Uses the binding protocol in the WSDL files.

http-get
Uses the HTTP binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/http/`).

http-post
Uses the HTTP binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/http/`).

soap-11
Uses the SOAP 1.1 binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/soap11/`).

soap-12
Uses the SOAP 1.2 binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/soap12/`).

Guidelines

All of the arguments for the **backend-rule** command must be specified in the documented order.

A Remote Endpoint specifies the location to which requests are sent by a Web Service Proxy after processing the request. This is the back end, or application endpoint, of the transaction. It is possible to direct traffic to an endpoint other than that specified in the underlying WSDL by rewriting the endpoint.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

listener-rule, **publisher-rule**

Examples

- Creates an Endpoint Rewrite policy named `someBanking`, moves to WS-Proxy Endpoint Rewrite configuration mode, and defines a remote endpoint rewrite rule with the following properties:
  - A PCRE of `{http://somebank.com}SomeBankPort` to match the web service port
  - A protocol of `http`
  - An IP address for the remote endpoint of `10.10.13.35`
  - A port of `2068`
  - A local path of `/SomeBankService/services/SomeBankPort`

```
# wsm-endpointrewrite someBanking
WS-Proxy Endpoint Rewrite configuration mode
# backend-rule
 "{http://somebank.com}SomeBankPort" "http" "10.10.13.35" "2068"
 "/SomeBankService/services/SomeBankPort"
#
```

listener-rule

Adds, edits, or deletes a local endpoint rewrite rule.

Syntax

listener-rule *pattern protocol host port uri front-protocol use-front-protocol*

Parameters

pattern Specifies a PCRE to specify web services port to rewrite endpoint for.

protocol
Specifies the part of the URL from web service binding that specifies the protocol. The default is `default`.

This argument is relevant when *use-front-protocol* is **off**. This argument is ignored when *use-front-protocol* is **on**.

host Specifies the part of the URL from web service binding that specifies the host alias or IP address. The default is `0.0.0.0`.

This argument is relevant when *use-front-protocol* is **off**. This argument is ignored when *use-front-protocol* is **on**.

port Specifies the part of the URL from web service binding that specifies the port. A value of `0` will use the port specified in the WSDL. The default is 0.

This argument is relevant when *use-front-protocol* is `off`. This argument is ignored when *use-front-protocol* is `on`.

uri Specifies the part of the URL from web service binding that specifies the local path. If no string is configured, the value from the WSDL will be used.

front-protocol
Specifies the front side handler to use for matching web service ports.

This argument is relevant when *use-front-protocol* is **on**. This argument is ignored when *use-front-protocol* is **off**.

use-front-protocol
Specifies a keyword (**on** or **off**) that indicates whether the front side protocol handler determines the protocol, interface, and port of the local address for any matched WSDL service port. Specifying **on** overrides the values specified for *protocol*, *hostname*, and *port* in this rewrite rule. The default is **off**.

Guidelines

All of the arguments for the **listener-rule** command must be specified in the documented order.

The Local Endpoint defines the IP address, TCP port and URI offered by the Proxy to clients making a request for service as described in the published WSDL. The Proxy listens for requests at the Local Endpoint URL.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

backend-rule, **publisher-rule**

Examples

- Creates an Endpoint Rewrite policy named `testing`, moves to WS-Proxy Endpoint Rewrite configuration mode, and defines a local endpoint rewrite rule with the following properties:
  - A PCRE of `.*` to match the web service port
  - Use the front side handler `Searcher` for matching web service ports.
  - Because the front side handler `Searcher` is used and is in use, the properties for the `default` protocol, the `0.0.0.0` IP address, and the `0` port are ignored (`use-front-protocol` is `on`).

```
# wsm-endpointrewrite testing
WS-Proxy Endpoint Rewrite configuration mode
# listener-rule ".*" "default" "0.0.0.0" "0" "/search/beta2" "Searcher" "on"
#
```

publisher-rule

Adds, edits, or deletes a publish endpoint rewrite rule.

Syntax

publisher-rule *pattern protocol host port uri*

Parameters

pattern Specifies a PCRE to specify web services port to rewrite endpoint for.

protocol
Specifies the part of the URL from web service binding that specifies the protocol.

host Specifies the part of the URL from web service binding that specifies the host name or IP address.

port Specifies the part of the URL from web service binding that specifies the port. A value of 0 will use the port specified in the WSDL. The default is 0.

uri Specifies the part of the URL from web service binding that specifies the local path. If no string is configured, the value from the WSDL will be used.

Guidelines

All of the arguments for the **publisher-rule** command must be specified in the documented order.

This rule rewrites the endpoint published to UDDI registries or included in the WSDL supplied by the Proxy in response to a request for a WSDL describing the services offered by the Proxy.

PCRE documentation is available at the following web site:

http://www.pcre.org

Related Commands

backend-rule, **listener-rule**

Examples

- Creates an Endpoint Rewrite policy named `someBanking`, moves to WS-Proxy Endpoint Rewrite configuration mode, and defines a publish endpoint rewrite rule with the following properties:
  - A PCRE of `{http://somebank.com}SomeBankPort` to match the web service port
  - A protocol of `http`
  - An IP address for the remote endpoint of `10.10.13.35`
  - A port of `2068`
  - A local path of `/SomeBankService/services/SomeBankPort`

```
# wsm-endpointrewrite someBanking
WS-Proxy Endpoint Rewrite configuration mode
# publisher-rule
 "{http://somebank.com}SomeBankPort" "http" "10.10.13.35"
 "2068" "/SomeBankService/services/SomeBankPort"
#
```

subscription-backend-rule

Adds, edits, or deletes a subscription remote endpoint rewrite rule.

Syntax

subscription-backend-rule *subscription protocol host port uri binding-protocol*

Parameters

subscription
Specifies the name of an existing UDDI Subscription to match against a subscription that the Proxy uses for this rewrite rule.

protocol
Specifies the part of the URL from web service binding that specifies the protocol. This defaults to the protocol found in the WSDL file.

host Specifies the part of the URL from web service binding that specifies the host name or IP address. If not specified, the value from the WSDL will be used.

port Specifies the part of the URL from web service binding that specifies the port. A value of `0` uses the port value from the WSDL. The default is `0`.

uri Specifies the part of the URL from web service binding that specifies the remote path. If no string is configured, the value from the WSDL will be used.

binding-protocol
Specifies the WSDL binding protocol to use in the rewritten web service.

default
(Default) Uses the binding protocol in the WSDL files.

http-get
Uses the HTTP binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/http/`).

http-post
Uses the HTTP binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/http/`).

soap-11
Uses the SOAP 1.1 binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/soap11/`).

soap-12
Uses the SOAP 1.2 binding for WSDL 1.1 (`http://schemas.xmlsoap.org/wsdl/soap12/`).

Guidelines

All of the arguments for the **subscription-backend-rule** command must be specified in the documented order.

A Remote Endpoint specifies the location to which requests are sent by a Web Service Proxy after processing the request. This is the backend endpoint, of the transaction. It is possible to direct traffic to an endpoint other than that specified in the underlying WSDL by rewriting the endpoint.

Related Commands

subscription-listener-rule, **subscription-publisher-rule**

Examples

- Creates an Endpoint Rewrite policy named `someBanking`, moves to WS-Proxy Endpoint Rewrite configuration mode, and defines a subscription remote endpoint rewrite rule with the following properties:
  - The `uddiSubscriber-SomeBankPort` UDDI subscription to match
  - A protocol of `http`
  - An IP address for the remote endpoint of `10.10.13.35`
  - A port of `2068`
  - A local path of `/SomeBankService/services/SomeBankPort`

```
# wsm-endpointrewrite someBanking
WS-Proxy Endpoint Rewrite configuration mode
# subscription-backend-rule
 "uddiSubscriber-SomeBankPort" "http" "10.10.13.35" "2068"
 "/SomeBankService/services/SomeBankPort"
#
```

subscription-listener-rule

Adds, edits, or deletes a subscription local endpoint rewrite rule.

Syntax

subscription-listener-rule *subscription protocol host port uri front-protocol use-front-protocol*

Parameters

subscription
: Specifies the name of an existing UDDI Subscription to match against a subscription that the Proxy uses for this rewrite rule.

protocol
: Specifies the part of the URL from web service binding that specifies the protocol. The default is `default`.

  This argument is relevant when *use-front-protocol* is **off**. This argument is ignored when *use-front-protocol* is **on**.

host
: Specifies the part of the URL from web service binding that specifies the host alias or IP address. The default is `0.0.0.0`.

  This argument is relevant when *use-front-protocol* is **off**. This argument is ignored when *use-front-protocol* is **on**.

port
: Specifies the part of the URL from web service binding that specifies the port. A value of `0` will use the port specified in the WSDL. The default is 0.

  This argument is relevant when *use-front-protocol* is `off`. This argument is ignored when *use-front-protocol* is `on`.

uri Specifies the part of the URL from web service binding that specifies the local path. If no string is configured, the value from the WSDL will be used.

front-protocol
Specifies the front side handler to use for matching web service ports.

This argument is relevant when *use-front-protocol* is **on**. This argument is ignored when *use-front-protocol* is **off**.

use-front-protocol
Specifies a keyword (**on** or **off**) that indicates whether the front side protocol handler determines the protocol, interface, and port of the local address for any matched WSDL service port. Specifying **on** overrides the values specified for *protocol*, *hostname*, and *port* in this rewrite rule. The default is **off**.

Guidelines

All of the arguments for the **subscription-listener-rule** command must be specified in the documented order.

The Local Endpoint defines the IP address, TCP port and URI offered by the Proxy to clients making a request for service as described in the published WSDL. The Proxy listens for requests at the Local Endpoint URL.

Related Commands

subscription-backend-rule, **subscription-publisher-rule**

Examples

- Creates an Endpoint Rewrite policy named `testing`, moves to WS-Proxy Endpoint Rewrite configuration mode, and defines a local endpoint rewrite rule with the following properties:
  - The `uddiSubscriber-SomeBankPort` UDDI subscription to match the web service port
  - Use the front side handler `Searcher` for matching web service ports.
  - Because the front side handler `Searcher` is used and is in use, the properties for the `default` protocol, the `0.0.0.0` IP address, and the `0` port are ignored (`use-front-protocol` is `on`).

```
# wsm-endpointrewrite testing
WS-Proxy Endpoint Rewrite configuration mode
# subscription-listener-rule "uddiSubscriber-SomeBankPort" "default" "0.0.0.0"
 "0" "/search/beta2" "Searcher" "on"
#
```

subscription-publisher-rule

Adds, edits, or deletes a subscription publish endpoint rewrite rule.

Syntax

subscription-publisher-rule *subscription protocol host port uri*

Parameters

subscription
Specifies the name of an existing UDDI Subscription to match against a subscription that the Proxy uses for this rewrite rule.

protocol
Specifies the part of the URL from web service binding that specifies the protocol.

host Specifies the part of the URL from web service binding that specifies the host name or IP address.

port Specifies the part of the URL from web service binding that specifies the port. A value of 0 will use the port specified in the WSDL. The default is 0.

uri Specifies the part of the URL from web service binding that specifies the local path. If no string is configured, the value from the WSDL will be used.

Guidelines

All of the arguments for the **publisher-rule** command must be specified in the documented order.

This rule rewrites the endpoint published to UDDI registries or included in the WSDL supplied by the Proxy in response to a request for a WSDL describing the services offered by the Proxy.

Related Commands

subscription-backend-rule, **subscription-listener-rule**

Examples

- Creates an Endpoint Rewrite policy named `someBanking`, moves to WS-Proxy Endpoint Rewrite configuration mode, and defines a publish endpoint rewrite rule with the following properties:
  - The `uddiSubscriber-SomeBankPort` UDDI subscription to match the web service port
  - A protocol of `http`
  - An IP address for the remote endpoint of `10.10.13.35`
  - A port of `2068`
  - A local path of `/SomeBankService/services/SomeBankPort`

```
# wsm-endpointrewrite someBanking
WS-Proxy Endpoint Rewrite configuration mode
# subscription-publisher-rule
 "uddiSubscriber-SomeBankPort" "http" "10.10.13.35"
 "2068" "/SomeBankService/services/SomeBankPort"
#
```

Chapter 116. WS-Proxy Processing Policy configuration mode

This chapter provides an alphabetic listing of commands that are available in WS-Proxy Processing Policy configuration mode. To enter this configuration mode, use the Global **wsm-stylepolicy** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in WS-Proxy Processing Policy configuration mode.

filter

Identifies a default style sheet to filter documents.

Syntax

filter *URL*

Parameters

URL Specifies the location of the style sheet.

Guidelines

The assigned default style sheet performs XML filtering only if a candidate XML document fails to match any of the filter rules defined within the processing policy.

Examples

- Identifies `validate.xsl` as the default style sheet to filter documents.

```
# filter store:///validate.xsl
#
```

match

Adds a Policy Map or deletes all Policy Maps.

Syntax

match *type value matching-rule global-rule subscription*

no match

Parameters

type Specifies the type of the WSDL component to match.

all (Default) Matches all input. Includes all component types.

operation
Matches when the identified operation is requested in the current transaction.

Matches `wsdl:binding/operation/@name` when formatted as `{bindingNamespace}name`, or matches `wsdl:service/wsdl:port` when formatted as `{serviceNamespace}port-name/operation-name`.

port Matches when the operation requested in the current transaction is included in the identified WSDL port.

Matches `wsdl:service/wsdl:port/@name` formatted `{serviceNamespace}port-name`.

service
Matches when the operation requested in the current transaction is included in the identified WSDL service.

Matches `wsdl:service/@name` formatted `{serviceNamespace}name`.

subscription
Matches an identified subscription key.

wsdl Matches when the operation requested in the current transaction is defined in the identified WSDL file.

value Identifies the name of the WSDL-defined component. The value to specify depends on the identified WSDL component type.

`all` Specify double quotation marks (""). This combination eliminates the WSDL component from consideration.

`operation`
Specifies the name of the WSDL operation.

`port` Specifies the name of the WSDL port.

`service`
Specifies the name of the WSDL service.

`subscription`
Specify double quotation marks (""). Any specified value is ignored.

`wsdl` Specifies the name of the WSDL file.

matching-rule
Specifies the name of a matching rule (previously created with the **matching** command and populated with the **httpmatch** or **urlmatch** commands) that serves as a source of URL or HTTP templates. Candidate documents that match any of the templates contained within the rule may be processed in accordance the associated named transformation or filtering rule.

global-rule
Specifies the name of a Processing Rule (previously created with the **rule** command) that defines processing procedures for documents that:

1. Match any of the expressions contained in the associated matching rule, and
2. Match the rule direction. A *request-rule* applies only to client-originated documents; a *response-rule* applies only to server-originated documents; while a *bidirectional* rule applies to all documents regardless of source.

subscription
Specifies the name of an existing Subscription object. Required when the value specified by the component type is `subscription`.

Guidelines

Use the **no match** command to delete all policy maps from the processing policy. To delete or modify a specific policy map, use the WebGUI.

Examples

- Adds the `star` matching rule and the `valClientServer` processing rule.

```
# match all "" star valClientServer
#
```

- Adds the `test` matching rule and the `valClientServer` processing rule when the match is against the `wsrrSub-1` subscription.

```
# match subscription "" test valClientServer wsrrSub-1
#
```

- Remove all rules from the current Processing Policy.

```
# no match
#
```

xsldefault

Identifies a default style sheet to transform documents.

Syntax

xsldefault *URL*

Parameters

URL Specifies the location of the default style sheet.

Guidelines

This default style sheet performs XML transformation only if a candidate XML document fails to match any transform rule in the processing policy.

Examples

- Identifies `identity.xsl` as the default style sheet to transform documents.

```
# xsldefault store:///identity.xsl
#
```

Chapter 117. WS-Proxy Processing Rule configuration mode

This chapter provides an alphabetic listing of commands that are available in WS-Proxy Processing Rule configuration mode. To enter this configuration mode, use the Global **wms-rule** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in WS-Proxy Processing Rule configuration mode.

aaa

Adds an AAA action.

Syntax

aaa *input-context name* [*output-context*]

Parameters

input-context
Identifies the context that contains the document authenticated or authorized by the AAA policy that is implemented in this Processing Rule.

name Specifies the name of an AAA Policy.

output-context
Optionally identifies the context where any post processing output is stored. Use `OUTPUT` to specify the final policy output, that is the transformed client request or transformed server response.

Examples

- Applies the `AAA-Policy1` AAA Policy to the original input to the Processing Rule.

```
# aaa INPUT AAA-Policy-1
#
```

action

Adds or deletes a named action.

Syntax

action *name*

no action *name*

no action

Parameters

name Specifies the name of the action.

Guidelines

Use the **no action** command to delete a named action from the current Processing Rule or to delete all actions from the current Processing Rule.

Examples

- Applies the `checkError` rule.

```
# action checkError
#
```

- Deletes the `checkError` rule from the current Processing Rule.

```
# no action checkError
#
```

---

call

Adds a `call` action.

Syntax

call *input-context rule output-context*

Parameters

input-context
Specifies the input context that contains the rule to invoke. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

rule Specifies the name of the rule to invoke.

output-context
Specifies the output context to store the result. Specify `OUTPUT` to use the final policy output, which is the transformed client request or transformed server response

Guidelines

A `call` action invokes another named rule

Examples

- Applies the specified call rule, `processRequest`, to the document in the `temp1` context and moves the rule results to the `temp2` context.

```
# call temp1 processRequest temp2
#
```

---

checkpoint

Applies a `checkpoint` action.

Syntax

checkpoint *event* [*input-context*]

Parameters

checkpointEvent
Identifies the event that triggers the checkpoint and takes one of the following values:

`AuthComplete`
Indicates the completion of an authentication process

`Fault` Indicates a fault condition

`Request`
Indicates the input of a client-originated document

`Response`
Indicates the input of a server-originated document

input-context
Optionally identifies the context in which the checkpoint is triggered. The default is `INPUT`.

Examples

- Adds a checkpoint triggered by a fault occurrence in the `temp1` context.

```
# checkpoint fault temp1
#
```

convert-http

Adds a `convert-http` action.

Syntax

convert-http *input-context output-context* [*map*]

Parameters

input-context
Specifies the context that contains the non-XML source. Use `INPUT` to specify the initial policy input, that is the original client request or server response.

output-context
Specifies an output context where the converted document is stored. Specify `OUTPUT` to use the final policy output, which is the transformed client request or transformed server response

map Optionally identifies the input conversion map to perform document encoding. The default is to treat as URL-escaped.

Guidelines

A `convert-http` action implements an XML conversion rule. A Processing Rule converts non-XML input into XML. Examples on non-XML input are an HTTP GET, an HTTP POST, or an HTML form.

Examples

- Converts the original input to the rule to XML and puts the XML content in the `tempParams` context. Specifies the encoding of non-XML input with the `ICM-normal` Input Conversion Map.

```
# convert-http INPUT tempParams ICM-normal
#
```

extract

Adds an `extract` action.

Syntax

extract *input-context output-context expression* [*variable*]

Parameters

input-context
Specifies the context to which to apply the XPath expression. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

output-context
Specifies the context that stores the result of the XPath expression. Specify `OUTPUT` to use the final policy output, which is the transformed client request or transformed server response

expression
Specifies the XPath expression that is applied and can be expressed in standard XPath format or as a `var://` URL that expands to an XPath expression.

variable
Optionally specifies a variable, within the output context, in which to store the result of the XPath expression.

Guidelines

An `extract` action applies an XPath expression to a context and stores the result in another context.

In the absence of *variable* argument, the results of the XPath expression are stored as the default contents (tree) of the destination context.

Examples

- Applies the `//games/url` XPath expression to the INPUT context and stores the result in the `three` context.

```
# extract INPUT three //games/url
#
```

- Applies the `//games/url` XPath expression to the INPUT context and stores the result in the `url` variable within the `three` context.

```
# extract INPUT three //games/url var://local/url
#
```

- Applies the XPath expression referenced by the local `xpath` variable and stores the result in the `url` variable within the `three` context.

```
# extract INPUT three var://local/xpath var://local/url
#
```

- Same as the previous example, but note the use of an explicit address for the optional *variable* argument.

```
# extract INPUT three var://local/xpath var://three/url
#
```

---

fetch

Adds a `fetch` action.

Syntax

fetch *url output-context*

Parameters

url Specifies the resource to be fetched and can be expressed as a URL or as a `var://` URL that expands to a URL.

output-context
Specifies the context in which to store the retrieved resource.

Guidelines

A `fetch` action retrieves a remote resource for use in a Processing Rule, You can use any protocol-specific URL when addressing the target resource.

Examples

- Retrieves the resource that is referenced by the `doc` variable in the default context and stores it in the `TestIt` context.

```
# fetch var://local/doc TestIt
#
```

- Retrieves the `count.xsl` style sheet from the `store:` directory and stores it in the `count` context.

```
# fetch store:///count.xsl count
#
```

- Retrieves the `simple.xsl` style sheet and store it in the `tmp1` context.

```
# fetch https://sona/TestBase/simple.xsl tmp1
#
```

filter

Adds a `filter` action.

Syntax

filter *input-context stylesheet*

Parameters

input-context
Specifies the context that contains the document to be filtered. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

stylesheet
Specifies the style sheet to filter the source document, and can take the form of a URL or of a `var://` URL that expands to a URL.

Guidelines

A `filter` action accepts or rejects a document.

Filters differ from XSL transformations in that filters produce no output. Filters are generally used to propose conditions against which a candidate document is evaluated. They result in an accept or reject decision.

Filters are implemented from a Processing Policy. A Processing Policy enables a DataPower service to select an appropriate style sheet with which to filter or transform an input document. A style sheet can be used in conjunction with, or instead of, processing instructions that are in the input document.

Refer to Appendix B, "Processing Policy procedures," on page 1107 for procedural details.

Related Commands

validate

Examples

- Uses the specified style sheet to filter the original input.

```
# filter INPUT store:///filter-1.xsl
#
```

- Uses the style sheet referenced by the `filter` variable in the `tools` context to filter the original input.

```
# filter INPUT var://context/tools/filter
#
```

input-filter

Specifies a decompression algorithm to apply to all incoming traffic before any processing.

Syntax

input-filter {**zip** | **pkzip** | **none**}

Parameters

zip Decompresses all incoming traffic with the ZIP algorithm.

pkzip Decompresses all incoming traffic with the PKZIP algorithm.

none (Default) Performs no decompression on incoming traffic.

Guidelines

Use the **input-filter** command only if all incoming traffic can be compressed with the selected algorithm. Attempts to decompress data that is not compressed data will result in data corruption.

log

Adds a `log` action.

Syntax

log *input-context destination* [*output-context*]

Parameters

input-context
Specifies the context whose contents are to be sent to a target location. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

destination
Specifies the URL of the recipient.

output-context
Optionally specifies the output context.

Guidelines

A `log` action generates a log message that contains the contents of a specified context and sends the message to a target location

Examples

- Sends the contents of the INPUT context to the `www.us.ibm/ragnarok/log` target.

```
# log INPUT http://www.us.ibm/ragnarok/log
#
```

non-xml-processing

Enables processing of non-XML contexts in a Processing Rule.

Syntax

non-xml-processing

no non-xml-processing

Guidelines

Use the **no non-xml-processing** command to disable non-XML processing.

on-error

Adds an `on-error` action.

Syntax

on-error *mode* [*rule*] [*input-context*] [*output-context*]

Parameters

mode Specifies the operational response to an error and takes one of the following forms:

abort Indicates that processing is ceased.

continue
Indicates that processing continues.

rule Optionally specifies an error rule that is executed in the event of an error condition.

input-context
Optionally identifies the input context for the error rule. The default is to use the input context of the failed action.

output-context
Optionally identifies the output context for the error rule. The default is to use the output context of the failed action.

Guidelines

An `on-error` action provides user-defined error-handling procedures.

Examples

- Specifies that rule processing ceases in the event of an error. Calls the `faultProcessing` rule as an error handler.

```
# on-error abort faultProcessing
#
```

output-filter

Specifies a compression algorithm to apply to all outgoing traffic after all processing.

