



Service Design



Published: 2013-06-21

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Copyright © 2013, Juniper Networks, Inc. All rights reserved.

Juniper Networks, Junos, Steel-Belted Radius, NetScreen, and ScreenOS are registered trademarks of Juniper Networks, Inc. in the United States and other countries. The Juniper Networks Logo, the Junos logo, and JunosE are trademarks of Juniper Networks, Inc. All other trademarks, service marks, registered trademarks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Products made or sold by Juniper Networks or components thereof might be covered by one or more of the following patents that are owned by or licensed to Juniper Networks: U.S. Patent Nos. 5,473,599, 5,905,725, 5,909,440, 6,192,051, 6,333,650, 6,359,479, 6,406,312, 6,429,706, 6,459,579, 6,493,347, 6,538,518, 6,538,899, 6,552,918, 6,567,902, 6,578,186, and 6,590,785.

Service Design

Copyright © 2013, Juniper Networks, Inc.
All rights reserved.

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at <http://www.juniper.net/support/eula.html>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

	About the Documentation	ix
	Documentation and Release Notes	ix
	Documentation Conventions	ix
	Documentation Feedback	xi
	Requesting Technical Support	xi
	Self-Help Online Tools and Resources	xii
	Opening a Case with JTAC	xii
Part 1	Overview	
Chapter 1	Understanding Service Template Design for Media Flow Controllers with Media Flow Activate	3
	Service Design Overview	4
	Network Optimization Services Overview	7
	HTTP Reverse Proxy Services Overview	8
	Content Ingest Services Overview	8
Part 2	Administration	
Chapter 2	Managing Provisioned Service Templates with Media Flow Activate	11
	Managing Provisioned Services	11
	Purging Content from Media Flow Controller Devices	14
	Actions on Services	15
Part 3	Configuration	
Chapter 3	Creating Service Templates for Media Flow Controllers with Media Flow Activate	19
	Creating Network Optimization Services	19
	Creating Network Optimization Service XML Files for Import	26
	Creating HTTP Reverse Proxy Services	30
	Creating Reverse Proxy Service XML Files for Export	41
	Creating Reverse Proxy Service XML Files for Import	42
	Creating Content Ingest Services	51
Part 4	Index	
	Index	55

List of Figures

Part 3	Configuration
Chapter 3	Creating Service Templates for Media Flow Controllers with Media Flow Activate 19
	Figure 1: Create Network Optimization Service—General Tab 20
	Figure 2: Create HTTP Reverse Proxy Service—General Tab 32

List of Tables

	About the Documentation	ix
	Table 1: Notice Icons	x
	Table 2: Text and Syntax Conventions	x
Part 3	Configuration	
Chapter 3	Creating Service Templates for Media Flow Controllers with Media Flow Activate	19
	Table 3: Example: Domain Regex Values	25
	Table 4: contentDirectDefinition Options and Elements	27
	Table 5: rproxyDefinition Options and Elements	44

About the Documentation

- Documentation and Release Notes on page ix
- Documentation Conventions on page ix
- Documentation Feedback on page xi
- Requesting Technical Support on page xi

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at <http://www.juniper.net/techpubs/>.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at <http://www.juniper.net/books>.

Documentation Conventions

Table 1 on page x defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page x defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the configure command: <code>user@host> configure</code>
Fixed-width text like this	Represents output that appears on the terminal screen.	<code>user@host> show chassis alarms</code> <code>No alarms currently active</code>
<i>Italic text like this</i>	<ul style="list-style-type: none"> Introduces or emphasizes important new terms. Identifies book names. Identifies RFC and Internet draft titles. 	<ul style="list-style-type: none"> A policy <i>term</i> is a named structure that defines match conditions and actions. <i>Junos OS System Basics Configuration Guide</i> RFC 1997, <i>BGP Communities Attribute</i>
<i>Italic text like this</i>	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# set system domain-name <i>domain-name</i>
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none"> To configure a stub area, include the stub statement at the [edit protocols ospf area area-id] hierarchy level. The console port is labeled CONSOLE.
< > (angle brackets)	Enclose optional keywords or variables.	<code>stub <default-metric metric>;</code>

Table 2: Text and Syntax Conventions (*continued*)

Convention	Description	Examples
(pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	broadcast multicast <i>(string1 string2 string3)</i>
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	rsvp { # Required for dynamic MPLS only
[] (square brackets)	Enclose a variable for which you can substitute one or more values.	community name members [community-ids]
Indentation and braces ({ })	Identify a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop address; retain; } } }
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none"> In the Logical Interfaces box, select All Interfaces. To cancel the configuration, click Cancel.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select Protocols>Ospf .

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at <https://www.juniper.net/cgi-bin/docbugreport/>. If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable)

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract,

or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at <http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <http://www.juniper.net/support/warranty/>.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: <http://www.juniper.net/customers/support/>
- Search for known bugs: <http://www2.juniper.net/kb/>
- Find product documentation: <http://www.juniper.net/techpubs/>
- Find solutions and answer questions using our Knowledge Base: <http://kb.juniper.net/>
- Download the latest versions of software and review release notes: <http://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://www.juniper.net/alerts/>
- Join and participate in the Juniper Networks Community Forum: <http://www.juniper.net/company/communities/>
- Open a case online in the CSC Case Management tool: <http://www.juniper.net/cm/>

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: <https://tools.juniper.net/SerialNumberEntitlementSearch/>

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at <http://www.juniper.net/cm/>.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see <http://www.juniper.net/support/requesting-support.html>.