Syntax

output-filter {**zip** | **pkzip** | **none**}

Parameters

zip Compresses all incoming traffic with the ZIP algorithm.

pkzip Compresses all incoming traffic with the PKZIP algorithm.

none (Default) Performs no compression on outgoing traffic.

results

Adds a `results` action.

Syntax

results *context* [*destination*] [*response*]

Parameters

context Specifies the target context, that is the target whose contents are transmitted.

destination
Optionally specifies the destination. In the absence of this argument, the contents of the target context are transmitted to the OUTPUT of the Processing Rule.

response
Specifies the target context that stores the parsed reply. This argument is required when a response is expected. Otherwise, it is not used.

Guidelines

A `results` action transmits the contents of a context to a specified destination.

Examples

- Sends the contents of the INPUT context to the destination of the rule.

```
# results INPUT
#
```

- Sends the contents of the INPUT context to the destination that is referenced by the local `var://local/dest` URL.

```
# results INPUT var://local/dest
#
```

- Sends the contents of the INPUT context to the loopback server for processing. Stores processing results in the `apple` context.

```
# results INPUT http://127.0.0.1:9000/ apple
#
```

results-async

Adds a `results-async` action.

Syntax

results *context destination*

Parameters

context Specifies the target context, which is the target whose contents are sent.

destination
Specifies the destination.

Guidelines

A `results-async` action transmits the contents of a context to a specified destination. This action differs from a `results` action in that `results-async` actions transmit messages asynchronously and that `results-async` actions never expects a response.

Examples

- Sends the contents of the INPUT context to the destination of the rule.

```
# results INPUT
#
```

- Sends the contents of the INPUT context to the destination referenced by the local `var://local/dest` URL.

```
# results INPUT var://local/dest
#
```

- Sends the contents of the INPUT context to the loopback server for processing. Stores results in the `apple` context.

```
# results INPUT http://127.0.0.1:9000/ apple
#
```

rewrite

Adds a `rewrite` action that implements a URL Rewrite Policy.

Syntax

rewrite *name*

Parameters

name Specifies the name of the URL Rewrite Policy to implement.

route-action

Adds a `route-action` action.

Syntax

route-action *input-context url*

route-action *input-context* **dynamic-stylesheet** *name*

Parameters

input-context
Specifies the context whose contents are to be routed by the specified style sheet. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

dynamic-stylesheet
Indicates that the action uses a dynamic style sheet.

url Specifies the style sheet to route the contents of the input context.

Guidelines

A `route-action` action enables routing that is based on a style sheet.

Examples

- Specifies routing against the contents of the `temp1` context using the `route.xsl` style sheet in the `local:` directory.

```
# route-action temp1 local:///route.xsl
#
```

route-set

Adds a `route-set` action.

Syntax

route-set *destination* [*sslProxyProfile*]

Parameters

destination
Specifies a URL that identifies the document destination. This argument can be expressed as a protocol-specific URL or as a `var://` URL that expands to a transport URL.

sslProxyProfile
Optionally specifies the name of an SSL Proxy Profile to establish a secure connection with the destination.

Guidelines

A `route-set` action enables dynamic routing in a Processing Rule by identifying the document destination and by providing optional SSL credentials

Examples

- Specifies a dynamic route expanded from the `dest-1` variable in the `destinations` context. Uses the `DySSL-1` SSL Proxy Profile to provide the credentials that are needed to establish a secure connection.

```
# route-set var://context/destinations/dest-1 DySSL-1
#
```

setvar

Sets the value of a variable.

Syntax

setvar *context variable value*

Parameters

context Specifies the context in which to set the variable.

variable
Specifies the name of the variable and takes the `var://` URL format.

value Assigns the value to the variable.

Guidelines

If the `var://` URL is not local, this value overrides the context that is specified by the *context* argument.

Examples

- Sets the `dest` variable to `http://ragnarok:9010/` in the `INPUT` context.

```
# setvar INPUT var://local/dest http://ragnarok:9010/
#
```

- Sets the `dest` variable to `http://ragnarok:9010/` in the `routing` context. Overrides `INPUT` as the *context* argument.

```
# setvar INPUT var://context/routing/dest http://ragnarok:9010/
#
```

slm

Adds an `slm` action that implements an SLM Policy.

Syntax

slm *input-context name*

Parameters

input-context
Specifies the context to monitor. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

name Specifies the name of an SLM Policy.

Examples

- Assigns the `SLM-1` SLM Policy to the `INPUT` context.

```
# slm INPUT SLM-1
#
```

strip-attachments

Adds a `strip-attachments` action.

Syntax

strip-attachments *context* [*uri*]

Parameters

context Specifies the context from which attachments are stripped.

uri Specifies the attachment to strip.

Guidelines

A `strip-attachments` action removes all or specified attachments from a target context

In the absence of a specified attachment, all attachments are stripped from the target context.

Examples

- Strips all attachments from the `temp1` context.

```
# strip-attachments temp1
#
```

---

type

Enables the dynamic retyping of the current Processing Rule.

Syntax

type {**error-rule** | **request-rule** | **response-rule** | **rule**}

Parameters

error-rule
Identifies the rule as an *error* rule to invoke in response to a fault condition.

request-rule
Identifies the rule as a *request* rule to apply to client requests.

response-rule
Identifies the rule as a *response* rule to apply to server responses.

rule Identifies the rule as a *bidirectional* rule to apply to client requests and to server responses.

Examples

- Classifies the current Processing Rule as a request rule.

```
# type request-rule
#
```

---

unprocessed

Enables data to passthrough subsequent actions of the current Processing Rule in an unprocessed state.

Syntax

unprocessed

no unprocessed

Examples

- Enables unprocessed mode.

```
# unprocessed
#
```

- Disables unprocessed mode.

```
# no unprocessed
#
```

---

validate

Adds a `validate` action.

Syntax

validate *input-context* [*output-context*]

validate *input-context* **attribute-rewrite** *name* [*output-context*]

validate *input-context* **dynamic-schema** *url* [*output-context*]

validate *input-context* **schema** *url* [*output-context*]

validate *input-context* **wsdl** *url* [*output-context*]

Parameters

input-context
Specifies the context whose contents are to be validated.

attribute-rewrite *name*
Specifies the name of the URL Rewrite Policy to rewrite the schema that is referenced by an `xsi:schemaLocation` attribute in the XML document. The rewritten schema reference usually specifies the location of a local, trusted copy of the schema to use for document validation.

dynamic-schema *url*
Regardless of `xsi:schemaLocation` attributes in the document, specifies the use of a dynamically generated schema to use for document validation. *url* identifies the URL of the dynamic schema to use for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

schema *url*
Regardless of `xsi:schemaLocation` attributes in the document, specifies the URL of the schema to for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

schema-rewrite *url*
Specifies the URL of the base schema for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

name Specifies the name of the URL Rewrite Policy to apply to the schema URL. The rewritten URL identifies the schema to use for document validation.

wsdl-url *url*
Regardless of `xsi:schemaLocation` attributes in the document, specifies the URL of the WSDL file that contains the schema for document validation. The value can be expressed as a URL or as a variable that expands to a URL.

output-context
Optionally specifies the output context of the validated document.

Guidelines

The **validate** command adds a `validate` action to the current processing rule. This action defines a policy-based XML schema validation filter.

If no methodology is identified, documents are validated in accordance with `xsi:schemaLocation` attributes in the specific context. Documents that do not contain these attributes are considered valid.

Related Commands

filter

Examples

- Adds a validation action. Validates the XML documents in the INPUT context with instruction in the document. Rewrites the document with the `URL-RW-1` URL Rewrite Policy. Uses the rewritten schema reference to validate the document.

```
# validate INPUT attribute-rewrite URL-RW-1
#
```

- Adds a validation action. Validates XML documents in the INPUT context with the schema that is referenced by the `var://context/schemas/1` variable.

```
# validate INPUT schema var://context/schemas/1
#
```

- Adds a validation action. Validates XML documents in the INPUT context with the local `SchemaOne.xsd` schema. Possibly stores the transformed document in the `Post-Validation` context.

```
# validate INPUT schema store:///SchemaOne.xsd Post-Validation
#
```

---

xform

Adds an `xform` action.

Syntax

xform *input-context URL output-context*

xform *input-context* **dynamic-stylesheet** *object output-context*

Parameters

input-context
Specifies the context that contains the transformed document. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

URL Specifies the style sheet to transform documents. Can be expressed as a local URL, as a remote URL, or as a variable that expands to a location.

dynamic-stylesheet *object*
Specifies the object, for example an XPath Routing Map from which to generate the dynamic style sheet.

output-context
Specifies the context for the transformed document. Specify `OUTPUT` to use the final policy output, which is the transformed client request or transformed server response.

Guidelines

An `xform` action defines a policy-based XSL transform. An `xform` action transforms the document using a specified style sheet.

Related Commands

convert-http

Examples

- Adds a transform action to the Processing Rule. Transforms the original input with the `xform-1.xsl` style sheet. Sends the transformed document to the final output.

```
# xform INPUT store:///xform-1.xsl OUTPUT
#
```

- Adds a transform action to the Processing Rule. Transforms the original input with the style sheet that the `var://stylesheets/1` variable references. Sends the transformed document to the `Step2` context.

```
# xform INPUT var://stylesheets/1 Step2
#
```

- Adds a transform action to the Processing Rule. Transforms the document in the `Step2` context with the style sheet that the `var://stylesheets/5` variable references. Sends the transformed document to the final output.

```
# xform Step2 var://stylesheets/5 OUTPUT
#
```

---

xformbin

Adds an `xformbin` action.

Syntax

xformbin *input-context URL output-context*

xformbin *input-context* **dynamic-stylesheet** *object output-context*

Parameters

input-context
Specifies the context that contains the binary document transformed by this rule. Specify `INPUT` to use the initial policy input, which is the original client request or server response.

URL Specifies the stylesheet used to transform documents. Can be expressed as a local, as a remote URL, or as a variable that expands to a location.

dynamic-stylesheet *object*
Specifies the object, for example an XPath Routing Map from which to generate the dynamic style sheet.

output-context
Specifies the context that receives the transformed document. Specify

OUTPUT to use the final policy output, which is the transformed client request or transformed server response

Guidelines

An xformbin action defines a policy-based transformation that converts binary data to XML

Related Commands

convert-http

Examples

- Adds a transform action to the Processing Rule. Transforms the original binary input to XML with the local binToXML.xsl style sheet. Sends the XML to the final output.

```
# xformbin INPUT store:///binToXML.xsl OUTPUT
#
```

- Adds a transform action to the Processing Rule. Transforms the original binary input to XML with the stylesheet that the var://stylesheets/bin-xform variable is references. Sends the transformed document to the FromBin context.

```
# xformbin INPUT var://stylesheets/bin-xform FromBin
#
```

xformpi

Adds an xformpi action.

Syntax

xformpi *input-context URL output-context*

xformpi *input-context* **dynamic-stylesheet** *object output-context*

Parameters

input-context
Specifies the context that contains the document transformed by this rule. Specify INPUT to use the initial policy input, which is the original client request or server response.

URL Specifies the style sheet to transform documents that lack internal processing instructions. Can be expressed as a local, as a remote URL, or as a variable that expands to a location.

dynamic-stylesheet *object*
Specifies the object, for example an XPath Routing Map, from which to generate the dynamic style sheet.

output-context
Specifies the context that receives the transformed document. Specify OUTPUT to use the final policy output, which is the transformed client request or transformed server response.

Guidelines

Adds an xformpi action; an xformpi action defines a policy-based XSL transformation performed according to processing instructions contained within the candidate XML document.

An `xformpi` action defines a policy-based transform that uses processing instructions in the XML document. An optional style sheet can be specifies when the XML document does not contain processing instructions.

An optional style sheet can be identified to transform documents that lack internal processing instructions.

Related Commands

convert-http

Examples

- Adds a transform to the Processing Rule. Transforms the original input with processing instructions in the XML document. Sends the transformed document to the final output. If the document does not contain processing instructions, uses the `identity.xsl` style sheet to perform the transform.

  ```
  # xformpi INPUT store:///identity.xsl OUTPUT
  #
  ```

Chapter 118. WSRR Server configuration mode

This chapter provides an alphabetic listing of commands that are available in WSRR Server configuration mode. While in this configuration mode, provide the information to access a WSSR server. To enter this configuration mode, use the Global **wsrr-server** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in WSSR Server configuration mode.

password

Specifies WSRR server credentials.

Syntax

password *passphrase*

Parameters

passphrase
Specifies the user password.

Guidelines

Used in conjunction with **username** command to provide the credentials used to access the WSRR Server.

Required when the WSRR server enforces authentication.

Related Commands

username

server-version

Identifies the WSRR server version.

Syntax

server-version {**WSRR_6.0** | **WSRR_6.1**}

Parameters

WSRR_6.0
(Default) Uses WSRR Server, version 6.0.

WSRR_6.1
Uses WSRR Server, version 6.1 or later.

Guidelines

The **server-version** command specifies the version of the WSRR server.

When the value is **WSRR_6.1**, use the WSRR Subscription **fetch-policy-attachments** command to configure the ability to retrieve policy attachments. If enabled, the subscription service can retrieve policy attachments from the registry.

Related Commands

fetch-policy-attachments (WSRR Subscription)

---

soap-url

Specifies the URL to access a WSRR server.

Syntax

soap-url *URL*

Parameters

URL Specifies the URL to access the SOAP API on the WSRR server. This URL takes the following format:

```
http://host:port/URI

https://host:port/URI
```

Guidelines

The **soap-url** command defines the URL to access the SOAP API on the WSRR server. The URL differs by WSRR version.

- The default listening port for HTTP is 9080.
- The default listening port for HTTPS is 9443.

For additional information about the URI for the Core Web service, refer to your version-specific WSRR documentation.

Examples

- Access the default internal HTTP transport port for the Core Web service on a WSRR 6.0.2 server.

  ```
  http://192.168.1.120:9080/WSRRCoreSDO/services/WSRRCoreSDOPort
  ```

- Access the default internal HTTPS transport port for the Core Web service on a WSRR 6.1 server.

  ```
  https://192.168.1.120:9443/WSRR6_1/services/WSRRCoreSDOPort
  ```

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ssl** command assigns an SSL Proxy Profile to support secure communications between the appliance and a remote WSRR server. Meaningful only if the SOAP API URL, as defined by the **soap-url** command, starts with `https:`.

Related Commands

soap-url

---

username

Provides WSRR server credentials.

Syntax

username *name*

Parameters

name Specifies the user name.

Guidelines

Use in conjunction with **password** command to provide the credentials used to access the WSRR Server.

Required when the WSRR server enforces authentication.

Related Commands

password

Chapter 119. WSRR Subscription configuration mode

This chapter provides an alphabetic listing of commands that are available in WSRR Subscription configuration mode. While in this mode, define the WSRR-stored content to which to subscribe. To enter this configuration mode, use the Global **wsrr-subscription** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in WSSR Subscription configuration mode.

fetch-policy-attachments

Indicates whether the subscription service can retrieve external policy attachments for a WSDL subscription.

Syntax

fetch-policy-attachments {**on** | **off**}

Parameters

on (Default) Enables the retrieval of policy attachments.

off Disables the retrieval of policy attachments.

Guidelines

The **fetch-policy-attachments** command indicates whether the subscription service can retrieve external policy attachments for a WSDL subscription. This command is relevant only when **server-version** is **WSRR_6.1**.

Related Commands

server-version (WSRR Server)

method

Specifies the method to synchronize the local copy with the WSRR server.

Syntax

method {**poll** | **manual**}

Parameters

poll (Default) Specifies an automatic, periodic refresh of the subscription by regularly scheduled WSRR queries that request the target subscribed-to resource.

manual
Specifies that synchronization is achieved by direct user-intervention, specifically the issuing of the Global **wsrr-synchronize** command.

Related Commands

refresh-interval, **wsrr-synchronize** (Global)

namespace

Used in conjunction with the **object-name** command to unambiguously identify a subscribed-to WSSR resource.

Syntax

namespace *namespace*

Parameters

namespace
Identifies the namespace of the resource.

Guidelines

Both the resource namespace and name are assigned when a resource, such as a WSDL file, is first loaded to a WSRR, or when a collection of resources is aggregated as a concept.

The identification of the resource namespace is required.

Related Commands

object-name

Examples

- Specifies the resource name and namespace, providing an unambiguous identification of the target resource.

```
# wsrr-subscription Proxy-1
New WSRR Subscription configuration
# namespace http://tonawanda.sr.ibm.com/ValidateInsurance
# object-name InsuranceService.wsdl
#
```

object-name

Used in conjunction with the **namespace** command to unambiguously identify a subscribed-to WSSR resource.

Syntax

object-name *name*

Parameters

name Specifies the name of the resource.

Guidelines

Both the resource name and namespace are assigned when a resource, such as a WSDL file, is first loaded to a WSRR, or when a collection of resources is aggregated as a concept.

The identification of the resource namespace is required.

Related Commands

namespace

Examples

- Specifies the resource name and namespace, which provides an unambiguous identification of the target resource.

```
# wsrr-subscription Proxy-1
New WSRR Subscription configuration
# namespace
 http://tonawanda.sr.ibm.com/ValidateInsurance
# object-name InsuranceService.wsdl
#
```

---

object-type

Identifies a resource type.

Syntax

object-type {**wsdl** | **concept**}

Parameters

wsdl (Default) Identifies the resource as a single WSDL file.

concept
Identifies the resource as a concept. A concept, which is created and maintained by the WSRR administrator, is simply a package for metadata, and can contain one or more WSDL files, possibly along with a number of associated files to include XSD schemas and XML files such as AAA info files.

---

refresh-interval

Specifies the synchronization frequency.

Syntax

refresh-interval *seconds*

Parameters

seconds
Specifies the interval in seconds between synchronization queries. The default is 86400.

Guidelines

A value of 0 disables synchronization. Web Service Proxy services that are dependant on this subscription will fail to reflect changes that are made to WSRR-stored WSDL files.

Related Commands

wsrr-synchronize

server

Specifies the WSSR server object.

Syntax

server *name*

Parameters

name Specifies the name of the WSSR server object

Guidelines

Specifies the WSSR server object, previously created with the **wsrr-server** command that identifies the WSSR Server that stores the subscribed-to resource.

Related Commands

wsrr-server

use-version

Indicates whether the subscription service can query the registry for the version of a WSDL file.

Syntax

use-version {**on** | **off**}

Parameters

on Enables the retrieval of a WSDL file of a specified version.

off (Default) Disables the retrieval of a WSDL file of a specified version.

Guidelines

The **use-version** command indicates whether the subscription service can query the registry for the version of a WSDL file. The WSRR registry maintains a `Version` attribute for WSDL files. This attribute is a user-defined suffix that identifies different versions of a WSDL file. For example, subsequent versions of a WSDL file could be identifies as `1.1` or `2`. If enabled, the value for the **version** command must match exactly the `Version` attribute.

- If disabled and there are multiple versions of the file in the registry, the subscription service uses its internal logic to retrieve one of the WSDL files.
- If enabled and there is no value for the **version** command and there is a WSDL in the registry with a `Version` attribute, the subscription service will not retrieve the WSDL file.

  Note: Do not enable if there is only one version of the WSDL file in the registry.

Related Commands

use-version

version

Specifies the version of a WSDL file.

Syntax

version *version*

Parameters

version Specifies the version of the WSDL file.

Guidelines

The **version** command specifies the version of the WSDL file to retrieve from the WSRR registry. The registry maintains a `Version` attribute for WSDL files.

This command is relevant only when **use-version** is **on** and there is more than one version of the WSDL file in the registry.

Related Commands

use-version

Chapter 120. XACML Policy Decision Point configuration mode

This chapter provides an alphabetic listing of commands that are available in XACML Policy Decision Point (PDP) configuration mode. While in XACML PDP configuration mode, you define the parameters needed to implement XACML-based authorization using an internal PDP. To enter this configuration mode, use the Global **xacml-pdp** command. The Global command creates the named internal PDP, if the PDP does not already exist and enters the new configuration mode.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in XACML PDP configuration mode.

cache-ttl

Assigns a Policy Decision Point (PDP) cache lifetime.

Syntax

cache-ttl *seconds*

Parameters

seconds

Specifies the time, in seconds, that compiled XACML policies are maintained in the PDP cache. Use an integer in the range of 0 to 2678400. The default is 0. A value of 0 indicates that the cache never expires.

Guidelines

There are several ways to control the XACML PDP policy caches.

- Use the **clear pdp cache** command to clear the cache.
- During PDP configuration, use the **cache-ttl** command to specify a cache lifetime.
- When the PDP is used by an AAA Policy for authorization, users can access the XML manager associated with the AAA Policy with the **clear xsl cache** command. This command also clears the compiled XACML policies that are referenced by AAA polices that are supported by the XML manager.
- You can use a URL Refresh Policy whose match conditions match the internal URL `xacmlpolicy:///pdpName` to perform periodic cache refreshes.
  - When the PDP TTL is 0, which indicates that the cache is never expired, the URL Refresh Policy controls cache refresh.
  - When the URL Refresh Policy is the **no-cache** type, XACML policies are never cached regardless of any assigned TTL value.
  - When the URL Refresh Policy is the **protocol-specified** type, the PDP-specified TTL setting governs cache refresh unless its value is 0.
  - When the URL Refresh Policy is the **default** type with a refresh interval setting, the PDP-specified TTL is ignored. The refresh interval set in the URL Refresh Policy governs cache refresh.

– When the URL Refresh Policy is the **no-flush** type with a refresh interval setting, the greater of the URL Refresh Policy refresh interval and PDP-specified TTL governs cache refresh.

Related Commands

clear pdp cache (Global), **clear xsl cache** (Global), **urlrefresh** (Global)

Examples

- Enters XACML Policy Decision Point configuration mode to create the `PDP-orderEntry` XACML PDP. Specifies the compiled XACML policies are cached for 8 hours.

```
# xacml-pdp PDP-orderEntry
XACML Policy Decision Point configuration mode
# cache-ttl 10800
# exit
#
```

---

combining-alg

Selects a policy-combining algorithm

Syntax

combining-alg {**deny-overrides** | **first-applicable** | **only-on-applicable** | **permit-overrides**}

Parameters

deny-overrides
Evaluates each policy in the order that it appears in the XACML policy set. If any policy in the set evaluates to `deny`, the policy combination evaluates immediately to `deny`. In other words a single `deny` takes precedence over other policy evaluations. If all policies are determined to be `NotApplicable`, the policy combination evaluates to `NotApplicable`.

first-applicable
Evaluates each policy in the order that it appears in the XACML policy set. For an individual policy, if the target (resource) evaluates to `TRUE` and the policy conditions evaluate unambiguously to `permit` or `deny`, evaluation is immediately halted, and the policy combination evaluates to the effect of that individual policy. If the individual policy evaluates the target as `FALSE` or the policy conditions as `NotApplicable`, then the next policy in the order is evaluated; if no further policy exists in the order, the policy combination evaluates to `NotApplicable`.

only-one-applicable
Evaluates each policy in the order that it appears in the XACML policy set; unlike the other policy combining algorithms, only-one-applicable must evaluate all policies to render a final evaluation. If after evaluating all policies, no policy is considered applicable by virtue of its target (the requested resource), the policy combination evaluates to `NotApplicable`. If after evaluating all policies, more than one policy is considered applicable by virtue of its target, the policy combination evaluates to `Indeterminate`. If after evaluating all policies, only one single policy is considered applicable by virtue of its target, the policy combination evaluates to the result of evaluating that single policy.

permit-overrides
Evaluates each policy in the order that it appears in the XACML policy set. If any policy in the set evaluates to `permit`, the policy combination evaluates immediately to `permit`. In other words a single `permit` takes precedence over other policy evaluations. If all policies are determined to be `NotApplicable`, the policy combination evaluates to `NotApplicable`.

Guidelines

Meaningful only when the value of the **equal-policies** command is **on**.

A policy-combining algorithm defines a procedure for arriving at an authorization decision given the individual results of evaluation of a set of policies.

OASIS Extensible Access Control Markup Language (XACML) version 2.0 defines the standard policy-combining algorithms.

Related Commands

equal-policies

Examples

- Enters XACML Policy Decision Point configuration mode to create the `PDP-orderEntry` XACML PDP. Specifies deny-overrides as the policy-combining algorithm.

```
# xacml-pdp PDP-orderEntry
New XACML Policy Decision Point configuration
# equal-policies on
# combining-alg deny-overrides
# exit
#
```

dependent-policy

Identifies a dependent file.

Syntax

dependent-policy *URL*

Parameters

URL Specifies the location of the file.

Guidelines

Use the **dependent-policy** command to identify a dependent file. This file that might be required by the PDP to evaluate submitted policies. Invoke this command as often as necessary to obtain required files.

Related Commands

directory, **general-policy**

Examples

- Enters XACML Policy Decision Point configuration mode to create the `PDP-orderEntry` XACML PDP. Specifies the location of a dependent policy file.

```
# xacml-pdp PDP-orderEntry
New XACML Policy Decision Point configuration
# dependent-file http://www.example.com/policy/xacml/mine.xml
# exit
#
```

directory

Identifies a specified local directory as a source of dependent files.

Syntax

directory *name*

Parameters

name Identifies a local directory or subdirectory.

Guidelines

In the specified directories all files with a .xml or .xacml extension are identified as *dependent* policy files.

Related Commands

dependent-files

Examples

- Enters XACML Policy Decision Point configuration mode to create the `PDP-orderEntry` XACML PDP. Specifies the location of a dependent policy directory.

```
# xacml-pdp PDP-orderEntry
New XACML Policy Decision Point configuration
# directory local:///xacml
# exit
#
```

equal-policies

Signals the presence of a top-level comprehensive XACML policy file.

Syntax

equal-policies {**on** | **off**}

Parameters

on Comprehensive file is present.

off (Default) Multiple XACML policy files are in use.

Related Commands

combining-alg

Examples

- Enters XACML Policy Decision Point configuration mode to create the `PDP-orderEntry` XACML PDP. Specifies multiple XACML file usage.

```
# xacml-pdp PDP-orderEntry
New XACML Policy Decision Point configuration
# equal-policies on
# exit
#
```

general-policy

Identifies a comprehensive XACML policy set file

Syntax

general-policy *URL*

Parameters

URL Specifies the location of the file.

Related Commands

dependent-file

Examples

- Enters XACML Policy Decision Point configuration mode to create the PDP-orderEntry XACML PDP. Specifies the location of a general policy file.

```
# xacml-pdp PDP-orderEntry
New XACML Policy Decision Point configuration
# general-policy local:///defPolicy.xacml
# exit
#
```

Chapter 121. XML Firewall configuration mode

This chapter provides an alphabetic listing of commands that are available in XML Firewall configuration mode.

To enter this configuration mode, use the global **xmlfirewall** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in XML Firewall configuration mode.

acl

Assigns an ACL.

Syntax

acl *name*

no acl

Parameters

name Specifies the name of the ACL.

Guidelines

An ACL restricts access to those IP addresses specified by the ACL. You can assign a single ACL to an XML Firewall.

Use the **no acl** command to remove the ACL/XML Firewall assignment.

Related Commands

acl, **allow**, **deny**

attachment-byte-count

Specifies the XML Firewall-specific maximum size for an attached document.

Syntax

attachment-byte-count *bytes*

Parameters

bytes Indicates the maximum number of bytes to allow in any attachment. The default is 2000000000.

Guidelines

Attachments that exceed the specified size result in a failure of the entire transaction. A value of 0 indicates that no size limit is enforced.

Related Commands

attribute-count, **bytes-scanned**, **element-depth**, **firewall-parser-limits**, **request-attachments**, **response-attachments**

---

attribute-count

Defines the XML-Firewall-specific maximum number of attributes associated with a given XML element.

Syntax

attribute-count *count*

Parameters

count Sets the gateway-specific maximum number of attributes. The default is 128.

Guidelines

If firewall-specific parser limitations are enabled by the **firewall-parser-limits** command, the maximum attribute count that is assigned by this command overrides the value that is inherited from the XML Manager that is assigned to the XML Firewall.

Related Commands

attachment-byte-count, **bytes-scanned**, **element-depth**, **firewall-parser-limits**, **max-message-size**, **max-node-size**

Examples

- Sets the maximum attribute count for the current XML Firewall to 512.

```
# xmlfirewall FW-1
XML Firewall configuration mode
# firewall-parser-limits on
# attribute-count 512
#
```

---

back-attachment-format

Specifies the attachment format output to back end servers.

Syntax

back-attachment-format {**dime** | **dynamic** | **mime**}

Parameters

dime Indicates that server attachments are output as DIME-encapsulated documents.

dynamic
Indicates that server attachments are output as deduced from front end content.

mime Indicates that server attachments are output as MIME-encapsulated documents.

Related Commands

front-attachment-format

bytes-scanned

Specifies the maximum scope of the XML parser scanning operation.

Syntax

bytes-scanned *bytes*

Parameters

bytes Specifies the maximum scan in bytes. The default is 4194304.

Guidelines

If firewall-specific parser limits are enabled by the **firewall-parser-limits** command, the maximum byte count that is assigned by this command overrides the value that is inherited from the XML Manager that is assigned to the XML Firewall.

The document scan includes the document itself, plus any external DTD, plus any text that is produced by expanding entity references.

Related Commands

firewall-parser-limits

Examples

- Specifies a maximum document scan of 2 MB.

```
# firewall-parser-limits on
# bytes-scanned 2097152
#
```

default-param-namespace

Specifies the default namespace for parameters made available via the CLI or WebGUI.

Syntax

default-param-namespace *namespace*

Parameters

namespace
Specifies the name of the default namespace.

Guidelines

Parameter-value pairs can be made available to an XML Firewall using the **parameter** command, the WebGUI, or by a received URL query string.

The default namespace for parameters introduced with the CLI or WebGUI is:

```
http://www.datapower.com/param/config
```

The default namespace for parameters introduced by a URL query string is:

```
http://www.datapower.com/param/query
```

Related Commands

parameter, **query-param-namespace**

Examples

- Assigns a default namespace for parameters made available via the CLI or WebGUI.

```
# default-param-namespace
 http://www.somecompany.com/namespaces/
#
```

---

element-depth

Defines the XML-Firewall-specific maximum depth of element nesting in an XML document.

Syntax

element-depth *depth*

Parameters

depth Specifies the gateway-specific maximum depth of element nesting. The default is 512.

Guidelines

If firewall-specific parser limitations are enabled by the **firewall-parser-limits** command, the nesting limit assigned by this command overrides the value which is inherited from the XML Manager assigned to the XML Firewall.

Related Commands

attachment-byte-count, **attribute-count**, **bytes-scanned**, **firewall-parser-limits**, **max-message-size**, **max-node-size**

Examples

- Sets the maximum nesting depth to 128.

```
# firewall-parser-limits on
# element-depth 128
#
```

---

external-references

Defines the XML Firewall-specific handling mode for input documents that contain external references, such as an external entity or external DTD definition.

Syntax

external-references {**allow** | **forbid** | **ignore**}

Parameters

allow Allows and resolves external references.

forbid Forbids external references. An external reference causes the XML parser to abort.

ignore Ignores external DTD references, and replaces external entities with the empty string

firewall-parser-limits

Indicates whether to use firewall-specific parser limitations.

Syntax

firewall-parser-limits {**on** | **off**}

Parameters

on Enables firewall-specific parser limits.

off (Default) Uses the parser limits that are defined for the associated XML Manager.

Guidelines

Parser limits protect against denial-of-service attacks that use malicious XML documents that seek to exhaust system resources. With firewall-specific parser limits enabled, incoming documents are evaluated using the values that are defined by the **attribute-count**, **bytes-scanned**, and **element-depth** commands.

Related Commands

attribute-count, **bytes-scanned**, **element-depth**

Examples

- Enables firewall-specific parser limits.

```
# firewall-parser-limits on
# attribute-count 512
# bytes-scanned 2097152
# element-depth 128
#
```

forbid-external-references (deprecated)

Comments

This command has been deprecated. Use the **external-references** command.

front-attachment-format

Specifies the attachment format received from front end clients.

Syntax

front-attachment-format {**dime** | **dynamic** | **mime**}

Parameters

dime Indicates that client attachments are DIME-encapsulated documents.

dynamic
Indicates that the format if client attachments is deduced from document content.

mime Indicates that client attachments are MIME-encapsulated documents.

Related Commands

back-attachment-format

---

fwcred

Assigns a Firewall Credentials List.

Syntax

fwcred *name*

no fwcred

Parameters

name Specifies the name of an existing Firewall Credentials List.

Guidelines

Assignment of a Firewall Credentials List is optional. A Firewall Credentials List provides a means to specify which keys and certificates are permitted with various cryptographic extension functions to support firewall activities. Without a Firewall Credentials List, all keys and certificates on the appliance are available to support firewall activities.

Before using this commands, a Firewall Credentials List must exist. If needed, create a Firewall Credentials List with the **fwcred** (Crypto) command.

Use the **no fwcred** command to remove the assignment of a Firewall Credentials List to an XML Firewall.

Related Commands

fwcred (Crypto)

---

local-address

Specifies the local interface-port pair to monitor for client requests.

Syntax

local-address *address port*

local-address 0 *port*

Parameters

address Binds the XML Firewall to a single, specific interface-port pair.

0 Binds the XML Firewall to specified port on all enabled interfaces. This configuration is strongly discouraged for production environments.

port Is a port number (within the range 0 to 65535) that binds the XML Firewall to a single, specific interface-port or to this port on all enabled interfaces.

Guidelines

You must specify both a local and remote address and an XML manager when configuring an XML Firewall. Other commands enable enhanced functions, or alter default values.

Related Commands

remote-address, **xml-manager**

Examples

- Binds the current XML Firewall to the specified IP interface-port pair.

```
# local-address 192.168.1.10 45000
#
```

- Binds the current XML Firewall to port 45000 on all enabled DataPower interfaces.

```
# local-address 0 45000
#
```

max-message-size

Specifies the maximum size of SOAP or XML messages to process.

Syntax

max-message-size *kilobytes*

Parameters

kilobytes

Specifies the maximum size of SOAP or XML messages in kilobytes. Use an integer in the range of 0 through 2097151. The default is 0. A value of 0 specifies unlimited size.

Guidelines

Limits the SOAP/XML payload, not the size of the incoming IP packet. Messages in excess of this limit are rejected and an error is reported.

Related Commands

attachment-byte-count, **attribute-count**, **bytes-scanned**, **element-depth**, **firewall-parser-limits**, **max-node-size**

max-node-size

Specifies the maximum size of an XML node accepted by the current XML Firewall.

Syntax

max-node-size *bytes*

Parameters

bytes Specifies the firewall-specific maximum number of bytes to allow in a single parsed XML node before the source XML document is considered malicious and dropped. The default is 0. A value of 0 indicates that no size limits are imposed.

Related Commands

attachment-byte-count, **attribute-count**, **element-depth**, **gateway-parser-limits**, **max-message-size**

---

mime-headers

Enables support for MIME (Multi-Purpose Internet Mail Extensions).

Syntax

mime-headers

no mime-headers

Guidelines

Use the **no mime-headers** command to disable MIME support.

---

monitor-count

Assigns a Count Monitor.

Syntax

monitor-count *name*

no monitor-count *name*

Parameters

name is the name of the message-count monitor assigned to the service.

Guidelines

After completing the configuration of a message count monitor, activate the monitor by assigning it to a service.

Use the **no monitor-count** command to remove the Count Monitor assignment.

Related Commands

monitor-duration, **monitor-service**

Examples

- Assigns the `LogSquelch` message count monitor to the current service.

```
# monitor-count LogSquelch
#
```

- Removes the assignment of the `LogSquelch` message count monitor from the current service.

```
# no monitor-count LogSquelch
#
```

monitor-duration

Assigns a Duration Monitor.

Syntax

monitor-duration *name*

no monitor-duration *name*

Parameters

name is the name of the duration monitor assigned to the service.

Guidelines

After completing the configuration of a duration monitor, activate the monitor by assigning it to a service.

Use the **no monitor-duration** command to remove the Duration Monitor assignment.

Related Commands

monitor-count (Global), **monitor-service** (Global)

Examples

- Assigns the `RateLimit1` duration monitor to the current service.

  ```
  # monitor-duration RateLimit1
  #
  ```

- Removes the assignment of the `RateLimit1` duration monitor from the current service.

  ```
  # no monitor-duration RateLimit1
  #
  ```

monitor-processing-policy

Sets the behavior when a service has multiple monitors.

Syntax

monitor-processing-policy {**terminate-at-first-throttle** | **terminate-at-first-match**}

Parameters

terminate-at-first-throttle
(Default) Monitors will execute in the order in which they are listed. After any monitor either shapes (buffers to delay) or rejects a message, none of the further monitors will execute.

terminate-at-first-match
Monitors will execute in the order in which they are listed. After any monitor matches a message and takes any action, none of the further monitors will execute.

Examples

- Allows only the first matching monitor to execute when a service has multiple monitors attached.

```
# monitor-processing-policy terminate-at-first-match
#
```

---

monitor-service

Assigns a Service Level Monitor (SLM).

Syntax

service-count *name*

no service-count *name*

Parameters

name is the name of the SLM assigned to the service.

Guidelines

After completing the configuration of an SLM, activate the monitor by assigning it to a service.

Use the **no service-count** command to remove the Service Level Monitor assignment.

Related Commands

monitor-count, **monitor-duration**

Examples

- Assigns the `wsdlPortSLM` SLM to the current service.

```
# monitor-service wsdlPortSLM
#
```

- Removes the assignment of the `wsdlPortSLM` SLM from the current service.

```
# no monitor-count LogSquelch
#
```

---

parameter

Makes a parameter available and to style sheets that is uses.

Syntax

parameter *name value*

no parameter *name*

no parameter

Parameters

name Specifies the name of the parameter.

value Specifies the value of the parameter.

Guidelines

The following namespace declaration must be included in a style sheet to enable that style sheet to access parameter-value pairs that are defined by the **parameter** command.

```
xmlns:dpconfig="http://www.datapower.com/param/config"
```

Use the **no parameter** command to remove parameters from the current XML Firewall.

Related Commands

default-param-namespace, **query-param-namespace**

Examples

- Makes two parameters (used for document encryption) available. The default parameter namespace is used.

  ```
  # parameter recipient ALICE
  # parameter type content
  #
  ```

- Makes a parameter available. {http://www.example.com} designates the parameter namespace.

  ```
  # parameter {http://www.example.com}foobar value
  #
  ```

- Makes a parameter available. {} designates no namespace.

  ```
  # parameter {}foobar value
  #
  ```

- Deletes the recipient parameter from the current XML Firewall.

  ```
  # no parameter recipient
  #
  ```

- Deletes all parameters from the current XML Firewall.

  ```
  # no parameter
  #
  ```

---

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal
(Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

---

query-param-namespace

Specifies the default namespace for parameters made available via a URL query string.

Syntax

query-param-namespace *namespace*

Parameters

namespace
Specifies the name of the default namespace.

Guidelines

Parameters can be made available to an XML Firewall using the **parameter** command.

The default namespace for parameters introduced with the CLI or WebGUI is:

```
http://www.datapower.com/param/config
```

The default namespace for parameters introduced by a URL query string is:

```
http://www.datapower.com/param/query
```

Related Commands

default-param-namespace, **parameter**

Examples

- Assigns a default namespace for parameters made available through a URL query string.

```
# query-param-namespace http://www.somecompany.com/namespaces/
#
```

---

remote-address

Specifies the address-port pair of the backend server.

Syntax

remote-address *address port*

remote-address *load-balancer*

remote-address %loopback%

remote-address %dynamic% | *

Parameters

address port
Specifies a dotted decimal IP address or host name with the port (in the range 0 to 65535) that identifies a single, specific server address-port pair.

Sets the XML Firewall type to *static backend*, which means that the XML Firewall supports the single, specified server.

load-balancer
Specifies the name of an existing Load Balancer Group that identifies server address-port pairs of its members.

Sets the XML Firewall type to *static backend*.

%loopback%
Sets the XML Firewall type to *loopback*.

%dynamic% | *
Sets the XML Firewall type to *dynamic backend*, which means that the address of the target server is dynamically extracted from the client request using the `dp:set-target` or `dp:xset-target` extension elements.

Guidelines

The **remote-address** command specifies the address-port pair of the backend server for the current XML Firewall. XML Firewall types differ in how they route client requests to the target server.

You must specify both a local and remote address and an XML manager when configuring an XML Firewall. Other commands enable enhanced functions or alter default values.

Related Commands

local-address, **xml-manager**

Examples

- Specifies the interface-port pair of the web or application server as 10.10.1.100:45000.

```
# remote-address 10.10.1.100 45000
#
```

- Supports multiple servers. Actual server addresses are extracted using the **set-target()** or **xset-target()** extension elements.

```
# remote-address %dynamic%
#
```

- Operating under server control, loops back a received document after performing document processing.

```
# remote-address loopback
#
```

request-attachments

Specifies the processing mode for SOAP attachments in client requests.

Syntax

request-attachment *mode*

Parameters

mode Specifies one of the following keywords to indicate the processing mode for SOAP attachments:

allow Allows messages that contain attachments, and processes *needed* attachments. Needed attachments are buffered, but attachments that are not needed might be streamed directly to output.

Attachments are buffered when an action in the processing rule requests any of the following:

- Needed attachments

- All attachments in the package before the needed attachment
- All attachments in the package for a needed manifest
- All attachments in the package if the package does not contain the needed attachment

reject Rejects messages that contain attachments.

strip (Default) Removes attachments from the message before processing.

streaming
Allows messages that contain attachments in streaming mode, but provides limited processing. Messages in the form of a *SOAP message package*, which is a SOAP with Attachments message, are supported. Processing can be applied individually to each attachment. The appliance does not create a manifest of all attachments. Attachments must be accessed and processed in the order that they appear in the package.

unprocessed
Allows messages that contain attachments, but does not process attachments. This option permits the forwarding of messages that contain large attachments. The root part of the message, which typically contains a SOAP message, is subject to filter and transform actions. No processing of parts other than the root part is possible. Accompanying documents can be passed intact.

Guidelines

The **request-attachment** command specifies the processing mode for attachments in client requests (as defined in RFC 2387). This type of request is a compound object that consists of several interrelated body parts and is the mechanism that is used to support the bundling of attachments in a *SOAP message package*, which is commonly referred to as a SOAP with Attachments message.

Meaningful only, if the value of the **request-type** command is **soap**.

Related Commands

request-type

Examples

- Provides full SOAP with Attachments support.

```
# request-attachments allow
#
```

- Provides partial SOAP with Attachments support.

```
# request-attachments streaming
#
```

request-type

Characterizes the client-originated traffic stream.

Syntax

request-type {**xml** | **soap** | **unprocessed**}

Parameters

xml Characterizes the client-originated traffic stream as raw (unencapsulated) XML.

soap Characterizes the client-originated traffic stream as SOAP.

unprocessed
Characterizes the client-originated traffic stream as non-XML traffic that is not transformed by the XML Firewall.

Guidelines

By default, both the client-originated (request) and server-originated (response) traffic streams are characterized as SOAP.

Related Commands

raw-mode, **response-type**

Examples

- Characterizes client-originated traffic as XML.

```
# request-type xml
#
```

- Characterizes client-originated traffic as SOAP, restoring the default condition.

```
# request-type soap
#
```

response-attachments

Specifies the processing mode for SOAP attachments in server responses.

Syntax

response-attachments *mode*

Parameters

mode Specifies one of the following keywords to indicate the processing mode for SOAP attachments:

allow Allows messages that contain attachments, and processes *needed* attachments. Needed attachments are buffered, but attachments that are not needed might be streamed directly to output.

Attachments are buffered when an action in the processing rule requests any of the following:

- Needed attachments
- All attachments in the package before the needed attachment
- All attachments in the package for a needed manifest
- All attachments in the package if the package does not contain the needed attachment

reject Rejects messages that contain attachments.

strip (Default) Removes attachments from the message before processing.

streaming
Allows messages that contain attachments in streaming mode, but provides limited processing. Messages in the form of a *SOAP message package*, which is a SOAP with Attachments message, are supported. Processing can be applied individually to each attachment. The appliance does not create a manifest of all attachments. Attachments must be accessed and processed in the order that they appear in the package.

unprocessed
Allows messages that contain attachments, but does not process attachments. This option permits the forwarding of messages that contain large attachments. The root part of the message, which typically contains a SOAP message, is subject to filter and transform actions. No processing of parts other than the root part is possible. Accompanying documents can be passed intact.

Guidelines

The **response-attachment** command specifies the processing mode for attachments in server responses (as defined in RFC 2387). This type of request is a compound object that consists of several interrelated body parts and is the mechanism that is used to support the bundling of attachments in a *SOAP message package*, which is commonly referred to as a SOAP with Attachments message.

Meaningful only when the value of the **response-type** command is **soap**.

Related Commands

response-type

Examples

- Provides full SOAP with Attachments support.

```
# request-attachments allow
#
```

- Provides partial SOAP with Attachments support.

```
# request-attachments streaming
#
```

---

response-type

Characterizes the server-originated traffic stream.

Syntax

response-type {**xml** | **soap** | **unprocessed**}

Parameters

xml Characterizes the server-originated traffic stream as raw (unencapsulated) XML.

soap Characterizes the server-originated traffic stream as SOAP.

unprocessed
Characterizes the server-originated traffic stream as non-XML traffic that is not transformed by the XML Firewall.

Guidelines

By default, both the client-originated (request) and server-originated (response) traffic streams are characterized as SOAP.

Related Commands

raw-mode, **request-type**

Examples

- Characterizes server-originated traffic as XML.

```
# response-type xml
#
```

- Characterizes server-originated traffic as SOAP, restoring the default condition.

```
# response-type soap
#
```

root-part-not-first-action

Sets the action to take when the MIME message root part is not first.

Syntax

root-part-not-first-action {**abort** | **buffer** | **process-in-order**}

Parameters

abort Stops the transaction and return an error.

buffer Buffers attachments before the root part into memory. Then processes the root part, buffered attachments, and subsequent attachments.

process-in-order
(Default) Processes the attachments and root part in the order that they appear in the original message. All parts are still processed in streaming mode even though only attachments after the root will be streamed from the network.

Guidelines

When streaming MIME messages, specifies the action to take when the root part is not the first part of the message. If the root part must be first (for example to do conformance checking) and the action is set to **process-in-order**, the attachments up to the root will be buffered.

This command is meaningful only when the value of either the **request-attachments** or **response-attachments** command is **streaming.**

Related Commands

request-attachments, **response-attachments**

soap-schema-url

Assigns a schema used to validate incoming SOAP messages.

Syntax

soap-schema-url *url*

Parameters

url Specifies the URL of the schema file.

Guidelines

When an XML Firewall is in SOAP mode, either on the request or response side, it validates the incoming messages against a W3C Schema that defines a conforming SOAP message.

It is possible to customize which schema is used on a per-firewall basis by using this command; different schemas can be used to accommodate nonstandard configurations or other special cases.

In the absence of an explicit schema assignment, the XML Firewall defaults to:

```
store:///schemas/soap-envelope.xsd
```

Related Commands

request-type, **response-type**

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

no ssl

Parameters

name Specifies the name of the SSL Proxy Profile assigned to the XML Firewall.

Guidelines

Assignment of an SSL Proxy Profile to an XML Firewall is optional. In the absence of an assigned SSL Proxy Profile, the XML Firewall client and server exchanges are accomplished over a nonsecure connection.

An SSL Proxy Profile specifies the SSL operational mode (client, server, or two-way) and identifies the cryptographic resources (key, certificates, and cipher lists) available to the SSL proxy. The SSL Proxy Profile must exist. If it does not exist, create one with the **sslproxy** command.

Use the **no ssl** command to remove the SSL Proxy Profile assignment.

Related Commands

stylesheet-policy, **urlrewrite-policy**, **xml-manager**

---

stylesheet-policy

Assigns a stylesheet policy to the XML Firewall.

Syntax

stylesheet-policy *name*

Parameters

name Specifies the name of a Processing Policy.

Guidelines

Assigning a Processing Policy is optional. In the absence of a Processing Policy, the XML Firewall uses processing instructions (if any) that are in the XML document.

Related Commands

ssl, **urlrewrite-policy**, **xml-manager**

---

type

Specifies the type of the XML Firewall service.

Syntax

type {**dynamic-backend** | **loopback-proxy** | **static-backend**}

Parameters

dynamic-backend
(Default) Sets the XML Firewall type to *dynamic backend*. The address of the target server is extracted from the client request with the `dp:set-target` or `dp:xset-target` extension element.

loopback-proxy
Sets the XML Firewall type to *loopback*.

static-backend
Sets the XML Firewall type to *static backend*. The address of the target server is identified with the **remote-address** command.

Guidelines

The **type** command specifies the mode of the XML Firewall.

- When dynamic backend, the XML Firewall identifies the backend server by examining the request. The XML Firewall processes request and response messages with the processing policy defined with the **stylesheet-policy** command. An SSL Client Profile can be defined with the **ssl** command to communicate with the target server, and an SSL Server Profile can be defined with the **ssl** command to communicate with the client.
- When loopback, the XML Firewall processes request messages with the processing policy defined with the **stylesheet-policy** command and returns the result to the client. No target server is involved. An SSL Server Profile can be defined with the **ssl** command to communicate with the client.
- When static backend, the XML Firewall identifies the backend server using the IP address and port defined with the **remote-address** command. The XML Firewall processes request and response messages with the processing policy defined with the **stylesheet-policy** command. An SSL Client Profile can be defined with the **ssl** command to communicate with the target server, and an SSL Server Profile can be defined with the **ssl** command to communicate with the client.

Do not use the **type** command to create a new XML Firewall. Use it to recast the type of an existing XML Firewall.

Related Commands

remote-address, **ssl**, **stylesheet-policy**

urlrewrite-policy

Assigns a URL Rewrite Policy.

Syntax

urlrewrite-policy *name*

Parameters

name Specifies the name of the URL Rewrite Policy.

Guidelines

Assignment of a URL Rewrite Policy is optional.

Related Commands

urlrewrite

wsdl-file-location

Designates the local WSDL file to provide in response to .NET WSDL requests.

Syntax

wsdl-file-location *url*

Parameters

url Specifies the location of the target WSDL file.

Guidelines

Used when the value of the **wsdl-response-policy** command is **serve** to designate the local WSDL file to provide in response to .NET WSDL requests received via the `http://domain.com/service?wsdl` convention.

Related Commands

wsdl-response-policy

Examples

- Specifies that .NET WSDL requests are responded to by serving the designated local `pseudoProxy.wsdl` file. Such requests are not forwarded to the backend server.

```
# wsdl-response-policy serve
# wsdl-file-location local;///pseudoProxy.wsdl
#
```

wsdl-response-policy

Specifies XML Firewall response to receipt of a .NET WSDL request via the `http://domain.com/service?wsdl` convention.

Syntax

wsdl-response-policy {**intercept** | **off** | **serve**}

Parameters

intercept
Indicates that the XML Firewall rewrites the `wsdl:service/wsdl:port/soap:address` field to point to the proxy.

off (Default) Indicates that the XML Firewall does not touch .NET requests and responses.

serve Indicates that the XML Firewall will serves a local WSDL file without forwarding the request to the backside server.

Guidelines

Specifies XML Firewall response to receipt of a .NET WSDL request via the `http://domain.com/service?wsdl` convention.

Related Commands

wsdl-file-location

Examples

- Specifies that .NET WSDL requests are responded to by serving the designated local `pseudoProxy.wsdl` file. Such requests are not forwarded to the backend server.

```
# wsdl-response-policy serve
# wsdl-file-location local;///pseudoProxy.wsdl
#
```

xml-manager

Assigns an XML Manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of the XML Manager.

Guidelines

An XML Manager obtains and controls resources required by the XML Firewall. You must specify an XML Manager and both a local and remote address to configure an XML Firewall. Other commands enable enhanced functions or alter default values.

Related Commands

ssl, **stylesheet-policy**, **urlrewrite-policy**

Chapter 122. XML Management Interface configuration mode

This chapter provides an alphabetic listing of commands that are available in XML Management Interface configuration mode. To enter this configuration mode, use the Global **xml-mgmt** command.

Many of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in this mode.

local-address

Identifies the local address to monitor for incoming requests.

Syntax

local-address *address*

Parameters

address Specifies an IP address that, in conjunction with the port, identifies the XML Management Interface. The default is 0.0.0.0, which indicates all active addresses.

Guidelines

The **local-address** command specifies the IP address to monitor for incoming requests.

Instead of specifying an IP address, you can specify the name of an existing Host Alias. Local host aliases help to ease migration tasks between machines.

Related Commands

host-alias (Global), **port**

Examples

- Configure the XML Management Interface on port 1080 of the specified interface.

```
# xml-mgmt
Modify XML Management Interface configuration
# local-address 10.10.13.7
# port 1080
#
```

mode

Identifies the local address to monitor for incoming requests.

Syntax

mode *mode*[+*mode*]

Parameters

mode Indicates which modes to enable. Separate multiple modes with the plus sign (+) character. The following keywords are available to indicate the modes to enable:

any — SOAP Management URI
Enables processing of messages received on any (*) URI for legacy applications. One example would be an application posting SOAP management requests to /.

By default, this mode is enabled.

soma — SOAP Configuration Management
Enables support for SOAP Configuration Management. The URI for the SOAP Configuration Management is `/service/mgmt/current`.

By default, this mode is enabled.

v2004 — SOAP Configuration Management (v2004)
Enables support for legacy SOAP Management format. The URI for the SOAP Configuration Management is `/service/mgmt/2004`.

By default, this mode is enabled.

amp — AMP Endpoint
Exposes an Appliance Management Protocol (AMP) endpoint. The URI for the AMP endpoint is `/service/mgmt/amp/1.0`.

By default, this mode is enabled.

slm — SLM Endpoint
Exposes a management endpoint that supports the SLM protocol. The URI for the SLM protocol is `/service/slm/datashare/1.0`. The SLM protocol is used to communicate SLM data between appliances and is not a public web service.

By default, this mode is enabled.

wsm — WS-Management Endpoint
Exposes a management endpoint that supports the WS-Management family of protocols. The URI for the WS-Management endpoint is `/service/ws-management`.

wsdm — WSDM Endpoint
Exposes a management endpoint that supports the WSDM 1.0 family of protocols. The URI for the WSDM 1.0 endpoint is `/service/wsdm10`.

uddi-subscription — UDDI Subscription
Exposes a service endpoint that is UDDI subscription listener web service. This service endpoint must be configured in the UDDI registry as the service endpoint of the subscription. Any number of subscriptions can use this endpoint. This endpoint processes subscription updates for all domains. The URI for the UDDI subscription endpoint is `/service/uddi-subscription`.

Guidelines

The **mode** command indicates which services to enable. For each enabled service, you can use the XML Management Interface to control it. To change the enabled services, specify all services to enable.

When the **mode** command exposes the SLM Endpoint (**slm** keyword), you can use the **slm-peering** command to indicate the frequency to update SLM peers.

Related Commands

slm-peering

Examples

- Changes the default modes to include the WS-Management Endpoint service and the WSDM Endpoint service.

```
# xml-mgmt
Modify XML Management Interface configuration
# mode any+soma+v2004+amp+slm+wsm+wsdm
#
```

port

Identifies the local port to monitor for incoming requests.

Syntax

port *port*

Parameters

port Identifies the listening port on the appliance that monitors SOAP/XML management traffic. The default is 5050.

Related Commands

local-address

Examples

- Configure the XML Management Interface on port 1080 of the specified interface.

```
# xml-mgmt
Modify XML Management Interface configuration
# local-address 10.10.13.7
# port 1080
#
```

slm-peering

Specifies the frequency to issue SLM peer-group updates.

Syntax

slm-peering *seconds*

Parameters

seconds
Specifies the number of seconds between SLM peer-group updates. The default is 10.

Guidelines

The **slm-peering** is required when the XML Management Interface exposes the SLM Endpoint.

Related Commands

`mode`

Examples

- Changes the interval between updates of SLM peer groups to 25 seconds.

```
# xml-mgmt
Modify XML Management Interface configuration
# slm-peering 25
#
```

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ssl** command identifies the SSL Proxy Profile to assign instead of the default profile. The SSL Proxy Profile must already exist in the current application domain To create a new SSL Proxy Profile, use the Global **sslproxy** command.

Related Commands

sslproxy (Global)

Examples

- Changes the assignment of the SSL Proxy Profile to `mgmtProxy`.

```
# xml-mgmt
Modify XML Management Interface configuration
# ssl mgmtProxy
```

---

user-agent

Assigns a User Agent.

Syntax

user-agent *name*

Parameters

name Specifies the name of an existing User Agent. The default is `xml-mgmt`.

Guidelines

The **user-agent** command identifies the User Agent to assign instead of the `xml-mgmt` User Agent. The User Agent must already exist in the current application domain To create a new User Agent, use the Global **user-agent** command.

user-agent (Global)

Examples

- Changes the assignment of the User Agent to `mgmtAgent`.

```
# xml-mgmt
Modify XML Management Interface configuration
# user-agent mgmtAgent
```

Chapter 123. XML Manager configuration mode

An XML Manager obtains and manages XML documents, style sheets, and other document resources on behalf of one or more services. An XML Manager also provides the following functionality:

- Set manager-associated limits on the parsing of XML documents
- Enable document caching
- Perform extension function mapping
- Enable XML Manager-based schema validation
- Schedule an XML Manager-initiated Processing Rule

This chapter provides an alphabetic listing of commands that are available in XML Manager configuration mode. To enter this configuration mode, use the Global **xmlmgr** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in XML Manager configuration mode.

loadbalancer-group

Associates a Load Balancer Group with an XML Manager.

Syntax

loadbalancer-group *name*

no loadbalancer-group

Parameters

name Specifies the name of the Load Balancer Group.

Guidelines

Assignment of a Load Balancer Group to an XML Manager and the subsequent assignment of that XML Manager to a DataPower service enables the DataPower service to utilize redundant server-side resources.

Use the **no loadbalancer-group** command to remove the association of the Load Balancer Group to the XML Manager.

schedule-rule

Schedules the invocation of a Processing Rule.

Syntax

schedule-rule *name* [*frequency*]

no schedule-rule *name*

Parameters

name Specifies the name of an existing Processing Rule.

frequency
Specifies the frequency of rule invocation.

Guidelines

The **schedule-rule** command schedules the XML Manager to run the specified Processing Rule. In the absence of the *frequency* argument, the rule is run a single time.

Use the **no schedule-rule** command to cancel rule execution.

---

user-agent

Assigns a User Agent.

Syntax

user-agent *name*

no user-agent

Parameters

name Specifies the name of the User Agent.

Guidelines

You can assign only one User Agent to an XML Manager.

Use the **no user-agent** command to remove the User Agent assignment from the XML Manager.

Chapter 124. XML Parser Limits configuration mode

This chapter provides an alphabetic listing of commands that are available in XML Parser Limits configuration mode. To enter this configuration mode, use the Global **xml parser limits** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in XML Parser Limits configuration mode.

attribute-count

Specifies the maximum per-element attribute count.

Syntax

attribute-count *limit*

Parameters

limit Specifies the maximum number of attributes to allow within an XML element. The default is 128.

Related Commands

bytes-scanned, **element-depth**

bytes-scanned

Specifies the maximum scope of the scan operation for the XML parser.

Syntax

bytes-scanned *bytes*

Parameters

bytes Specifies the maximum scan in bytes. The default is 4 MB.

Guidelines

The document scan includes the document itself, plus any external DTD, plus any text that is produced by expanding entity references.

Related Commands

attribute-count, **element-depth**

element-depth

Specifies the maximum depth of element nesting.

Syntax

element-depth *limit*

Parameters

limit Specifies the maximum nesting depth. The default is 512.

Related Commands

attribute-count, **bytes-scanned**

external-references

Defines the handling mode for input documents that contain external references.

Syntax

external-references {**allow** | **forbid** | **ignore**}

Parameters

allow Allows and resolves external references.

forbid Forbids external references. An external reference causes the XML parser to abort.

ignore Ignores external DTD references, and replaces external entities with the empty string.

Guidelines

An external reference is an external entity or external DTD definition.

forbid-external-references (deprecated)

Comments

This command is deprecated. Use the **external-reference** command.

max-node-size

Specifies the maximum size of a single XML node in kilobytes.

Syntax

max-node-size *kilobytes*

Parameters

kilobytes

Specifies the maximum message node size in kilobytes. The default is 0. This value means that no size limit is applied to incoming message nodes.

Related Commands

attachment-byte-count, **attribute-count**, **element-depth**, **gateway-parser-limits**, **max-message-size**

Chapter 125. XPath Routing Map configuration mode

This chapter provides an alphabetic listing of commands that are available in XPath Routing Map configuration mode. To enter this configuration mode, use the Global **xpath-routing** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in XPath Routing Map configuration mode.

namespace-mapping

Adds XML namespace data.

Syntax

namespace-mapping *prefix URI*

Parameters

prefix Specifies the namespace prefix.

URI Specifies the namespace location.

Examples

- Creates the `Departmental` XPath Routing Map and enters XPath Routing Map configuration mode. Adds XML namespace data.

```
# xpath-routing Departmental
XPath Routing Map configuration mode
# namspace-mapping dp http://www.datapower.com/extensions
#
```

rule

Creates a forwarding rule.

Syntax

rule *expression host port* {**on** | **off**}

Parameters

expression
Specifies an XPath expression.

host Identifies the destination by host name or by IP address.

port Identifies a port number on the destination host. Use an integer in the range of 0 to 65535.

on | off
Indicates whether to use a secure connection to the destination.

on Uses a secure connection

off Uses a nonsecure connection

Guidelines

The **rule** command creates XPath-based forwarding rule by adding an XPath expression and associated forwarding data to the current XPath Routing Map. That is, the selection of a target Web or application server is based upon the contents of the XML document being processed.

With XPath-based forwarding enabled, the appliance scans incoming XML documents for XPath expressions. If the document contains an XPath expression contained in an XPath Forwarding Rule, the document is forwarded to the specified server-port pair.

You can add multiple rules to an XPath Routing Map.

Related Commands

xpath-routing

Examples

- Creates the `Departmental` XPath Routing Map and enters XPath Routing Map configuration mode. Adds four XPath-based forwarding rules to `Departmental`. XML documents that contain elements that match the XPath expression `//request[dept='dev']` are forwarded to server 127.0.0.1:8001 via a secure connection; documents that contain elements that match the expression `//request[dept='sales']` are forwarded to 127.0.0.1:8002 via a secure connection; documents that contain elements that match the expression `//request[dept='admin']` are forwarded to 192.168.1.100:8000 via an insecure connection; all other documents are forwarded to 192.168.1.10:8000 via a secure connection.

```
# xpath-routing Departmental
# rule //request[dept='dev'] 127.0.0.1 8001 on
# rule //request[dept='sales'] 127.0.0.1 8002 on
# rule //request[dept='admin'] 192.168.1.100 8000 off
# rule true() 192.168.1.10 8000 on
#
```

Chapter 126. XSL Coprocessor Service configuration mode

This chapter provides an alphabetic listing of commands that are available in XSL Coprocessor Service configuration mode. To enter this configuration mode, use the Global **xslcoproc** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are also available in XSL Coprocessor Service configuration mode.

cache-relative-url

Enables or disables the caching of relative client URLs in the document cache.

Syntax

cache-relative-url {**on** | **off**}

Parameters

on Enables caching of relative client URLs.

off (Default) Disable caching of relative client URLs. Alternatively, use the **no cache-relative-url** command.

Guidelines

connection-timeout

Specifies the amount of time that the XSL Coprocessor maintains an idle connection.

Syntax

connection-timeout *seconds*

Parameters

seconds

Specifies the number of seconds the XSL Coprocessor maintains an idle connection. Use a value in the range of 3 through 7200. The default is 60.

crypto-extensions

Controls XSL Coprocessor access to cryptographic extensions.

Syntax

crypto-extensions {**on** | **off**}

Parameters

on (Default) Enables access to cryptographic extensions.

off Disable access to cryptographic extensions. Alternatively, use the **no crypto-extensions** command.

default-param-namespace

Specifies the default namespace for stylesheet parameters.

Syntax

default-param-namespace *namespace*

Parameters

namespace
Specifies the name of the default namespace. The default namespace for parameters is:

```
http://www.datapower.com/param/config
```

intermediate-result-timeout

Specifies the time that the XSL Coprocessor retains an unused intermediate-result node set.

Syntax

intermediate-result-timeout *seconds*

Parameters

seconds
Specifies the number of seconds that an XSL Coprocessor retains an unused intermediate-result node set). Use an integer in the range of 1 through 600. The default is 20.

ip-address

Specifies the local IP address to monitor for incoming traffic.

Syntax

ip-address {*address* | **0**}

Parameters

address Specifies the IP address (primary or secondary) of a DataPower Ethernet interface.

0 Indicates all DataPower Ethernet interfaces.

Guidelines

In conjunction with the **port** command, identifies the IP addresses and ports that the XSL Coprocessor service monitors.

Related Commands

port

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current XSL Coprocessor service monitor.

```
# xslcoproc proxy-1
XSL Coprocessor Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

port

Specifies the local port monitored for incoming traffic.

Syntax

port *port*

Parameters

port Specifies the port.

Guidelines

Use the **port** command to change the port that is assigned with the **ip-address** command.

Related Commands

ip-address

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current XSL Coprocessor service monitor.

```
# xslcoproc proxy-1
XSL Coprocessor Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal
(Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies the name of the SSL Proxy Profile assigned to the XSL Coprocessor Service.

Guidelines

The SSL Proxy Profile enables a secure Coprocessor-to-server connection.

---

stylesheet-policy

Assigns a Processing Policy.

Syntax

stylesheet-policy *name*

Parameters

name Specifies the name of the Processing Policy.

Guidelines

This command enables the implementation of a static processing policy applied to all server-originated documents.

The assigned policy is used in place of any processing instructions contained within the server-originated documents.

Examples

- Assigns the `processHeaders` Document Processing Rule to the current XSL Coprocessor.

```
# stylesheet-policy processHeaders
#
```

---

stylesheet-rule

Assigns a Processing Rule.

Syntax

stylesheet-rule *name*

Parameters

name Specifies the name of the Processing Rule.

Guidelines

The **stylesheet-rule** command allows assigns a global Processing Rule to an XSL Coprocessor. After the assignment is made, any Java input is transformed by the specified rule and the result becomes the Java transform output.

The assignment of a Processing Rule allows the Java client code to instantiate a minimal identity transformer and invoke statically configured rule-based transformations with little overhead.

For example, consider the following two examples. This command sequence creates the global `coprocXform` Processing Rule.

```
rule coprocXform
   xform INPUT http://10.10.1.66/Diff_part_1.xsl x1
   xform x1 http://10.10.1.66/Diff_part_2.xsl x2
   xform x2 http://10.10.1.66/Diff_part_3.xsl OUTPUT
exit
```

Now create the `coproc-1` XSL Coprocessor and assign the `coprocXform` Processing Rule to it.

```
xslcoproc coproc-1
   local-address 0 6003
   xml-manager mgr1
   stylesheet-rule coprocXform
exit
```

The following minimal Java client code invokes the rule and performs the multistep transformation.

```
// create new "identity" transformer
transformer = factory.newTransformer();

// transform input file and send result to
stdout

transformer.transform(
   new StreamSource(args[0]),
   new StreamResult(System.out));
```

This command sequence creates the global `SignSoapSec` Processing Rule.

```
rule signSoapSec
   xform INPUT store:///sign-soapsec.xsl OUTPUT
exit
```

Now create the `coprocCrypto` XSL Coprocessor and assign the `SignSoapSec` Processing Rule to the Coprocessor.

```
xslcoproc coprocCrypto
   local-address 0 6003
   xml-manager mgr1
   stylesheet-rule signSoapSec
   crypto-extensions on
exit
```

The `coprocCrypto` XSL Coprocessor signs the INPUT provided by the client and returns the signed OUTPUT to the client.

The following minimal Java client code enables the DataPower appliance to function as a SOAP signature engine.

```
// create new "identity" transformer
transformer = factory.newTransformer();

// set stylesheet parameter
transformer.setParameter ("keypair", "ALICE");

// transform input file and send result to
```

```
stdout
transformer.transform(
   new StreamSource(args[0]),
   new StreamResult(System.out));
```

Examples

- Assigns the `coprocXform` Processing Rule to the current XSL Coprocessor.

```
# stylesheet-rule coprocXform
#
```

urlrewrite-policy

Assigns a URL Rewrite Policy.

Syntax

urlrewrite-policy *name*

Parameters

name Specifies the name of the URL Rewrite Policy.

use-client-resolver

Enables or disables the use of a client-based (JAXP) URI-resolver to resolve external URLs.

Syntax

use-client-resolver {**on** | **off**}

Parameters

on (Default) Enables client-based URI-resolver.

off Disable client-based URI-resolver. Alternatively, use the **no use-client-resolver** command.

xml-manager

Assigns an XML Manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of the XML Manager.

Chapter 127. XSL Proxy Service configuration mode

This chapter provides an alphabetic listing of commands that are available in XSL Proxy Service configuration mode. To enter this configuration mode, use the Global **xslproxy** command.

All of the commands that are listed in "Common commands" on page 2 and most, but not all, of the commands that are listed in Chapter 129, "Monitoring commands," on page 1053 are available in XSL Proxy Service configuration mode.

acl

Assigns an Access Control List (ACL).

Syntax

acl *name*

no acl

Parameters

name is the name of the ACL assigned to the current XSL Proxy.

Guidelines

The **acl** command assigns an ACL to an XSL Proxy. An ACL restricts access to the current XSL Proxy to those IP addresses that are specified by the ACL.

Use the **no acl** command to remove the ACL assignment from the XSL Proxy.

Related Commands

acl (Global), **allow**, **deny**

Examples

- Assigns the `ACL-2` ACL to the current XSL Proxy.

```
# acl ACL-2
#
```

- Removes the ACL.

```
# no acl
#
```

default-param-namespace

Specifies the default namespace for parameters made available via the command line or WebGUI.

Syntax

default-param-namespace *namespace*

Parameters

namespace
Specifies the name of the default namespace.

Guidelines

The default namespace for parameters introduced with the CLI or WebGUI is:

```
http://www.datapower.com/param/config
```

Related Commands

parameter, **query-param-namespace**

Examples

- Assigns a default namespace for parameters made available to the current XSL Proxy via the command line or WebGUI.

```
# default-param-namespace
 http://www.somecompany.com/namespaces/
#
```

ip-address

Specifies the local IP address to monitor for incoming traffic.

Syntax

ip-address {*address* | **0**}

Parameters

address Specifies the IP address (primary or secondary) of a DataPower Ethernet interface.

0 Indicates all DataPower Ethernet interfaces.

Guidelines

In conjunction with the **port** command, identifies the IP addresses and ports that the XSL Proxy service monitors.

Related Commands

port

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current XSL Proxy service monitor.

```
# xslproxy proxy-1
XSL Proxy Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

monitor-count

Assigns a message-count monitor to an XSL Proxy.

Syntax

monitor-count *name*

no monitor-count

Parameters

name Specifies the name of the message-count monitor assigned to the XSL Proxy.

Guidelines

After completing the configuration of a message-count monitor, you activate the monitor by assigning it to an XML Firewall or XSL Proxy.

Use the **no monitor-count** command to remove the message count monitor assignment from the XSL Proxy.

Related Commands

monitor-count (Global)

Examples

- Assigns the `LogSquelch` incremental monitor to the current XSL Proxy.

```
# monitor-count LogSquelch
#
```

- Removes the `LogSquelch` incremental monitor.

```
# no monitor-count LogSquelch
#
```

---

monitor-duration

Assigns a duration monitor to an XSL Proxy.

Syntax

monitor-duration *name*

no monitor-duration *name*

Parameters

name Specifies the name of the duration monitor assigned to the XSL Proxy.

Guidelines

After completing the configuration of a duration monitor, you activate the monitor by assigning it to an XML Firewall or XSL Proxy.

Use the **no monitor-duration** command to remove the duration monitor/XSL Proxy assignment.

Related Commands

monitor-duration (Global)

Examples

- Assigns the RateLimit1 duration monitor to the current XSL Proxy.

```
# monitor-duration RateLimit1
#
```

- Removes the assignment of the RateLimit1 duration monitor.

```
# no monitor-duration RateLimit1
#
```

monitor-processing-policy

Sets the behavior when a service has multiple monitors.

Syntax

monitor-processing-policy {**terminate-at-first-throttle** | **terminate-at-first-match**}

Parameters

terminate-at-first-throttle
(Default) Monitors will execute in the order in which they are listed. After any monitor either shapes (buffers to delay) or rejects a message, none of the further monitors will execute.

terminate-at-first-match
Monitors will execute in the order in which they are listed. After any monitor matches a message and takes any action, none of the further monitors will execute.

Examples

- Allows only the first matching monitor to execute when a service has multiple monitors attached.

```
# monitor-processing-policy terminate-at-first-match
#
```

parameter

Makes a parameter available for the processing policy.

Syntax

parameter *name value*

no parameter *name*

Parameters

name Specifies the name of the parameter made available to the current XSL Proxy.

value Specifies the value of the parameter.

Guidelines

The following namespace declaration must be included in any style sheet to enable that style sheet to access parameter-value pairs that are defined by the **parameter** command.

```
xmlns:dpconfig="http://www.datapower.com/param/config"
```

Use the **no parameter** command to delete a parameter and associated value.

Related Commands

default-param-namespace, **query-param-namespace**

Examples

- Makes a parameter-value pair available to the current XSL Proxy.

  ```
  # parameter foo BAR
  #
  ```

- Makes a single parameter-value pair available to the current XSL Proxy. {http://www.example.com} designates the parameter namespace.

  ```
  # parameter {http://www.example.com}foobar test
  #
  ```

- Makes a single parameter-value pair available to the current XSL Proxy. {} designates no namespace.

  ```
  # parameter {}foobar value
  #
  ```

- Deletes the foo parameter.

  ```
  # no parameter foo
  #
  ```

- Deletes all parameters.

  ```
  # no parameter
  #
  ```

---

priority

Assigns a service-level priority.

Syntax

priority {**low** | **normal** | **high**}

Parameters

low Receives below normal priority for scheduling or for resource allocation.

normal
(Default) Receives normal priority for scheduling or for resource allocation.

high Receives above normal priority for scheduling or for resource allocation.

---

port

Specifies the local port monitored for incoming traffic.

Syntax

port *port*

Parameters

port Specifies the port.

Guidelines

Use the **port** command to change the port that is assigned with the **ip-address** command.

Related Commands

ip-address

Examples

- Specifies `10.10.13.35:23000` as the local IP address-port that the current XSL Proxy service monitor.

```
# xslproxy proxy-1
XSL Proxy Service configuration mode
# ip-address 10.10.13.35
# port 23000
#
```

---

query-param-namespace

Specifies the default namespace for parameters made available to the current XSL Proxy via a URL query string.

Syntax

query-param-namespace *namespace*

Parameters

namespace
Specifies the name of the default namespace.

Guidelines

The default namespace for parameters introduced by a URL query string is:

```
http://www.datapower.com/param/query
```

Related Commands

default-param-namespace, **parameter**

Examples

- Assigns a default namespace for parameters made available to the current XSL Proxy through a URL query string.

```
# query-param-namespace
 http://www.somecompany.com/namespaces/
#
```

---

remote-address

Specifies the IP address-port pair of the backend servers.

Syntax

remote-address *address port*

remote-address *load-balancer*

remote-address %dynamic%

remote-address %loopback%

remote-address {**%proxy%** | *}

Parameters

address port
Specifies a dotted decimal IP address or host name with the port (in the range 0 to 65535) that identifies a single, specific server address-port pair.

Sets the XSL Proxy type to static backend, which means that the XSL Proxy supports the single, specified server.

load-balancer
Specifies the name of an existing Load Balancer Group that identifies server address-port pairs of its members.

Sets the XSL Proxy type to static backend.

%dynamic%
Sets the XSL Proxy type to dynamic backend, which means that the address of the target server is dynamically extracted from the client request using the `dp:set-target` or `dp:xset-target` extension elements.

%loopback%
Sets the proxy type to loopback.

%proxy% | *
Sets the XSL Proxy type to strict proxy, which means that the address of the target server is extracted from HTTP directives.

Guidelines

XSL Proxy types differ in how they route and forward client requests to the target server.

You must specify both a local and remote address when configuring an XSL Proxy.

Related Commands

local-address, **xml-manager**

Examples

- Specifies a backend server at 10.10.1.100:45000, which is a static-backend proxy.

```
# remote-address 10.10.1.100 45000
#
```

---

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

no ssl

Parameters

name Specifies the name of an existing SSL Proxy Profile.

Guidelines

The **ssl** command assigns an SSL Proxy Profile to an XSL Proxy. In the absence of an assigned SSL Proxy Profile, the XSL Proxy uses nonsecure connections in client and server exchanges.

An SSL Proxy Profile is required only to establish the following secure connections:

- Between the XSL Proxy and the server
- Between the XSL Proxy and the client
- Between the XSL Proxy and the client, and between the XSL Proxy and the server

The SSL operational modes are:

- `forward`
- `reverse`
- `two-way`

Use the **no ssl** command to remove the SSL Proxy Profile assignment.

Related Commands

stylesheet-policy urlrewrite-policy xml-manager

Examples

- Assigns the `SSL1` SSL proxy to the current XSL Proxy.

```
# ssl SSL1
#
```

- Removes the assignment of an SSL Proxy Profile from the current XSL Proxy.

```
# no ssl
#
```

---

stylesheet-policy

Assigns a Processing Policy.

Syntax

stylesheet-policy *name*

Parameters

name Specifies the name of a Processing Policy to use in XSL transforms.

Guidelines

You need not specify a Processing Policy when configuring an XSL Proxy.

In the absence of a named Processing Policy, the XSL Proxy will use the processing instructions included in the XML document.

Related Commands

ssl, **urlrewrite-policy**, **xml-manager**

Examples

- Assigns the `WebQuery` Stylesheet Policy to the current XSL Proxy.

```
# stylesheet-policy WebQuery
#
```

type

Specifies the XSL Proxy type.

Syntax

type {**loopback-proxy** | **static-backend** | **strict-proxy**}

loopback-proxy
Sets the XSL Proxy type to loopback.

static-backend
(Default) Sets the XSL Proxy type to static backend. The address of the target server is identified with the **remote-address** command.

strict-proxy
Sets the XSL Proxy type to static backend. The address of the target server is identified with HTTP directives in the request header.

Guidelines

The **type** command specifies the mode of the XSL Proxy.

- When loopback, the XSL Proxy processes request messages with the processing policy defined with the **stylesheet-policy** command and returns the result to the client. No target server is involved. An SSL Server Profile can be defined with the **ssl** command to communicate with the client.
- When static proxy, the XSL Proxy identifies the backend server using the IP address and port defined with the **remote-address** command. The XSL Proxy processes request and response messages with the processing policy defined with the **stylesheet-policy** command. An SSL Client Profile can be defined with the **ssl** command to communicate with the target server, and an SSL Server Profile can be defined with the **ssl** command to communicate with the client.
- When strict proxy, the XSL Proxy identifies the backend server by examining the request headers. The XSL Proxy processes request and response messages with the processing policy defined with the **stylesheet-policy** command. An SSL Client Profile can be defined with the **ssl** command to communicate with the target server, and an SSL Server Profile can be defined with the **ssl** command to communicate with the client.

Do not use the **type** command to create a new XSL Proxy. Use it to recast the type of an existing XSL Proxy.

Related Commands

remote-address, **ssl**, **stylesheet-policy**

Examples

- Changes the XSL Proxy type to strict proxy.

```
# type strict-proxy
#
```

urlrewrite-policy

Assigns a URL Rewrite Policy.

Syntax

urlrewrite-policy *name*

Parameters

name Specifies the name of the URL Rewrite Policy to assign.

Guidelines

You need not specify a URL Rewrite Policy when configuring an XSL Proxy.

Related Commands

ssl, **stylesheet-policy**, **xml-manager**

Examples

- Assigns the `Rw1` URL Rewrite Policy to the current XSL Proxy.

```
# urlrewrite-policy Rw1
#
```

xml-manager

Assigns an XML Manager.

Syntax

xml-manager *name*

Parameters

name Specifies the name of an existing XML Manager.

Guidelines

You must assign an XML manager to the current XSL Proxy.

Related Commands

ssl stylesheet-policy urlrewrite-policy

Examples

- Assigns the `mgr1` XML Manager to the current XSL Proxy.

```
# xml-manager mgr1
#
```

Chapter 128. z/OS NSS Client configuration mode

This chapter provides an alphabetic listing of commands that are available in z/OS NSS Client configuration mode. To enter this configuration mode, use the Global **zos-nss** command.

client-id

Client ID for registration with the NSS server.

Syntax

client-id *string*

Parameters

string Specifies the client ID to be used for registration with the NSS server. Minimum length is 1. Maximum length is 24.

Valid characters are:
- `a` through `z`
- `A` through `Z`
- `0` through `9`
- `_` (underscore)
- `-` (dash)

Embedded spaces are invalid.

Guidelines

The **client-id** command identifies the client ID to register the DataPower appliance with the NSS server. The NSS client ID is a unique string used by the NSS Server to track clients. The client ID does not have to correspond to any preexisting object. It is provided by the NSS client to the server at the time of registration. If another client attempts to register with the same client ID to the same NSS Server, the NSS server will send a heartbeat to the first client. If the first client responds to the heartbeat, the second client's registration will be rejected. If the first client does not respond, the connect to the first client will be severed and the second client will be registered.

Related Commands

host, **port**

host

Identifies the NSS server by host name or IP address.

Syntax

host *host*

Parameters

host Specifies the host name or IP address of an NSS server.

Guidelines

The **host** command identifies the NSS server by domain name or IP address. In conjunction with the **port** command, identifies the host and listening port of the NSS server. The NSS server must have the `XMLAppliance` discipline support enabled.

Related Commands

port

Examples

- Sets `nssServer1.datapower.com` as the host name to identify the target NSS server.

```
# zos-nss nssClient1
New zOS NSS Client configuration
# host nssServer1.datapower.com
```

- Uses the IP address `192.168.1.109` to identify the target NSS server.

```
# zos-nss nssClient2
New zOS NSS Client configuration
# host 192.168.1.109
```

---

password

Password to use to authenticate as in SAF on the NSS server.

Syntax

password *password*

Parameters

password

Specifies the password to use to authenticate to the NSS server. Minimum length is 1. Maximum length is 8.

Valid characters are:

- `a` through `z`
- `A` through `Z`
- `0` through `9`
- `_` (underscore)
- `-` (dash)

Embedded spaces are invalid.

Guidelines

The **password** command specifies the password to use in conjunction with the value provided by the **user-name** command.

Related Commands

user-name

Examples

- Sets `testUser` as the user with the password `pword` as the credentials to authenticate.

```
# zos-nss nssClient1
New zOS NSS Client configuration
# user-name testUser
# password pword
```

port

Identifies the listening port on the NSS server.

Syntax

port *port*

Parameters

port Specifies a destination port on the NSS server.

Guidelines

The **port** command is used in conjunction with the **host** command to identify the listening port on the specified NSS server.

Related Commands

host

Examples

- Sets the host to IP address 192.168.1.109 on port 4159.

```
# zos-nss nssClient1
New zOS NSS Client configuration
# host 192.168.1.109
# port 4159
```

ssl

Assigns an SSL Proxy Profile.

Syntax

ssl *name*

Parameters

name Specifies the name of an existing SSL Proxy Profile to use for a secure connection.

Guidelines

An SSL Proxy Profile must be assigned to the z/OS NSS Client to use secure communication. The SSL Proxy Profile must exist in the current application domain. To create an SSL Proxy Profile, use the Global **sslproxy** command.

Related Commands

sslproxy (Global)

system-name

Specifies a name that the NSS server uses to identify the NSS client.

Syntax

system-name *string*

Parameters

string Specifies a name for the NSS client. Minimum length is 1. Maximum length is 8.

Valid characters are:
- a through z
- A through Z
- 0 through 9
- _ (underscore)
- - (dash)

Embedded spaces are invalid.

Guidelines

The **system-name** command specifies a name that the NSS server uses to identify the NSS client. NSS server commands identify NSS clients by system name in the output when displaying information for connected NSS clients. When an NSS client is a z/OS system, this field contains the system name, such as MVS046.

Examples

- Sets DP001 as the system-name.

```
# zos-nss nssClient1
New zOS NSS Client configuration
# system-name DP001
```

---

user-name

Specifies a user name to authenticate as in SAF on the NSS server

Syntax

user-name *user*

Parameters

user Specifies a user name to use to authenticate to the NSS server. Minimum length is 1. Maximum length is 8.

Valid characters are:
- a through z
- A through Z
- 0 through 9
- _ (underscore)
- - (dash)

Embedded spaces are invalid.

Guidelines

The user name must match an existing user ID on the NSS Server. Use in conjunction with the **password** command.

Related Commands

password

Examples

- Sets the user name to `testUser` with the password `pword` as the credentials to authenticate on the NSS server.

```
# zos-nss nssClient1
New zOS NSS Client configuration
# user-name testUser
# password pword
```

Chapter 129. Monitoring commands

This chapter provides an alphabetic listing of all commands for status objects and for configuration objects. These commands are available in all configuration modes, unless otherwise indicated.

show aliases

Displays a list of expanded macros.

Syntax

show aliases [*name*]

Parameters

name Specifies the name of an existing alias.

Guidelines

Alias names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

alias

show application-security-policy

Displays a list of Web Application Security Policies.

Syntax

show application-security-policy [*name*]

Parameters

name Specifies the name of an existing Application Security Policy.

Guidelines

Policy names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

application-security-policy

show audit-log

Displays the contents of the audit log.

Syntax

show audit-log [**-np**]

show audit-log [**-np**] **user**

show audit-log [**-np**] **date**

show audit-log [**-np**] **time**

show audit-log [**-np**] **address**

Parameters

-np Indicates no pagination.

user Sorts the events in the audit log alphabetically by user name.

address
Sorts the events in the audit log numerically by IP address.

date Sorts the events in the audit log numerically by date.

time Sorts the events in the audit log numerically by time.

Guidelines

The **show audit-log** command displays the audit log with or without pagination. Use the **-np** keyword to display the audit log without pagination. When displaying the audit log, use the **user**, **date**, **time**, or **address** keyword to indicate the sorting sequence.

The **date** and **time** keywords are equivalent.

Examples

- Displays the events in the audit log in date sequence.

```
# show audit-log date
#
```

---

show audit-search

Searches the audit log and displays matching events.

Syntax

show audit-search [**-np**] **user** *name*

show audit-search [**-np**] **date** *start* [*end*]

show audit-search [**-np**] **time** *start* [*end*]

show audit-search [**-np**] **address** *address*[/*netmask*]

Parameters

-np Indicates no pagination.

user *name*
Displays events in the audit log for the specified user.

date *start* [*end*]
Displays events in the audit log from the specified start date to optional end date. Without an end date, displays events to the most recent date.

time *start* [*end*]
Displays events in the audit log from the specified start time to the optional end time. Without an end time, displays events until 23:59:59.

address *address*[/*netmask*]
Displays events in the audit log for the specified IP address or, if using a netmask, the IP address range.

Guidelines

The **show audit-search** command searches the audit log and displays events that match the specified criteria. Use the **-np** parameter to indicate no pagination for the output.

Examples

- Display events in the audit log for the `joesmith` account one screen at a time.

```
# show audit-search user joesmith
#
```

- Display events in the audit log from February 10, 2008 onward one screen at a time.

```
# show audit-search time 20080210
#
```

- Display events in the audit log from IP address 10.10.10.15 upward as one continuous list.

```
# show audit-search -np address 10.10.10.15
#
```

- Display events in the audit log from the IP address in the range of 10.10.10.0 through 10.10.10.255 one screen at a time.

```
# show audit-search address 10.10.10.0/24
#
```

show chkpoints

Displays information about the available checkpoint configuration files.

Syntax

show chkpoints

Related Commands

remove chkpoint, **save chkpoint**

show clock

Displays the current date, time, and system uptime.

Syntax

show clock

Related Commands

date, **show time**, **time**

show compact-flash (Type 9235)

Displays the configuration of the compact flash.

Syntax

show compact-flash `cf0`

Context

Available only of Type 9235 appliances with the compact flash as auxiliary storage.

show conformancepolicy

Displays configuration settings for Conformance Policy objects.

Syntax

show conformancepolicy [*name*]

Context

Available in Global configuration mode only.

show cpu

Displays average CPU utilization data for the last 10 seconds, 1 minute, 10 minutes, 1 hour, and 1 day.

Syntax

show cpu

show crypto

Displays SSL configuration information.

Syntax

show crypto [**tree** | **type**]

Parameters

tree (Default) Displays the information in a tree format with each SSL Proxy Profile as the root.

type Displays an object-oriented view of the cryptographic resources.

Context

Available in Crypto configuration mode only.

show default-gateway

Displays the IP address of the default gateway.

Syntax

show default-gateway

Related Commands

ip default-gateway

Context

Available in Interface configuration mode only.

show deployment-policy

Displays configuration settings for Deployment Policy objects.

Syntax

show deployment-policy [*name*]

Context

Available in Global configuration mode only.

show documentcache

Displays the current size of the document cache and the number of documents cached.

Syntax

show documentcache *xml-mgr*

Parameters

xml-mgr
Specifies the name of an existing XML Manager.

Related Commands

documentcache

show domain

Displays configuration settings for domains.

Syntax

show domain [*name*]

show domains

Displays status information about each domain.

Syntax

show domains

Guidelines

The **show domains** command displays the following status information:

Needs Save
Indicates whether the domain contains unsaved changes.

File Capture
Indicates whether the XML File Capture utility is enabled in the domain.

Debug Log
Indicates whether the domain is using the **debug** logging level.

Probe Enabled
Indicates whether one or more services in the domain has the probe enable.

Diagnostics
Indicates that diagnostic tracing is enabled. Diagnostic tracing applies to all domain.

Note: The only time that tracing should be enabled is at the explicit direction of IBM Support.

---

show file

Displays a specified printable file.

Syntax

show file *URL*

Parameters

URL Identifies the URL of the file to display. The URL takes the *directory*:///*filename* format, where:

directory
Specifies a directory on the appliance. Refer to "Directories on the appliance" on page xxiv for details.

filename
Specifies the name of a file in the directory.

Guidelines

You cannot use the show file command to display a file stored in the `cert:` directory.

Related Commands

copy, **dir**, **delete**

---

show firmware

Displays the current firmware version, with image type and installation date.

Syntax

show firmware

Guidelines

The **show firmware** command provides a subset of the details of the **show firmware-version** command and includes information about whether the current firmware image is the primary or secondary installation image and the date on which the image was installed.

Related Commands

show firmware-version

show firmware-version

Displays the current firmware version, without image type and installation date.

Syntax

show firmware-version

Guidelines

The **show firmware-version** command provides information about the current firmware version. This command provides the same details as the **show version** command, but it does not provide the versions of the licenses that are available with the **show library-version** command.

The **show firmware-version** command does not include information about whether the current firmware image is the primary or secondary installation image or the date on which the image was installed. For these details, use the **show firmware** command.

Related Commands

show firmware, **show library-version**, **show version**

show http

Displays HTTP configuration details for a specified DataPower service, or displays transaction counts and times for all DataPower services.

Syntax

show http [*name*]

Parameters

name Specifies the name of an existing XML Firewall or XSL Proxy.

Guidelines

In the absence of an argument, displays transaction counts and times for all XML Firewall and XSL Proxy services.

Related Commands

http, **inject**, **suppress**, **version**

show interface

Displays transaction details for all Ethernet interfaces.

Syntax

show interface

Guidelines

The **show interface** command displays the following inofrmation:

- The IP address for the interface
- Statistics about received transactions:
  - Number of kilobytes/second
  - Number of packets
  - Number of aggregated errors
- Statistics about transmitted transaction:
  - Number of kilobytes/second
  - Number of packets
  - Number of aggregated errors

The count for aggregated errors could indicate the number of packet drops and collisions.

Related Commands

interface, **show interface mode**

---

show interface mode

Displays configuration information about all Ethernet interfaces.

Syntax

show interface mode

Guidelines

The **show interface mode** displays the following configuration information about the Ethernet interfaces:

- The connection status
- Indicates whether the physical mode (speed and duplex) is negotiated
- The interface speed
- The MAC address

Related Commands

interface, **show interface mode**

---

show ip

Displays IP information about the current Ethernet interface.

Syntax

show ip address

show ip domains

show ip hosts [*hostname*]

show ip name-servers

Parameters

address
Displays the primary and standby addresses, if any, that are assigned to the current interface.

domains
Displays the IP domain search suffix table.

hosts *hostname*
Displays all host-to-IP address mappings, or display this information about the specified host.

name-servers
Displays the addresses of the DNS servers.

Guidelines

The **show ip address** command displays the IP addresses (primary and standby) for the current interface. If the current interface is a member of a standby group, displays the virtual IP address that is assigned to the group and the group number.

The **show ip domains** command displays the IP domain search suffix table.

The **show ip host** command displays all current host-IP address maps, or display this information about the specified host. Displays each map with an alphabetic prefix:

S Designates a static mapping (entered from the CLI)

D Designates a dynamic mapping (learned from the DNS)

The **show ip name-servers** command displays the IP addresses of the DNS servers.

Context

show ip address command is available in Interface configuration mode only.

Related Commands

ip address, **ip domain**, **ip host**, **ip name-server**, **standby**

---

show library-version

Displays the versions of the installed libraries.

Syntax

show library-version

Guidelines

The **show library-version** command provides information about the versions of the installed libraries. This command provides the same details as the **show version** command, but it does not provide information about the firmware that is available with the **show firmware-version** command.

Related Commands

show firmware-version, **show version**

show license

Displays the installed licenses.

Syntax

show license

Guidelines

The **show license** command provides information about which of the available licenses are enabled. Some licenses are available because of the type of DataPower appliance, but some licenses must be purchased to be enabled.

Licenses cannot be updated. The license must be enabled as part of the initial purchase.

show loadbalancer-group

Displays all current Load Balancer Group objects or a specific object.

Syntax

show loadbalancer-group [*group*]

Parameters

group Specifies the name of an existing Load Balancer Group.

Related Commands

loadbalancer-group, **show loadbalancer-status**

show loadbalancer-status

Displays the status of all Load Balancer Group objects.

Syntax

show loadbalancer-status

Related Commands

loadbalancer-group, **show loadbalancer-group**

show log

Displays the appliance default log.

Syntax

show log

Guidelines

Use **show log** to display the default log. Use **show logging** to display the default or other logs.

Related Commands

show logging

---

show logging

Displays a specified appliance log.

Syntax

show logging *log-name* [*pcre*]

show logging archive

show logging category [*log-category*]

show logging encrypt

show logging event

show logging format

show logging priority

show logging sign

show logging target [*target-name*]

show logging timestamp

show logging type [*log-type*]

show logging upload

Parameters

log-name [*pcre*]
Specifies the name of a log, and optionally displays only the events from the specified log that match the specified expression.

archive
Displays a list of available archival methods.

category [*log-category*]
Displays summary information about all active log categories, or displays summary information about the specified log category.

encrypt
Displays a list of available log encryption methods.

event Displays a list of supported event classes.

format
Displays a list of supported log formats.

priority
Displays a list of event priorities

sign Displays a list of supported digital signature algorithms.

target [*target-name*]
Displays summary information about all active log targets, displays detailed information about a specific log target.

timestamp
Displays a list of timestamp formats

type [*log-type*]
Displays summary information about all available logging types, or displays detailed information about the specified logging type.

upload
Displays a list of available upload methods.

Guidelines

Use **show log** to display the default log.

Related Commands

show log

---

show loglevel

Displays the log-level.

Syntax

show loglevel

Guidelines

Log messages are characterized (in descending order of criticality) as `emergency`, `alert`, `critical`, `error`, `warning`, `notice`, and `info`. The log levels can also be expressed as integer values, with 0 equating to `emergency` and 6 equating to `info`.

Related Commands

loglevel

---

show matching

Displays a list of all matching rules or displays a specific Stylesheet Policy matching rule.

Syntax

show matching [*matching-rule*]

Parameters

matching-rule
Specifies the name of an existing Matching Rule.

Guidelines

In the absence of an argument, displays a list of all matching rules. When issued with an argument, displays the contents of the named matching rule.

Related Commands

matching

show memory

Displays memory usage.

Syntax

show memory

Guidelines

The **show memory** command displays memory usage. This command is also available from the **diag** (login) mode.

Output

```
• # show memory
        Memory Usage: 10 %
        Total Memory: 4149324 kbytes
         Used Memory: 433761 kbytes
         Free Memory: 3715563 kbytes
    Requested Memory: 503216 kbytes
  XG4 Resource Usage: 1 %
  #
```

show netarp

Displays the address resolution table

Syntax

show netarp

Related Commands

arp

show ntp-refresh

Displays the refresh status for the current NTP server.

Syntax

show ntp-refresh

Guidelines

The **show ntp-refresh** command provides the following details about the current NTP server, if configured:

- The IP address of the last NTP server that was contacted
- The results of the contact
- The time after the refresh

Related Commands

ntp, **show ntp-service**

show ntp-service

Displays the refresh interval for the current NTP server.

Syntax

show ntp-service

Related Commands

ntp, **show ntp-refresh**

show password-map

Displays the Password map.

Syntax

show password-map

Context

Available in Crypto configuration mode only.

Related Commands

password-map

show radius

Displays RADIUS configuration settings.

Syntax

show radius

Related Commands

id, **retries**, **server**, **timeout**

show raid-phys-disks (Type 9235)

Displays the status of the physical disks in the RAID volume.

Syntax

show raid-phys-disks

Context

Available only of Type 9235 appliances with the hard disk array as auxiliary storage.

show raid-volume (Type 9235)

Displays the configuration of the Hard Disk Array.

Syntax

show raid-volume `raid0`

Context

Available only of Type 9235 appliances with the hard disk array as auxiliary storage.

show raid-volumes (Type 9235)

Displays the status of the disks in the hard disk array.

Syntax

show raid-volumes

Context

Available only of Type 9235 appliances with the hard disk array as auxiliary storage.

show route

Displays the appliance routing table.

Syntax

show route

Related Commands

ip default-gateway

show rule

Displays a list of named transformation or filtering rules.

Syntax

show rule

Related Commands

rule (Global)

show running-config

Displays the running configuration as a set of commands

Syntax

show running-config

Related Commands

write memory

show sensors (deprecated)

This command is deprecated.

Syntax

show sensors

Guidelines

The **show sensors** command has been deprecated. Use one of the following commands:

- **show sensors-fans**
- **show sensors-other**
- **show sensors-temperature**
- **show sensors-voltage**

show sensors-fans

Displays the values for sensors that read the speed of the fans.

Syntax

show sensors-fans

Guidelines

The **show sensors-fans** command provides values for sensors that read fan speed.

show sensors-other

Displays the values for sensors that read nonnumeric (enumerated) values.

Syntax

show sensors-other

Guidelines

The **show sensors-other** command provides truth values for the intrusion switch, each of the two power supply modules, and the battery. If the appliance uses the hard disk array configuration, provides truth values for each of the two disks in the array.

- A value of `true` indicates that the condition exists.
- A value of `false` indicates that the conditions does not exist.

For the intrusion switch, the value indicates whether it has been tripped.

For each power supply, the values indicate the following conditions:

- AC Lost
- Fan Slow
- High Temperature
- Output Failure
- Over-Temperature

For each hard disk in the array and the battery, the values indicate the following conditions:

- Fault
- Present

show sensors-temperature

Displays the values for sensors that read temperatures.

Syntax

show sensors-temperature

Guidelines

The **show sensors-temperature** command provides values for sensors that read temperatures. These sensors provide the temperature of the air flowing through the system and of key components in the system.

show sensors-voltage

Displays the values for sensors that read voltage.

Syntax

show sensors-voltage

Guidelines

The **show sensors-voltage** command provides values for sensors that read voltages. These sensors provide the voltage of the power supplies and for other parts of the system.

show services

Displays a list of all active services.

Syntax

show services

Guidelines

Use the **show services** command to display a concise list of all active services.

The local IP field of the list contains the IP address and port (in the form *address*:*port*) where the service is active. An IP address of 0.0.0.0 indicates that the service is active on all interfaces.

Related Commands

cli telnet, **httpserv**, **tcpproxy**, **xmlfirewall**, **xslcoproc**, **xslproxy**

show simple-rate-limiter

Displays a list of Simple Rate Limiter Policies.

Syntax

show simple-rate-limiter [*name*]

Parameters

name Specifies the name of an existing Policy.

Guidelines

Policy names are followed by the associated command or command sequence.

In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

simple-rate-limiter

show snmp

Displays SNMP configuration data

Syntax

show snmp

Related Commands

port, **show system**, **version**

show standby

Displays failover configuration information.

Syntax

show standby

Related Commands

standby

show startup-config

Displays the contents (commands) of the configuration with which the appliance was last booted or restarted.

Syntax

show startup-config

Guidelines

Displays the configuration with which the appliance was booted. The startup configuration might not reflect the current operational state, or the startup configuration designated by the boot config file.

Context

Available in Global Configuration mode only.

show startup-errors

Displays the contents on the startup error log, a listing of syntactical errors found in the startup configuration.

Syntax

show startup-errors

Guidelines

Should the appliance find an error, it displays and logs the following message:

```
Notice: startup config contains errors.
```

You can access the startup error log to locate the source in the startup configuration.

Context

Available in Global configuration mode only.

---

show statistics

Displays information about XSL transformations.

Syntax

show statistics

Guidelines

displays XSL transformation statistics.

Uptime
shows the number of days, hours, and minutes since the appliance was last restarted.

Open Connections
shows the number of currently open connections, to include XML Firewall, XSL Proxy, XSL Coprocessor, TCP Server, HTTP Server, Telnet Server, and CLI Connections services.

Memory use
shows the amount of memory currently being used and total memory.

Connections accepted
shows the number of connections over the last 10 seconds, 1 minute, 10 minutes, 1 hour, and 24 hours.

CPU usage (%)
shows the average CPU utilization over the last 10 seconds, 1 minute, 10 minutes, 1 hour, and 24 hours.

executions
shows the number of times a specific style sheet was executed over the last 10 seconds, 1 minute, 10 minutes, 1 hour, and 24 hours.

---

show stylepolicy

Displays the default style sheets and matching rules for a Processing Policy.

Syntax

show stylepolicy [*processing-policy-name*]

Parameters

processing-policy-name
Specifies the name of an existing Processing Policy.

Guidelines

When issued without an argument, displays data for all Processing Policy objects. When issued for a specific Processing Policy, displays data for the specified Stylesheet Policies.

For each Processing Policy, the results contain the following details:
- The name of the Processing Policy
- The default style sheet for transforms
- The default style sheet for SOAP filtering
- The match patterns for the Processing Policy
- The transform rules for the Processing Policy

Related Commands

httpmatch, **stylepolicy**, **urlmatch**

---

show stylesheet

Displays compilation information about a specified stylesheet.

Syntax

show stylesheet *XML-manager URL*

Parameters

XML-manager
Specifies the name of an XML Manager.

URL Specifies the local URL of the style sheet.

Guidelines

Use this command to obtain the URL of the target style sheet.

Related Commands

show stylesheets

---

show stylesheets

Displays data about style sheets cached by an XML Manager.

Syntax

show stylesheets [*xml-mgr*]

Parameters

xml-mgr
Specifies the name of an XML Manager.

Guidelines

Displays information for style sheets:

OK Indicates a valid (compilable) style sheet

BAD Indicates an erroneous style sheet (possibly indicating internal errors

within the style sheet, or a corrupted document, possibly caused by transient network conditions at the time the style sheet was accessed)

DUPLICATE
Usually indicates a temporary style sheet that was generated during a pipeline transformation

PENDING
Indicates that the style sheet is being retrieved or undergoing compilation

Related Commands

show stylesheet, **xsl cache size**

---

show system

Displays values for the SNMP system group in addition to a appliance-specific serial number.

Syntax

show system

Related Commands

contact, **location**, **name**, **show snmp**

---

show tcp

Displays a list of current TCP connections.

Syntax

show tcp

---

show throttle

Displays the throttle settings that specify the behavior of the DataPower appliance when faced with a user-defined low memory condition.

Syntax

show throttle

Guidelines

The appliance monitors its memory usage and reacts to low memory conditions by first refusing to accept new connections. If the refusal to accept new connections does not free sufficient memory, the appliance responds by restarting itself.

When free memory falls below the *throttle threshold* (a measure of free memory expressed as a percentage of total memory), the appliance refuses to accept new connections. If the amount of free memory does not rise above the throttle threshold in the specified timeout (expressed in seconds), the appliance restarts. If free memory falls below the *kill threshold* (also a measure of free memory expressed as a percentage of total memory), the appliance restarts immediately

Related Commands

throttle

show throughput

Displays interface-specific traffic counts.

Syntax

show throughput

show time

Displays the current date, time, and appliance uptime.

Syntax

show time

Related Commands

clock, **show clock**

show urlmap

Displays a list of all URL maps (along with match patterns contained within the map) or displays the contents of a specific URL map.

Syntax

show urlmap [*URL-map*]

Parameters

URL-map
Specifies the name of an existing URL map.

Guidelines

Must be used in Global configuration mode. In the absence of an argument, this command displays a list of all URL maps. When issued with an argument, this command displays the contents of the named URL map.

Related Commands

urlmap

show urlrefresh

Displays a list of Stylesheet-Refresh Policies.

Syntax

show urlrefresh [*policy*]

show useragent

Displays HTTP proxy agent configuration details.

Syntax

show useragent

show usergroups

Displays a list of User Groups and the commands suites to which group members are granted access.

Syntax

show usergroups

Related Commands

usergroup

show usernames

Displays a list of all current user accounts with associated access levels.

Syntax

show usernames

Related Commands

show users, **username**

show users

Displays a list of all users currently logged into the appliance.

Syntax

show users

Related Commands

show usernames

show version

Displays the current version of the firmware and libraries.

Syntax

show version

Guidelines

The **show version** command provides the combined details of the **show firmware-version** and **show library-version** commands.

Related Commands

show firmware-version, **show library-version**

show web-application-firewall

Displays a list of Web Application Firewalls.

Syntax

show web-application-firewall [*name*]

Parameters

name Specifies the name of an existing Web Application Firewall.

Guidelines

Firewall names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

web-application-firewall

---

show webapp-error-handling

Displays a list of Web Application Error Handling Policy objects.

Syntax

show webapp-error-handling [*name*]

Parameters

name Specifies the name of an existing Web Application Error Handling Policy.

Guidelines

Policy names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

webapp-error-handling

---

show webapp-gnvc

Displays a list of Web Application Name Value Profile objects.

Syntax

show webapp-gnvc [*name*]

Parameters

name Specifies the name of an existing Web Application Name Value Profile.

Guidelines

Profile names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

webapp-gnvc

show webapp-request-profile

Displays a list of Web Application Request Profile objects.

Syntax

show webapp-request-profile [*name*]

Parameters

name Specifies the name of an existing Web Application Request Profile.

Guidelines

Profile names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

webapp-request-profile

show webapp-response-profile

Displays a list of Web Application Response Profile objects.

Syntax

show webapp-response-profile [*name*]

Parameters

name Specifies the name of an existing Web Application Response Profile.

Guidelines

Profile names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

webapp-response-profile

show webapp-session-management

Displays a list of Web Application Session Management Policy objects.

Syntax

show webapp-session-management [*name*]

Parameters

name Specifies the name of an existing Web Application Session Management Policy.

Guidelines

Policy names are followed by the associated command or command sequence. In the absence of the optional name argument, the system displays a list of all current command macros.

Related Commands

webapp-session-management

---

show wsrr-server

Displays the configuration of WSRR servers.

Syntax

show wsrr-server [*name*]

Parameters

name Identifies the name of the target WSRR Server object.

Guidelines

In the absence of the optional argument, the command provides configuration details of all WSRR Server objects.

Context

Available in Global configuration mode only.

Related Commands

wsrr-server (Global)

---

show wsrr-subscription

Displays the configuration of WSRR subscriptions.

Syntax

show wsrr-subscription *name*

Parameters

name Specifies the name of the target WSRR Subscription object.

Guidelines

In the absence of the optional argument, the command provides configuration details of all WSRR Subscription objects.

Context

Available in Global configuration mode only.

Related Commands

wsrr-subscription (Global), **show wsrr-subscription-status**, **show wsrr-subscription-service-status**

show wsrr-subscription-status

Displays operational details of WSRR subscriptions.

Syntax

show wsrr-subscription-status [*name*]

Parameters

name Specifies the name of the target WSRR Subscription object.

Guidelines

This command provides the following operational details:

Subscription
The name of the WSRR subscription object that is assigned during the configuration of the subscription.

Status The status of the last synchronization. The status is one of the following values:
- `Error`
- `Okay`
- `Synchronizing`

Synchronization Method
The synchronization method assigned during the configuration of the subscription. The method is one of the following values:
- `Poll`
- `Manual`

Refresh Interval
The refresh interval assigned during the configuration of the subscription. This value is meaningful only when the synchronization method is `Poll`.

WSDLs
The number of WSDL files covered by the subscription.

In the absence of the optional argument, the command provides operational details of all WSRR Subscription objects.

Context

Available in Global configuration mode only.

Related Commands

wsrr-subscription (Global), **show wsrr-subscription**, **show wsrr-subscription-service-status**

show wsrr-subscription-service-status

Displays WSDL file data for WSRR subscriptions.

Syntax

show wsrr-subscription-service-status [*name*]

Parameters

name Identifies the name of the target WSRR Subscription object.

Guidelines

This command provides the following WSDL file data:

Subscription
The name of the WSRR subscription object that is assigned during the configuration of the subscription.

bsrURI
The WSRR-assigned document identifier.

WSDL Name
The WSRR-assigned logical name of the WSDL file.

Location
The WSRR-assigned location. Normally, the value is the file name.

Description
The WSRR-assigned file or resource description.

In the absence of the optional argument, the command provides configuration details of all WSRR Subscription objects.

Context

Available in Global configuration mode only.

Related Commands

wsrr-subscription (Global), **show wsrr-subscription**, **show wsrr-subscription-status**

---

show xmlfirewall

Displays configuration details for XML Firewall objects.

Syntax

show xmlfirewall [*name*]

Parameters

name Specifies the name of an existing XML Firewall.

Guidelines

In the absence of an argument, displays configuration data for all firewall services. When issued with an argument, displays configuration data for the named target firewall only.

Related Commands

http, **inject**, **suppress**, **version**

---

show xmlmgr

Displays a list of XML Manager objects.

Syntax

show xmlmgr

show xslcoproc

Displays information about XSL Coprocessors.

Syntax

show xslcoproc

show xslproxy

Displays configuration details for XML Proxy objects.

Syntax

show xslproxy [*name*]

Parameters

name Specifies the name of an existing XSL Proxy.

Guidelines

In the absence of an argument, displays configuration data for all proxies. When issued with an argument, it displays configuration data for the named target proxy only.

show xslrefresh

Displays information about Stylesheet-Refresh Policy objects.

Syntax

show xslrefresh [*xml-manager*]

Appendix A. Working with variables

Variables can be used in most context, except PIPE. To use a variable, you must create it with the setvar action. A setvar action creates a variable in a specified context and assigns it a value.

Note: You can view the value of variables for a transaction with the multistep probe. Edit the DataPower service to enable the multistep probe. After enabling the multistep probe and recording transactions, you can view variables and their values.

There are the following distinct variable types, each expressed in the var://*URL* format:

var://local/*variable*
: A local context variable to addresses a variable called *variable* in the default (current) context. The following example transforms the document in the tmp1 context with a style sheet that is referenced by the stylesheet-1 variable (also in the tmp1 context) and stores the transformed document in the tmp2 context:

```
xform tmp1 var://local/stylesheet-1 tmp2
```

The local context does not persist beyond the scope of the multistep transaction. A multistep transaction can include both a request component and a response component. The local context cannot be accessed by any object outside of the scope of the multistep transaction. In other words, the service cannot read and use the variable.

A local context variables can be user-defined or based on an extension variable. For a complete list of the available extension variables, refer to "Extension variables" on page 1098.

var://context/*context*/*variable*
: Addresses a variable called *variable* in a context called *context*. The following example transforms the document in the tmp1 context with a style sheet that is referenced by the stylesheet-1 variable (in the apple context) and stores the transformed document in the tmp2 context:

```
xform tmp1 var://context/apple/stylesheet-1 tmp2
```

A named context does not persist beyond the scope of the multistep transaction. A multistep transaction can include both a request component and a response component. The local context cannot be accessed by any object outside of the scope of the multistep transaction. In other words, the service cannot read and use the variable.

Note: Creating variables in a named context is the recommended approach. This form decouples the variable from the input and output contexts and allows the variable to be accessed from any step in a multistep scope.

A named context variables can be user-defined or based on an extension variable. For a complete list of the available extension variables, refer to "Extension variables" on page 1098.

`var://service/variable`
: Address a variable that is made available to a service (such as HTTP or XSL Co-Processor) that is attached to a multistep session. The majority of service variables are read-only and cannot be set.

  For a complete list of the available service variables, refer to "Service variables."

`var://system/variable`
: Addresses a global variable that is available in all contexts. System variables persist beyond the multistep scope and can be read by other objects in the system. If the content of a variable needs to be read or set outside the scope of the multistep process, use a system variable.

  For a complete list of the available global system variables, refer to "System variables" on page 1100.

Note: Table 35 on page 1101 lists all of the variables that are available when using a DataPower appliance.

Service variables

Service variables enable the setting and retrieval of pieces of state that usually reflect the state of the current transaction.

The available service variables are separated alphabetically into the following categories:
- General service variables that are available to all DataPower services
- Service variables that are available only to Multi-Protocol Gateway services and to Web Service Proxy services
- Configuration services
- Load balancer service
- MQ-specific services

General service variables

This section contains information about general variables in alphabetic order by permission category. General variables are available to all services. Table 18 lists the names and permission for these variables.

Table 18. Names and permissions for variables that are available to all DataPower services

Variable name	Permission
`var://service/soap-fault-response`	Read-write
`var://service/system/ident`	Read-only
`var://service/system/status/status-enumeration`	Read-only

Read-only variables

`var://service/system/ident`
: Gets the system identification information. This information includes the product number, the model, the configured name of the DataPower appliance, and the serial number.

`var://service/system/status/status-enumeration`
: Gets the contents of a specific status object. You must include the enumeration value of the status object. Refer to the `StatusEnum` type in the `store:///xml-mgmt.xsd` schema for the list of the available status objects.

Use the following variable to return the contents of the Load Balancer status object. This status object corresponds to the `LoadBalancerStatus` enumeration value.

`var://service/system/status/LoadBalancerStatus`

Read-write variables

`var://service/soap-fault-response`
Set when the response input rule is treated as a SOAP fault.

Multi-Protocol Gateway and Web Service Proxy service variables

This section contains information about general variables in alphabetic order by permission category. General variables are available to all services. Table 19 lists the names and permission for these variables.

Table 19. Names and permissions for variables that are available only to Multi-Protocol Gateway services and to Web Service services

Variable name	Permission
`var://service/mpgw/backend-timeout`	Read-write
`var://service/mpgw/request-size`	Read-only
`var://service/mpgw/response-size`	Read-only
`var://service/mpgw/skip-backside`	Write-only

Read-only variables

`var://service/mpgw/request-size`
For Multi-Protocol Gateway and Web Service Proxy services only, gets the size of a request message. The value 0 indicates that the size cannot be determined, perhaps temporarily, due to message streaming or some other processing issue.

`var://service/mpgw/response-size`
For Multi-Protocol Gateway and Web Service Proxy services only, gets the size of a response message. The value 0 indicates that the size cannot be determined, perhaps temporarily, due to message streaming or some other processing issue.

Write-only variables

`var://service/mpgw/skip-backside`
For Multi-Protocol Gateway and Web Service Proxy services only, indicates that the service skips backside processing.

Set this variable to 1 to prevent backside processing. Use this variable as a custom redirect implementation, not as the point of the service. Because the service is not aware of the processing flow, unusual messages might be written to the event log.

Read-write variables

`var://service/mpgw/backend-timeout`
For Multi-Protocol Gateway and Web Service Proxy services only, gets or sets the backend timeout, in seconds. Setting this variable overrides the default timeout. Use an integer in the range of 1 through 86400.

Configuration services service variables

This section contains information about configuration services variables in alphabetic order by permission category. Table 20 lists the names and permission for these variables.

Table 20. Names and permissions for variables that are available for configuration services

Variable name	Permission
var://service/back-attachment-format	Read-only
var://service/config-param	Write-only
var://service/default-stylesheet	Read-only
var://service/domain-name	Read-only
var://service/front-attachment-format	Read-only
var://service/system/frontwsdl	Read-only
var://service/max-call-depth	Read-write
var://service/processor-name	Read-only
var://service/processor-type	Read-only
var://service/xmlmgr-name	Read-only

Read-only variables

`var://service/back-attachment-format`
: Gets the format for the backside attachment.

`var://service/default-stylesheet`
: Gets the name of the default processing policy.

`var://service/domain-name`
: Gets domain of the service.

`var://service/front-attachment-format`
: Gets the format for the frontend attachment.

`var://service/system/frontwsdl`
: Gets the frontend WSDL URL of the service.

`var://service/processor-name`
: Gets the name of the service (processor).

`var://service/processor-type`
: Gets the service (processor) type.

`var://service/xmlmgr-name`
: Gets the name of the XML manager service.

Write-only variables

`var://service/config-param/`*`parameterName value`*
: Sets the specified stylesheet parameter to the specified value.

Read-write variables

`var://service/max-call-depth`
: Gets or sets the maximum call depth for each transaction. This variable controls how many levels of called rules can be layered before an error is thrown. The default is 128.

Load balancer service variables

This section contains information about load balancer variables in alphabetic order by permission category. Table 21 lists the names and permission for these variables.

Table 21. Names and permissions for variables that are available for load balancers

Variable name	Permission
var://service/lb/group	Read-only
var://service/lb/member	Read-only
var://service/lbhealth/	Write-only

Read-only variables

var://service/lb/group
: Gets the name of the load balancer group.

var://service/lb/member
: Gets the group member for the current load balancer.

Write-only variables

var://service/lbhealth/
: Sets the member and state of a load balancer group.

MQ-specific service variables

This section contains information about MQ-specific variables in alphabetic order by permission category. MQ-specific variables are available to MQ Host services and MQ Proxy services. Table 22 lists the names and permission for these variables.

Table 22. Names and permissions for variables that are available to MQ objects

Variable name	Permission
var://service/accounting-token	Read-only
var://service/backout-count	Read-only
var://service/correlation-identifier	Read-write
var://service/expiry	Read-write
var://service/format	Read-write
var://service/message-identifier	Read-write
var://service/message-type	Read-write
var://service/mq-ccsi	Write-only
var://service/mq-error-code	Read-only
var://service/mqmd-reply-to-q	Write-only
var://service/mqmd-reply-to-qm	Write-only
var://service/original-length	Read-only
var://service/persistence	Read-write
var://service/priority	Read-write
var://service/put-date	Read-only
var://service/put-time	Read-only
var://service/reply-to-q	Read-write
var://service/reply-to-qm	Read-write
var://service/report	Read-write

Table 22. Names and permissions for variables that are available to MQ objects (continued)

Variable name	Permission
var://service/user-identifier	Read-only

Read-only variables

var://service/accounting-token
: Gets the MQ message descriptor AccountingToken.

var://service/backout-count
: Gets the MQ Backout Count.

var://service/mq-error-code
: Gets the MQ Reason Code for the last MQ API call.

var://service/original-length
: Gets the MQ message descriptor OriginalLength.

var://service/put-date
: Gets the MQ message descriptor PutDate.

var://service/put-time
: Gets the MQ message descriptor PutTime.

var://service/user-identifier
: MQ User Identifier - Gets the MQ message descriptor UserIdentifier.

Write-only variables

var://service/mq-ccsi
: Sets the MQ message descriptor character set for an MQ Host or Proxy service.

var://service/mqmd-reply-to-q
: Sets the output MQ message descriptor.ReplyToQ value for an MQ Host or Proxy service.

var://service/mqmd-reply-to-qm
: Sets the output MQ message descriptor.ReplyToQMgr value for an MQ Host or Proxy service.

Read-write variables

var://service/correlation-identifier
: Read and write the MQ value in the Correlation Identifier header for MQ Host and Proxy services.

var://service/expiry
: Read and write the MQ value in the Expiry header for MQ Host and Proxy services.

var://service/format
: Read and write the MQ value in the Format header for MQ Host and Proxy services.

var://service/message-identifier
: Read and write the MQ value in the Message Identifier header for MQ Host and Proxy services.

var://service/message-type
: Read and write the MQ value in the Message Type header for MQ Host and Proxy services.

`var://service/persistence`
: Read and write the MQ value in the `Persistence` for MQ Host and Proxy services.

`var://service/priority`
: Read and write the MQ value in the `Priority` header for MQ Host and Proxy services.

`var://service/reply-to-q`
: Read and write the MQ value in the `ReplyToQ` (Reply to Queue) header for MQ Host and Proxy services. When read, shows the input message value. When write, changes the dynamic routing.

`var://service/reply-to-qm`
: Read and write the MQ value in the `ReplyToQMgr` (Reply to Queue Manager) header for MQ Host and Proxy services. When read, shows the input message value. When write, changes the dynamic routing.

`var://service/report`
: Read and write the MQ value in the Report header for MQ Host and Proxy services.

Multistep variables

This section contains information about system variables in alphabetic order by permission category. Multistep variables usually impact the behavior of specific actions in the context of a processing rule. Table 23 lists the names and permission for these variables.

Table 23. Names and permissions for variables that are available to all services

Variable name	Permission
`var://multistep/loop-count`	Read-only
`var://multistep/loop-iterator`	Read-only
`var://service/log/soapversion`	Read-write
`var://service/multistep/contexts`	Read-only

Read-only variables

`var://multistep/loop-count`
: Gets the loop iterator for the innermost `for-each` action. If the `for-each` action is set to run with a fixed iteration count, returns the input context of the loop action. If the loop runs over a node set, returns the current element in the node set.

`var://multistep/loop-iterator`
: Gets the current loop count for the innermost `for-each` action. For the first action in the loop, returns 1; for the second action, returns 2, and so forth.

`var://service/multistep/contexts`
: Gets all existing contexts.

Read-write variables

`var://service/log/soapversion`
: Gets or sets the version of SOAP for use by a SOAP log targets. Use a `setvar` action before a `log` action to change the version of SOAP to use when logging this message.

  Supports the following values:

`soap11` Uses SOAP 1.1.

`soap12` (Default) Uses SOAP 1.2.

Transaction variables

The available transaction variables are separated alphabetically into the following categories:

- Asynchronous transactions
- Error handling
- Headers
- Information
- Persistent connections
- Routing
- Statistics
- URL
- Web Services Management (WSM)

Asynchronous transaction variables

This section contains information about asynchronous transaction variables in alphabetic order by permission category. Table 24 lists the names and permission for these variables.

Table 24. Names and permissions for variables that are available for asynchronous transactions

Variable name	Permission
`var://service/soap-oneway-mep`	Read-write
`var://service/transaction-key`	Write-only
`var://service/transaction-name`	Write-only
`var://service/transaction-timeout`	Write-only

Write-only variables

`var://service/transaction-key`
Sets the token for asynchronous transactions.

`var://service/transaction-name`
Sets the name for asynchronous transactions.

`var://service/transaction-timeout`
Sets the timeout for asynchronous transactions.

Read-write variables

`var://service/soap-oneway-mep`
Gets or sets the SOAP one-way Message Exchange Pattern (MEP) notification.

- When `true`, notifies the service layer that this transaction is performing a one-way MEP operation. This setting enables the service layer to optimize resource usage while preventing Web Services Addressing (WSA) from waiting for and faulting on a response that will never arrive.
- When `false`, no notification is sent. When using WSA and one-way MEPs, the service layer will time out waiting for a response.

When a DataPower service is configured for WSA-to-WSA and it receives a WSA annotated message without the `wsa:MessageId`, the DataPower service assumes that this is a one-way MEP and notifies the service layer by setting this value of this variable to `true`.

This variable is not needed for Web Service Proxy services, as one-way MEPs are identified by reviewing the specifics of the port operation.

Error handling transaction variables

This section contains information about error handling variables in alphabetic order by permission category. Table 25 lists the names and permission for these variables.

Table 25. Names and permissions for variables that are available for error handling

Variable name	Permission
`var://service/error-code`	Read-write
`var://service/error-headers`	Read-only
`var://service/error-ignore`	Read-write
`var://service/error-message`	Read-write
`var://service/error-protocol-reason-phrase`	Write-only
`var://service/error-protocol-response`	Write-only
`var://service/error-subcode`	Read-write
`var://service/formatted-error-message`	Read-only
`var://service/strict-error-mode`	Read-write

Read-only variables

`var://service/error-headers`
: Gets the error headers. This variable contains the name of the HTTP header field that contains error information.

`var://service/formatted-error-message`
: Gets the formatted error message. This variable contains the formatted version of the error text in the `var://service/error-message` variable. The formatted error message is the message that is written to the log file.

Write-only variables

`var://service/error-protocol-reason-phrase`
: Sets the protocol-specific reason phrase for an error. This variable overwrites the reason phrase in the response to provide a short description that an be understood by people.

`var://service/error-protocol-response`
: Sets the protocol-specific response for an error. This variable overwrites the protocol-specific response code in an error condition.

Read-write variables

`var://service/error-code`
: Gets or sets the assigned error code from the Result Code table.

`var://service/error-ignore`
: Gets or sets a flag that controls how the Front Side Handler processes error condition. If the value is set and greater than zero, it does not run any

error handling action and produces a regular response. The content of the message is produced by an error rule.

The default value is 0.

Currently, on the TIBCO EMS and WebSphere JMS Front Side Handler use this variable. If any error happens and the variable is set, the Front Side Handler acknowledges a request message and puts the response message in the PUT queue. This response message will be a SOAP-fault or any output that error rule generates.

`var://service/error-message`
Gets or sets the generic error message that is sent to the client. This variable contains the error condition that stopped multistep processing. Setting this variable overwrites the error response that is sent to the client in an error condition. To set the error message that is written to the log file, use the `var://service/formatted-error-message` variable.

`var://service/error-subcode`
Gets or sets the error sub-code. This variable can help to disambiguate the reason for which the error rule was invoked. Often, the sub-code is the same as the value of the `var://service/error-code` variable. Sometimes, the sub-code is a more specific result code.

`var://service/strict-error-mode`
Gets or sets the strict error mode. This variable controls the error mode for multistep processing.

- If the value is set, an invocation of the `dp:reject` extension element stops multistep processing.
- If the value is not set, an invocation of the `dp:reject` extension element logs a message but does not stop multistep processing.

Headers transaction variables

This section contains information about header variables in alphabetic order by permission category. Table 26 lists the names and permission for these variables.

Table 26. Names and permissions for variables that are available for headers

Variable name	Permission
`var://service/append-request-header/`	Write-only
`var://service/append-response-header/`	Write-only
`var://service/header-manifest`	Read-only
`var://service/set-request-header/`	Write-only
`var://service/set-response-header/`	Write-only

Read-only variables

`var://service/header-manifest`
Gets the transaction header manifest. The manifest lists all protocol headers of current transaction.

Write-only variables

`var://service/append-request-header/`
Appends to the protocol request header.

`var://service/append-response-header/`
Appends to the protocol response header.

`var://service/set-request-header/`
: Sets the protocol request header. This variable directly correlates to the **dp:set-request-header()** extension function. Setting the `var://service/set-request-header/FOO` variable to the value `BAR` would set the request header `FOO` to `BAR`.

`var://service/set-response-header/`
: Sets the protocol response header. This variable directly correlates to the **dp:set-response-header()** extension function. Setting the `var://service/set-response-header/FOO` variable to the value `BAR` would set the response header `FOO` to `BAR`.

Information transaction variables

This section contains information about information variables in alphabetic order by permission category. Table 27 lists the names and permission for these variables.

Table 27. Names and permissions for variables that are available for information

Variable name	Permission
`var://service/current-call-depth`	Read-only
`var://service/input-size`	Read-only
`var://service/transaction-audit-trail`	Read-only
`var://service/transaction-client`	Read-only
`var://service/transaction-id`	Read-only
`var://service/transaction-policy-name`	Read-only
`var://service/transaction-rule-name`	Read-only
`var://service/transaction-rule-type`	Read-only

Read-only variables

`var://service/current-call-depth`
: Gets the current call depth. This variable returns the current depth of called rules. The maximum call depth is set with the `var://service/max-call-depth` variable.

`var://service/input-size`
: Gets the size of the parsed input message (request or response). The value 0 indicates that the size cannot be determined, perhaps temporarily, due to message streaming or some other processing issue.

`var://service/transaction-client`
: Gets the IP Address of transaction client.

`var://service/transaction-id`
: Gets the identifier of transaction.

`var://service/transaction-audit-trail`
: Gets the transaction audit trail.

`var://service/transaction-policy-name`
: Gets the policy name of the transaction

`var://service/transaction-rule-name`
: Gets the rule name of the transaction.

`var://service/transaction-rule-type`
: Gets the rule type of the transaction.

Persistent connection transaction variables

This section contains information about persistent connection variables in alphabetic order by permission category. Table 28 lists the names and permission for these variables.

Table 28. Names and permissions for variables that are available for persistent connections

Variable name	Permission
var://service/connection/note	Read-write
var://service/persistent-connection-counter	Read-only

Read-only variables

var://service/persistent-connection-counter
: Gets the persistent connection counter. This variable returns the number of transactions that were completed on the current protocol session.

Read-write variables

var://service/connection/note
: Gets or sets the annotation for the current connection. This variable allows the user to annotate the current protocol session. The value could be an identifier that could be used to maintain the state based on an existing protocol session.

Routing transaction variables

This section contains information about routing variables in alphabetic order by permission category. Table 29 lists the names and permission for these variables.

Table 29. Names and permissions for variables that are available for routing

Variable name	Permission
var://service/routing-url	Write-only
var://service/routing-url-sslprofile	Write-only

Write-only variables

var://service/routing-url
: For XML Firewall, Multi-Protocol Gateway, and Web Service Proxy services, sets the routing URL. This variable can be set one time only and takes the following format:

```
<dp:set-variable name="var://service/routing-url"
  value="'protocol://target/URI'" />
```

- For XML Firewall services:
  - The protocol must be HTTP or HTTPS. If any other protocol, the service generates an error.
  - The URI is stripped. To specify the URI, use the var://service/URI variable, as shown in the following excerpt:

```
<dp:set-variable name="'var://service/routing-url'"
  value="'http://10.10.36.11:2064'" />
<dp:set-variable name="'var://service/URI'"
  value="'/services'" />
```

- For Multi-Protocol Gateway and Web Service Proxy services:
  - The protocol can be any valid backend protocol.

– The URI is absolute and cannot be controlled with the **Propagate URI** toggle (WebGUI) or **propagate-uri** command.

The `var://service/routing-url` variable is an addition to the `dp:set-target` and `dp:xset-target` extension elements. These extension elements do not allow the specification of a protocol. These extension element, if provided, overrides the value of the target server that is specified in this variable.

`var://service/routing-url-sslprofile`
Sets the SSL proxy profile for the routing URL (dynamic route). Use this variable when the `ssl` property for the DataPower service is not sufficient for the route to be selected. Use this variable before using the `var://service/routing-url` variable.

Statistics variables

This section contains information about statistics variables in alphabetic order by permission category. Table 30 lists the names and permission for these variables.

Table 30. Names and permissions for variables that are available for statistics

Variable name	Permission
`var://service/time-elapsed`	Read-only
`var://service/time-forwarded`	Read-only
`var://service/time-response-complete`	Read-only
`var://service/time-started`	Read-only

Read-only variables

`var://service/time-elapsed`
Gets the duration of the transaction.

`var://service/time-forwarded`
Gets the timestamp for when the request messaged was forwarded.

`var://service/time-response-complete`
Gets the timestamp for when the transaction completes (ends).

`var://service/time-started`
Gets the timestamp for when the request was received (started).

URL-based transaction variables

This section contains information about URL-based transaction variables in alphabetic order by permission category. Table 31 lists the names and permission for these variables.

Table 31. Names and permissions for variables that are available for URL-based transactions

Variable name	Permission
`var://service/client-service-address`	Read-only
`var://service/local-service-address`	Read-only
`var://service/protocol`	Read-only
`var://service/URI`	Read-write
`var://service/URL-in`	Read-only
`var://service/URL-out`	Read-only

Read-only variables

`var://service/client-service-address`
Gets the address of the frontend client.

`var://service/local-service-address`
Gets the address of the frontend service.

`var://service/protocol`
Gets the frontend Protocol

`var://service/URL-in`
Gets the URL of the incoming request.

`var://service/URL-out`

Gets the outbound URL to the backend

Read-write variables

`var://service/URI`
Gets or sets the request URI of the transaction.

Web Services Management transaction variables

This section contains information about Web Services Management (WSM) variables in alphabetic order by permission category. Table 32 lists the names and permission for these variables.

Table 32. Names and permissions for variables that are available to WSM

Variable name	Permission
var://service/wsa/timeout	Read-write
var://service/wsa/genpattern	Read-write
var://service/wsm/aaa-policy-name	Read-only
var://service/wsm/binding	Read-only
var://service/wsm/enabled	Read-only
var://service/wsm/validate-faults	Read-only
var://service/wsm/validate-headers	Read-only
var://service/wsm/validate-message	Read-only
var://service/wsm/wsdl	Read-only
var://service/wsm/wsdl-error	Write-only
var://service/wsm/wsdl-warning	Write-only
var://wsm/num-subschema	Read-only
var://wsm/operation	Read-only
var://wsm/schemalocation	Read-only
var://wsm/resolve-hrefs	Read-only
var://wsm/service	Read-only
var://wsm/service-port	Read-only
var://wsm/service-port-operation	Read-only
var://wsm/strict-fault-document-style	Read-only

Read-only variables

`var://service/wsm/aaa-policy-name`
Gets the name of the WSM AAA policy.

`var://service/wsm/binding`
Gets the WSM service binding.

`var://service/wsm/enabled`
Gets the WSM enabled flag.

`var://service/wsm/validate-faults`
Gets the WSM fault validation.

`var://service/wsm/validate-headers`
Gets the WSM header validation.

`var://service/wsm/validate-message`
Gets the WSM validate message.

`var://service/wsm/wsdl`
Gets the WSM WSDL.

`var://wsm/num-subschema`
Gets the number of WSM subschema.

`var://wsm/operation`
Gets the WSM service operation.

`var://wsm/schemalocation`
Gets the WSM schema location.

`var://wsm/resolve-hrefs`
Gets the WSM resolve HREFs.

`var://wsm/service`
Gets the WSM service name.

`var://wsm/service-port`
Gets the WSM service port.

`var://wsm/service-port-operation`
Gets the WSM service port operation.

`var://wsm/strict-fault-document-style`
WSM strict fault document style. Do not expect RPC wrappers on RPC faults.

Write-only variables

`var://service/wsm/wsdl-error`
Sets the WSDL error.

`var://service/wsm/wsdl-warning`
Sets the WSDL warning.

Read-write variables

`var://service/wsa/timeout`
Gets or sets the timeout value for the WS-Addressing asynchronous reply.

`var://service/wsa/genpattern`
Gets or sets the pattern for the WS-Addressing asynchronous reply.

Extension variables

This section contains information about system variables in alphabetic order by permission category. Extension variables usually impact the behavior of specific actions, particularly `fetch`, `results`, and `results-async` actions. Table 33 lists the names and permission for these variables.

Table 33. Names and permissions for extension variables

Variable name	Permission
`var://local/_extension/allow-compression`	Write-only
`var://local/_extension/attachment-format`	Read-only
`var://local/_extension/attachment-manifest`	Read-only
`var://local/_extension/attachment-root-uri`	Read-only
`var://local/_extension/donot-follow-redirect`	Write-only
`var://local/_extension/error`	Read-only
`var://local/_extension/header/`	Write-only
`var://local/_extension/http-10-only`	Write-only
`var://local/_extension/messages`	Read-only
`var://local/_extension/prevent-persistent-connection`	Write-only
`var://local/_extension/response-headers`	Read-only
`var://local/_extension/response-header/`*`headerName`*	Read-only
`var://local/_extension/responsecode`	Read-only
`var://local/_extension/sslprofile`	Write only
`var://local/_extension/variables`	Read-only

Read-only variables

`var://local/_extension/attachment-format`
: Gets the output format of the attachment.

`var://local/_extension/attachment-manifest`
: Gets the manifest for the attachment.

`var://local/_extension/attachment-root-uri`
: Get the base URI for document attachments.

`var://local/_extension/error`
: Gets the error manifest. This variable contains the error message, if any, from the last **dp:transform()**, **dp:parse()**, or **document()** invocation. If the variable is empty, no error occurred. If an error occurs in any subsequent calls to one of these functions, the existing error message, if any, will be overwritten.

`var://local/_extension/messages`
: Gets the `xsl:message` manifest.

`var://local/_extension/response-headers`
: Gets the manifest for the response header. This variable, on the output context of a **dp:url-open()** extension function or results action or fetch action, contains in the response header manifest.

`var://local/_extension/response-header/`*`headerName`*
: Gets the contents of the specified response header. This variable, in the

output context of a **dp:url-open()** extension function or results action or fetch action, contains the contents of the specified response header.

`var://local/_extension/responsecode`
Gets the response code. This variable is set on an output context to indicate the protocol-level response code of a **dp:url-open()** extension function or results action or fetch action. For instance, if the following action is successful:

```
results tmpvar2 http://foo.bar.com/foome.asp tmpvar3
```

The value of 200 would be written to the `var://context/tmpvar3/_extension/responsecode` context variable.

`var://local/_extension/variables`
Gets the variable manifest.

Write-only variables

`var://local/_extension/allow-compression`
Enables compression of HTTP requests. Set this variable to allow compression of outgoing results content and negotiate the returned document to be compressed if the underlying protocol supports it. For HTTP, this means the `content-encoding` and `accept-encoding` headers.

`var://local/_extension/donot-follow-redirect`
Disables HTTP redirects. Set this variable to prevent the following of protocol-level redirect sequences on the outgoing results and fetch calls that are associated with this context. By default, redirects are followed.

`var://local/_extension/header/`
Appends the specified header field to the protocol connection. Variables of the following form can be set to append headers to the **dp:url-open()** extension function or results action or fetch action connection when a context that contains them is used as the input context:

```
_extension/header/*
```

The following example would add the HTTP header `X-foo: bar` to the HTTP request:

```
setvar tmpvar2 var://local/_extension/header/X-foo bar
results tmpvar2 http://foo.bar.com/foome.asp tmpvar3"
```

`var://local/_extension/http-10-only`
Restricts HTTP to version 1.0. Set this variable to prevent the use of HTTP/1.1 on the related context of a results action or fetch action.

`var://local/_extension/prevent-persistent-connection`
Disables HTTP persistent connection. Set this variable to prevent persistent connections of the outgoing a results action call or fetch action call that is associated with this context. Persistent connections are supported by default, where appropriate.

`var://local/_extension/sslprofile`
Sets the SSL proxy profile for the request. This variable can be set on the input context to a **dp:url-open()** extension function or to a results action or to a fetch action to override the selection of an SSL Proxy Profile. For instance:

```
results tmpvar2 https://foo.bar.com/foome.asp tmpvar3
```

would normally use the SSL Proxy Profile that is associated with any user-agent configuration for the URL

```
https://foo.bar.com/foome.asp
```

If the profile needed to be determined programmatically, perhaps based on AAA, it could be set up as follows to dynamically resolve the value of `*sslprofiletouse`:

```
setvar tmpvar2 var://local/_extension/sslprofile
   var://context/notepad/sslprofiletouse
results tmpvar2 https://foo.bar.com/foome.asp tmpvar3
```

`var://local/_extension/timeout`
: Sets the request timeout on an input context to override any previously set timeout parameter. Set the value in seconds.

System variables

This section contains information about system variables in alphabetic order by permission category. Table 34 lists the names and permission for these variables.

Table 34. Names and permissions for system variables

Variable name	Permission
`var://system/map/debug`	Read-write
`var://system/tasktemplates/debug`	Read-write

Read-write variables

`var://system/map/debug`
: Gets or sets the debugging level for role-based management (RBM).

`var://system/tasktemplates/debug`
: Gets or sets the debugging level for task templates.

List of available variables

Table 35 lists all of the variables that are available when using a DataPower appliance.

Table 35. All available variables

Short variable name	Full variable name	Category	Permission
aaa-policy-name	var://service/wsm/aaa-policy-name	Transaction, WSM	Read-only
accounting-token	var://service/accounting-token	Service, MQ	Read-only
allow-compression	var://local/_extension/allow-compression	Extension	Write-only
append-request-header	var://service/append-request-header	Transaction, headers	Write-only
append-response-header	var://service/append-response-header	Transaction, headers	Write-only
attachment-format	var://local/_extension/attachment-format	Extension	Read-only
attachment-manifest	var://local/_extension/attachment-manifest	Extension	Read-only
attachment-root-uri	var://local/_extension/attachment-root-uri	Extension	Read-only
back-attachment-format	var://service/back-attachment-format	Service, configuration	Read-only
backend-timeout	var://service/mpgw/backend-timeout	Service, general	Read-write
backout-count	var://service/backout-count	Service, MQ	Read-only
binding	var://service/wsm/binding	Transaction, WSM	Read-only
client-service-address	var://service/client-service-address	Transaction, URL	Read-only
config-param	var://service/config-param	Service, configuration	Write-only
contexts	var://service/multistep/contexts	Service, multistep	Read-only
correlation-identifier	var://service/correlation-identifier	Service, MQ	Read-write
current-call-depth	var://service/current-call-depth	Transaction, information	Read-only
debug	var://system/map/debug	System	Read-write
	var://system/tasktemplates/debug		
default-stylesheet	var://service/default-stylesheet	Service, configuration	Read-only
domain-name	var://service/domain-name	Service, configuration	Read-only
donot-follow-redirect	var://local/_extension/donot-follow-redirect	Extension	Write-only
enabled	var://service/wsm/enabled	Transaction, WSM	Read-only
error	var://local/_extension/error	Extension	Read-only
error-code	var://service/error-code	Transaction, error handling	Read-write
error-headers	var://service/error-headers	Transaction, error handling	Read-only

Table 35. All available variables (continued)

Short variable name	Full variable name	Category	Permission
error-ignore	var://service/error-ignore	Transaction, error handling	Read-write
error-message	var://service/error-message	Transaction, error handling	Read-write
error-protocol-reason-phrase	var://service/error-protocol-reason-phrase	Transaction, error handling	Write-only
error-protocol-response	var://service/error-protocol-response	Transaction, error handling	Write-only
error-subcode	var://service/error-subcode	Transaction, error handling	Read-write
expiry	var://service/expiry	Service, MQ	Read-write
format	var://service/format	Service, MQ	Read-write
formatted-error-message	var://service/formatted-error-message	Transaction, error handling	Read-only
front-attachment-format	var://service/front-attachment-format	Service, configuration	Read-only
frontwsdl	var://service/system/frontwsdl	Service, configuration	Read-only
genpattern	var://service/wsa/genpattern	Transaction, WSM	Read-write
group	var://service/lb/group	Service, load balancer	Read-only
header	var://local/_extension/header	Extension	Write-only
header-manifest	var://service/header-manifest	Transaction, headers	Read-only
http-10-only	var://local/_extension/http-10-only	Extension	Write-only
ident	var://service/system/ident	Service, general	Read-only
input-size	var://service/input-size	Transaction, information	Read-only
lbhealth	var://service/lbhealth	Service, load balancer	Write-only
local-service-address	var://service/local-service-address	Transaction, URL	Read-only
loop-count	var://multistep/loop-count	Service, multistep	Read-only
loop-iterator	var://multistep/loop-iterator	Service, multistep	Read-only
max-call-depth	var://service/max-call-depth	Service, configuration	Read-write
member	var://service/lb/member	Service, load balancer	Read-only
message-identifier	var://service/message-identifier	Service, MQ	Read-write
message-type	var://service/message-type	Service, MQ	Read-write
messages	var://local/_extension/messages	Extension	Read-only
mq-ccsi	var://service/mq-ccsi	Service, MQ	Write-only

Table 35. All available variables (continued)

Short variable name	Full variable name	Category	Permission
mq-error-code	var://service/mq-error-code	Service, MQ	Read-only
mqmd-reply-to-q	var://service/mqmd-reply-to-q	Service, MQ	Write-only
mqmd-reply-to-qm	var://service/mqmd-reply-to-qm	Service, MQ	Write-only
note	var://service/connection/note	Transaction, persistent connection	Read-write
num-subschema	var://wsm/num-subschema	Transaction, WSM	Read-only
operation	var://wsm/operation	Transaction, WSM	Read-only
original-length	var://service/original-length	Service, MQ	Read-only
persistence	var://service/persistence	Service, MQ	Read-write
persistent-connection-counter	var://service/persistent-connection-counter	Transaction, persistent connection	Read-only
prevent-persistent-connection	var://local/_extension/prevent-persistent-connection	Extension	Write-only
priority	var://service/priority	Service, MQ	Read-write
processor-name	var://service/processor-name	Service, configuration	Read-only
processor-type	var://service/processor-type	Service, configuration	Read-only
protocol	var://service/protocol	Transaction, URL	Read-only
put-date	var://service/put-date	Service, MQ	Read-only
put-time	var://service/put-time	Service, MQ	Read-only
request-size	var://service/mpgw/request-size	Service, general	Read-only
reply-to-q	var://service/reply-to-q	Service, MQ	Read-write
reply-to-qm	var://service/reply-to-qm	Service, MQ	Read-write
report	var://service/report	Service, MQ	Read-write
resolve-hrefs	var://wsm/resolve-hrefs	Transaction, WSM	Read-only
response-header	var://local/_extension/response-header	Extension	Read-only
response-headers	var://local/_extension/response-headers	Extension	Read-only
response-size	var://service/mpgw/response-size	Service, general	Read-only
responsecode	var://local/_extension/responsecode	Extension	Read-only
routing-url	var://service/routing-url	Transaction, routing	Write-only
routing-url-sslprofile	var://service/routing-url-sslprofile	Transaction, routing	Write-only
schemalocation	var://wsm/schemalocation	Transaction, WSM	Read-only
service	var://wsm/service	Transaction, WSM	Read-only

Table 35. All available variables (continued)

Short variable name	Full variable name	Category	Permission
service-port	var://wsm/service-port	Transaction, WSM	Read-only
service-port-operation	var://wsm/service-port-operation	Transaction, WSM	Read-only
set-request-header	var://service/set-request-header	Transaction, headers	Write-only
set-response-header	var://service/set-response-header	Transaction, headers	Write-only
skip-backside	var://service/mpgw/skip-backside	Service, general	Write-only
soap-fault-response	var://service/soap-fault-response	Service, general	Read-write
soap-oneway-mep	var://service/soap-oneway-mep	Transaction, asynchronous	Read-write
soapversion	var://service/log/soapversion	Service, multistep	Read-write
sslprofile	var://local/_extension/sslprofile	Extension	Write-only
status/	var://service/system/status/*status-enumeration*	Service, general	Read-only
strict-error-mode	var://service/strict-error-mode	Transaction, error handling	Read-write
strict-fault-document-style	var://wsm/strict-fault-document-style	Transaction, WSM	Read-only
time-elapsed	var://service/time-elapsed	Transaction, statistics	Read-only
time-forwarded	var://service/time-forwarded	Transaction, statistics	Read-only
time-response-complete	var://service/time-response-complete	Transaction, statistics	Read-only
time-started	var://service/time-started	Transaction, statistics	Read-only
timeout	var://service/wsa/timeout	Transaction, WSM	Read-write
transaction-audit-trail	var://service/transaction-audit-trail	Transaction, information	Read-only
transaction-client	var://service/transaction-client	Transaction, information	Read-only
transaction-id	var://service/transaction-id	Transaction, information	Read-only
transaction-key	var://service/transaction-key	Transaction, asynchronous	Write-only
transaction-name	var://service/transaction-name	Transaction, asynchronous	Write-only
transaction-policy-name	var://service/transaction-policy-name	Transaction, information	Read-only
transaction-rule-name	var://service/transaction-rule-name	Transaction, information	Read-only
transaction-rule-type	var://service/transaction-rule-type	Transaction, information	Read-only

Table 35. All available variables (continued)

Short variable name	Full variable name	Category	Permission
transaction-timeout	var://service/transaction-timeout	Transaction, asynchronous	Write-only
URI	var://service/URI	Transaction, URL	Read-write
URL-in	var://service/URL-in	Transaction, URL	Read-only
URL-out	var://service/URL-out	Transaction, URL	Read-only
user-identifier	var://service/user-identifier	Service, MQ	Read-only
validate-faults	var://service/wsm/validate-faults	Transaction, WSM	Read-only
validate-headers	var://service/wsm/validate-headers	Transaction, WSM	Read-only
validate-message	var://service/wsm/validate-message	Transaction, WSM	Read-only
variables	var://local/_extension/variables	Extension	Read-only
wsdl	var://service/wsm/wsdl	Transaction, WSM	Read-only
wsdl-error	var://service/wsm/wsdl-error	Transaction, WSM	Write-only
wsdl-warning	var://service/wsm/wsdl-warning	Transaction, WSM	Write-only
xmlmgr-name	var://service/xmlmgr-name	Service, configuration	Read-only

Appendix B. Processing Policy procedures

Stylesheet policies can be created using two slightly different methods. The first method (referred to as the *inline rule* method) initially creates the processing policy and then defines transformation and filtering rules specific to that policy. The second method (referred to as the *global rule* method) initially defines transformation and filtering rules that are available to all stylesheet policies. These global rules are later associated with a Processing Policy during its creation.

Stylesheet policies using inline rules

Use the following procedure to implement a Processing Policy using inline rules.

1. Use the **matching** command to enter Matching Rule configuration mode and to create a named matching rule or rules.
2. Use the **urlmatch** or **httpmatch** command to populate matching rules with shell-style match patterns. The URL or HTTP match patterns specify the conditions under which policy-based XSL filtering or transformation will be performed.

   You can use wildcards to define a match pattern as follows:

   ***** The string wildcard matches 0 or more occurrences of any character.

   ? The single character wildcard matches one occurrence of any single character.

   [] The delimiters bracket a character or numeric range:

   `[1-5]` Matches 1, 2, 3, 4, or 5

   `[xy]` Matches x or y
3. Use the Global **stylepolicy** command to create a Processing Policy and enter Processing Policy configuration mode. You can also use the **stylepolicy** command to specify a default style sheets used for SOAP filtering and XSL transformations of candidate documents that fail to match Processing Policy rules.
4. For XML Firewall services only, use the **request-rule**, **response-rule**, or **rule** commands in conjunction with the `filter` and `validate` actions to add direction-specific filters to the Processing Policy. Filters (which result in an accept or reject decision) enable validation of an XML document against a specified schema, verification of a document's digital signature, or content-based XML/SOAP filtering.

   Each filtering rule references a named matching rule (created in Steps 1 and 2). If the candidate URL or HTTP header matches a pattern in the matching rule, XSL filtering is immediately performed as defined by the rule.
5. For both XML Firewall and XSL Proxy services, use the **request-rule**, **response-rule**, or **rule** commands in conjunction with the `xform` and `xformpi` actions to add direction-specific transformation rules to a Processing Policy.

   Each transformation rule references a named matching rule (created in Steps 1 and 2), and defines a transform procedure. If the candidate URL or HTTP header matches a pattern in the matching rule, XSL transformation is immediately performed as defined in the rule.
6. Use the **stylesheet-policy** command (XML Firewall and XSL Proxy) to associate the Processing Policy with either an XML firewall or an XSL proxy.

Configuring a Matching Rule

This command sequence creates a matching rule, named `star`, that provides a universal match for all URLs.

```
# matching star
Matching configuration mode
# urlmatch *
(config-stylesheet-matching)# exit
Matching 'star' successfully created
#
```

Configuring a Processing Policy

This command sequence creates the `validate-sign-encrypt-all` Processing Policy that:

1. Validates all client and server generated documents against a named schema
2. Signs all validated documents
3. Encrypts all signed documents
4. Forwards encrypted documents to the destination client or server.

```
# stylepolicy validate-sign-encrypt-all
Processing Policy configuration mode
# rule star
Processing Policy Rule configuration mode
# validate INPUT schema store:///soap-envelope-1.1.xsd
# xform INPUT store:///sign-wssec.xsl tmp1
# xform tmp1 store:///encrypt-soap.xsl OUTPUT
# exit
Stylesheet Rule "star" successfully created
# exit
Processing Policy "validate-sign-encrypt-all" successfully created
#
```

This command sequence creates the `multi-step` Processing Policy that:

1. Transforms all client requests using a specified style sheet
2. Transforms the results of the initial transforms using a second style sheet
3. Performs a final transformation using another style sheet
4. Forwards the final transformation to the target server

```
# stylepolicy multi-step
Processing Policy configuration mode
# request-rule star
Stylesheet Rule configuration mode
# xform INPUT http://10.1.1.8/XSL/XForm_1.xsl tmp1
# xform tmp1 http://10.1.1.8/XSL/XForm_2.xsl tmp2
# xform tmp2 http://10.1.1.8/XSL/XForm_3.xsl OUTPUT
# exit
Stylesheet Rule "star" successfully created
# exit
Processing Policy "multi-step" successfully created
#
```

Assigning a Processing Policy to a DataPower service

This command sequence creates the `validate-sign-encrypt-all` XML Firewall. The sequence assigns the `validate-sign-encrypt-all` Processing Policy to the XML Firewall of the same name.

```
# xmlfirewall validate-sign-encrypt-all
XML firewall configuration mode
# local-address 0 9050
# remote-address 10.10.0.1 9000
# xml-manager mgr1
```

```
# stylesheet-policy validate-sign-encrypt-all
# parameter keypair ALICE
# parameter recipient Alice
# request-type xml
# response-type unprocessed
# exit
XML Firewall update successful
#
# xmlfirewall validate-sign-encrypt-all
XML firewall configuration mode
# local-address 0 9050
# remote-address 10.10.0.1 9000
# xml-manager mgr1
# stylesheet-policy validate-sign-encrypt-all
# parameter keypair ALICE
# parameter recipient Alice
# request-type xml
# response-type unprocessed
# exit
XML Firewall update successful
#
```

This command sequence creates the `multi-step` XSL Proxy. The sequence assigns the `multi-step` Processing Policy to the XSL Proxy of the same name.

```
# xslproxy multi-step
XSL proxy configuration mode
# local-address 0 64000
# remote-address 10.12.12.1 64000
# xml-manager mgr1
# stylesheet-policy multi-step
# request-type xml
# response-type unprocessed
# exit
XSL Proxy update successful
#
```

Stylesheet policies using global rules

Use the following procedure to implement a processing policy using global rules.

1. Use the **matching** command to enter Matching Rule configuration mode and to create a named matching rule or rules.
2. Use the **urlmatch** command or the **httpmatch** command to populate the matching rules with shell-style match patterns. The URL or HTTP match patterns specify the conditions under which policy-based XSL filtering or transformation will be performed.

   You can use wildcards to define a match pattern as follows:

   ***** The string wildcard matches 0 or more occurrences of any character.

   ? The single character wildcard matches one occurrence of any single character.

   [] The delimiters bracket a character or numeric range:

   `[1-5]` Matches 1, 2, 3, 4, or 5

   `[xy]` Matches x or y
3. Use the Global **rule** command to create a named directional rule; this named global rule will be available to all stylesheet policies.
4. For XML Firewall services only, use the **request-rule**, **response-rule**, or **rule** commands in conjunction with the `filter` and `validate` actions to add direction-specific filters to the global rule. Filters (which result in an

accept/reject decision) enable validation of an XML document against a specified schema, verification of a document's digital signature, or content-based XML/SOAP filtering.

5. Use the **request-rule**, **response-rule**, or **rule** commands in conjunction with the `xform` and `xformpi` actions to add direction-specific transformation rules to the global rule.
6. Use the Global **stylepolicy** command to create a Processing Policy.
7. Use the Processing Policy **match** command to associate a global rule with a matching rule (created in Steps 1 and 2), and to assign the associated global rule-matching rule pair to the Processing Policy.
8. Use the **stylesheet-policy** command (XML Firewall and XSL Proxy) to associate the Processing Policy with an XML Firewall or XSL proxy.

Configuring a Matching Rule

This command sequence creates a matching rule, named `star`, that provides a universal match for all URLs.

```
# matching star
Matching configuration mode
# urlmatch *
# exit
Matching "star" successfully created
#
```

Configuring a Global Rule

This command sequence creates the `validate-sign-encrypt` global rule that:

1. Validates client and server generated documents against a named schema
2. Signs validated documents
3. Encrypts signed documents
4. Forwards encrypted documents to the destination client or server.

```
# rule validate-sign-encrypt-all
Processing Policy Rule configuration mode
# validate INPUT schema store:///soap-envelope-1.1.xsd
# xform INPUT store:///sign-wssec.xsl tmp1
# xform tmp1 store:///encrypt-soap.xsl OUTPUT
# exit
#
```

This command sequence creates the `multi-step` global rule that:

1. Transforms client requests using a specified style sheet
2. Transforms the results of the initial transforms using a second style sheet
3. Performs a final transformation using another style sheet
4. Forwards the final transformation to the target server

```
# rule multi-step request
Stylesheet Rule configuration mode
# xform INPUT http://10.1.1.8/XSL/XForm_1.xsl tmp1
# xform tmp1 http://10.1.1.8/XSL/XForm_2.xsl tmp2
# xform tmp2 http://10.1.1.8/XSL/XForm_2.xsl OUTPUT
# exit
#
```

Configuring a Processing Policy

This command sequence creates the `validate-sign-encrypt-all` Processing Policy that uses the `validate-sign-encrypt` global rule with the `star` matching pattern.

```
# stylepolicy validate-sign-encrypt-all
Processing Policy configuration mode
# match star validate-sign-encrypt
# exit
Processing Policy "validate-sign-encrypt-all" successfully created
#
```

This command sequence creates the `multi-step-all` Processing Policy that uses the `multi-step` global rule with the `star` matching pattern.

```
# stylepolicy multi-step-all
Processing Policy configuration mode
# match star multi-step
# exit
Processing Policy "multi-step" successfully created
#
```

Assigning a Processing Policy to a DataPower service

This command sequence creates the `validate-sign-encrypt-all` XML Firewall. The sequence assigns the `validate-sign-encrypt-all` Processing Policy to the XML Firewall of the same name.

```
# xmlfirewall validate-sign-encrypt-all
XML firewall configuration mode
# local-address 0 9050
# remote-address 10.10.0.1 9000
# xml-manager mgr1
# stylesheet-policy
validate-sign-encrypt-all
# parameter keypair ALICE
# parameter recipient Alice
# request-type xml
# response-type unprocessed
# exit
#
```

This command sequence creates the `multi-step` XSL Proxy. The sequence assigns the `multi-step` Processing Policy to the XSL Proxy of the same name.

```
# xslproxy multi-step
XSL proxy configuration mode
# local-address 0 64000
# remote-address 10.12.12.1 64000
# xml-manager mgr1
# stylesheet-policy multi-step
# request-type xml
# response-type unprocessed
# exit
XSL Proxy update successful
#
```

Appendix C. Stylesheet Refresh Policy configuration

Disabling the cache can be performance concern and might not be your goal. When style sheets are not cached, an XSLT compilation is run on every single transaction.

If you need to disable stylesheet caching, create a separate XML Manager for the particular service where stylesheet caching is not required.

Do not disable stylesheet caching for a widely used XML Manager. For example, do not disable caching on the default XML Manager.

High-level procedure

Use the following procedure to implement a Stylesheet Refresh Policy:

1. Use the **urlmap** command to create one or more URL maps.
2. Use the **match** command to add shell-style match patterns to the URL map or maps.

If desired, you can use the **test urlmap** command to test candidate patterns against a specific URL map.

1. Use the **urlrefresh** command to create a Stylesheet Refresh Policy.
2. Use the **disable cache**, **disable flush**, or **interval urlmap** command to populate the Stylesheet Refresh Policy with one or more URL maps, and (when required) to assign a refresh interval to each URL map that was added to the Stylesheet Refresh Policy.

If desired, you can use the **test urlrefresh** command to test a given pattern against a specific Stylesheet Refresh Policy.

1. Use the **xslrefresh** command to assign the Stylesheet Refresh Policy to a specific XML Manager.

Example

The following example shows how to use commands to create the `dataglue_no_refresh` URL refresh policy that disables stylesheet caching for the `contivo-mgr` XML Manager:

```
xmlmgr contivo-mgr

urlrefresh "dataglue_no_refresh"
reset
disable cache "ALL"
exit

xslrefresh contivo-mgr "dataglue_no_refresh"
```

Appendix D. Compile Options Policy configuration

Profiling overview

With profiling enabled, the appliance measures and reports processing times for the profiled style sheets.

The appliance reports time measurements as a percentage of the total time spent processing the document. As the control flow of an XSLT program is not well defined or necessarily straightforward, those totals may not add up to 100% and in some cases the same time may count toward multiple regions of the style sheet.

Measurements fall into the following categories.

Global Variables
The time spent to define each global variable is measured. If the global variable calls a template or performs an *apply-templates*, the time spent executing that template counts toward both the template called and the global variable.

Templates
Time spent in a template does *not* include the time spent in templates called by that template. It is a flat measurement of total time spent in that particular template; one could think of as a stop watch which starts when the template is entered and is stopped when another template is called; when the other template returns, the stopwatch resumes measurement.

Special Entities
Certain specially constructed entities are also measured, such as the time spent gathering key data on the document the first time that **key()** is called.

User-Defined Regions
Users may define custom measurement regions with the extension element:

```
{http://www.datapower.com/extensions}profile
```

or with the extension attribute of the same name.

The `dp:profile` extension element measures the time spent in its contents, and the `dp:profile` attribute measures the time spent in the instruction it is found upon.

The following XSLT example demonstrates both techniques

```
<xsl:stylesheet
   xmlns:dp="http://www.datapower.com/extensions"
   xsl:version="1.0"
   xmlns:xsl="...">

   <xsl:template match="/">

   <!--
     Time spent in statements appearing here will show up in the template profile
     -->

   <xsl:.../>

   <dp:profile name="region-1">
```

```
      <!--
        Time spent in statements appearing here will show up in both the template
        profile and region-1.
        -->

      <xsl:.../>
      <xsl:for-each select="something" dp:profile="region-2">
         <!--
           Time spent in statements appearing here will show up in the template
           profile, region-1, AND region-2
           -->
      <xsl:.../>
   </xsl:for-each>
      </dp:profile>
      <xsl:for-each select="..." dp:profile="region-3">
         <!-- time spent here will show up in the template profile and region-3 -->
          <xsl:.../>
      </xsl:for-each>

      <!-- time spent here will only show up in the template profile -->
      <xsl:.../>
   </xsl:template>
</xsl:stylesheet>
```

Like global variables, if a user region encompasses a call-template, it includes the time spent in all templates called.

Profiling results are available with the **show profile** command, from the WebGUI (**STATUS** → **Stylesheet Profiles**), or through the XML management interface.

Note: Debug mode changes the output of the program. Instead of generating its normal output, the program is modified to output an HTML web page describing which line of the style sheet generated each piece of the output. In addition, the debug page is annotated with notes when templates are entered, with the values that are assigned to variables, and other messages that may assist the user in XSLT development or troubleshooting.

Configuration overview

Use the following procedure to implement a Compile Options Policy:

1. Use the **compile-options** command (Global) to create a named Compile Options Policy.
2. Use the **profile** or **debug** command (Compile Options Policy) to define URL sets to be profiled.

   You assign URL Maps to the Compile Options Policy to define one or more URL sets that are subject to profiling.

   Any style sheet whose URL matches the criteria specified by the **profile** or **debug** command will be profiled.
3. Use the **exit** command to commit the newly-defined Profiling Policy and return to Global configuration mode.
4. Use the **xslconfig** command (Global) to assign the Profiling Policy to an XML Manager, thus enabling profiling by that XML Manager.

Appendix E. Getting help and technical assistance

This section describes the following options for obtaining support for IBM products:

- "Searching knowledge bases"
- "Getting a fix"
- "Contacting IBM Support" on page 1118

Searching knowledge bases

If you encounter a problem, you want it resolved quickly. You can search the available knowledge bases to determine whether the resolution to your problem was already encountered and is already documented.

Documentation

The IBM WebSphere DataPower documentation library provides extensive documentation in Portable Document Format (PDF). You can use the search function of Adobe® Acrobat to query information. If you download and store the documents in a single location, you can use the search facility to find all references across the documentation set.

IBM Support

If you cannot find an answer in the documentation, use the *Search Support* feature from the product-specific support page.

From the **Search Support (this product)** area of the product-specific support page, you can search the following IBM resources:

- IBM technote database
- IBM downloads
- IBM Redbooks®
- IBM developerWorks®

Getting a fix

A product fix might be available to resolve your problem. To determine what fixes are available for your IBM product, check the product support site by performing the following steps:

1. Go to the IBM Support site at the following Web address:

   http://www.ibm.com/support
2. Select **Support & Downloads** → **Download** to open the Support & downloads page.
3. From the **Category** list, select **WebSphere**.
4. From the **Sub-Category** list, select **WebSphere DataPower SOA Appliances**.
5. Click the **GO** icon to display the list of most recent updates.
6. Click the link for the firmware and documentation download that is specific to your WebSphere DataPower product.
7. Follow the instructions in the technote to download the fix.

Contacting IBM Support

IBM Support provides assistance with product defects. Before contacting IBM Support, the following criteria must be met:

- Your company has an active maintenance contract.
- You are authorized to submit problems.

To contact IBM Support with a problem, use the following procedure:

1. Define the problem, gather background information, and determine the severity of the problem. For help, refer to the *Software Support Handbook*. To access the online version of this handbook, use the following procedure:
   a. Access the IBM Software Support Web page at the following Web address:

      http://www.ibm.com/software/support
   b. Scroll down to the **Additional support links** section of the page.
   c. Under **Support tools**, click the **Software Support Handbook** link.
   d. Bookmark this page for future reference.

   From this page, you can obtain a PDF copy of the handbook.
2. Gather diagnostic information.
   a. Access the product support at the following Web address:

      http://www.ibm.com/software/integration/datapower/support
   b. Locate the **Assistance** area of the product support page.
   c. Click **Information to include** to access that technote that lists the information that is required to report a problem.
3. Submit the problem in one of the following ways:

   Online

   From the IBM Support Web site (http://www.ibm.com/support), select **Support & Downloads → Open a service request**. Following the instructions.

   By phone

   For the phone number to call in your country, refer to "Contacts" in the *Software Support Handbook*. From the Software Support Handbook Web site, click **Contacts**. In the U.S. and Canada, call 1–800–IBM-SERV (1–800–426–7378) and select option 2 for software.

If the problem you should submit is for a software defect or for missing or inaccurate documentation, IBM Support creates an authorized program analysis report (APAR). The APAR describes the problem in detail. Whenever possible, IBM Support provides a workaround that you can implement until the APAR is resolved and a fix is delivered.

Notices and trademarks

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information about the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements or changes in the product(s) or the program(s) described in this publication at any time without notice.

Trademarks

IBM, the IBM logo, developerWorks, DB2, DataPower, IMS, RACF, Redbooks, Tivoli, WebSphere, and z/OS are registered trademarks of the International Business Machines Corporation in the United States or other countries.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Index

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Printed in USA