PART 1

Overview

- [Understanding Service Template Design for Media Flow Controllers with Media Flow Activate on page 3](#)

CHAPTER 1

Understanding Service Template Design for Media Flow Controllers with Media Flow Activate

- [Service Design Overview on page 4](#)
- [Network Optimization Services Overview on page 7](#)
- [HTTP Reverse Proxy Services Overview on page 8](#)
- [Content Ingest Services Overview on page 8](#)

Service Design Overview

You use the **Service Design** workspace to create content delivery services. Transparent, reverse proxy, and content ingest content delivery services are supported. You create these services for the types of media you are delivering.

If you are an Internet service provider, create a **Network Optimization Service** to transparently cache a popular website, thereby saving bandwidth and optimizing Internet content delivery.

If you are a content provider or a Content Delivery Network (CDN) service provider who owns or manages any content, create an **HTTP Reverse Proxy Service** to efficiently deliver that content.

If you are a CDN service provider, create a **Content Ingest Service** to ingest popular content before it is needed. Using content ingest service, you can periodically refresh the cached content for sites that are regularly updated.

Before you begin configuring proxy services, it is good to have the following information handy:

- The regular expression (regex) for the **Domain** that will be used in requests for the media for this service. In addition, the file **Path** to those videos in that domain.
- The precedence you want to give this service among all the configured services. The higher the **Precedence** value, the lower the precedence. See “Understanding Precedence” in this topic for details about using the “precedence” parameter.
- For cache-misses (requests for media not in cache), how you want the service to obtain the requested media.
- The **Virtual Player** you want to manage video delivery policies for that site. This is needed when the site being cached serves HTTP video. A single virtual player can be used for multiple sites. See *Creating Virtual Players* to create a virtual player.
- The **Cache Tuning Policy** you want to fine-tune the cache settings on the basis of the network conditions and caching requirements. See *Creating Cache-Tuning Policies* to create a cache-tuning policy.
- If you are creating a reverse proxy service, consider the list of origin servers that Media Flow Controller must access to fetch content upon a cache-miss.

Understanding Precedence

Use **Precedence** to set unambiguous mapping of incoming GET requests in the case of **Domain** overlap; precedence can be set on all domains. The lower the number, the higher the preference for that domain; values 0 (highest precedence) through 10 (lowest precedence) can be used. All services have a default precedence of 0.

Example for network optimization service: Consider three websites and service configurations as follows:

- **Website 1**

http://a.com/abc/def/file1.flv

Name website1

Domain a.com

Path /abc/def

Precedence 1

Origin Server Resolution Follow Host Header

Use Client IP

- Website 2

http://a.com/abc/file2.flv

Name website2

Domain a.com

Path /abc

Precedence 2

Origin Server Resolution Follow Host Header

Use Client IP

- Website 3

http://a.com/pqr/file3.flv

Name website3

Domain a.com

Path /

Precedence 3

Origin Server Resolution Follow Host Header

Use Client IP

All three websites match **Domain a.com** and **Path /** (slash). In order to ensure that media files at **/abc/def** map to **website 1** and not **website 3**, set the **Precedence** value higher for **website 1**. This is the same as the case with mapping media files at **/abc** to **website 2**. Only media files at the default location, **/**, should be mapped to **website 3** with the **Precedence** value configured the lowest, as shown above. Similar logic applies to the reverse proxy service as well.

Example for reverse proxy service:

- Website 1

http://a.com/abc/def/file1.flv

Name website1

Domain a.com

Path /abc/def

Precedence 1

HTTP Origin FQDN Server Name 10.102.0.81 Port 80

- Website 2

http://a.com/abc/def/file2.flv

Name website2

Domain a.com

Path /abc

Precedence 2

HTTP Origin Origin map xyz



BEST PRACTICE: Excessive use of precedence has a performance impact because precedence allows for the longest prefix matching. If possible, configure websites so that there are no overlaps in the Domain and Path combination, that is used for mapping the incoming HTTP GET to the requested media files.

Understanding Dynamic URI Remapping

Some popular content providers generate dynamically created URIs for the same content for various reasons, including security. This causes caches to have low cache-hit ratios even when the content is in the cache. Media Flow Controller can identify these dynamic URIs as pointing to the same content and can ensure delivery of the correct content. Media Flow Controller identifies dynamic URIs via regex substring-addressing of matches that allows access to various portions of the matched string, denoted by parentheses in the regex expression. The complete string match is referred to as \$0, the left-most substring is referred to as \$1, each subsequent substring being \$2, \$3, and so on. You configure a regular expression and mapping string on a per-namespace basis. The mapping string is an ASCII-printable string that describes the mapping from the various substring matches in the regular expression, to a new URI.

The commands for dynamic URI mapping specify that an incoming request having a URI matching the configured regex value (**url to match**) is matched to a cache index string value (**map-to**). The **tunnel unmatched** option specifies that if the match fails, the request is tunneled. With **revalidate cache hit = true** configured, if the origin server returns "Object Is Modified," the transaction is tunneled, and the object is deleted from the cache, provided that its time-to-live (TTL) has expired; and the new object is fetched into the cache.



NOTE: See the *Juniper Networks Media Flow Controller Administrators Guide* for detailed information about namespaces.

- Related Documentation**
- *Media Flow Activate Overview*
 - *Understanding Media Flow Controller Management with Media Flow Activate*
 - [Network Optimization Services Overview on page 7](#)
 - [HTTP Reverse Proxy Services Overview on page 8](#)
 - [Content Ingest Services Overview on page 8](#)
 - *Tagging Media Flow Controller Objects*
 - *Quick Reference to Tasks in Media Flow Activate*

Network Optimization Services Overview

This topic describes what Network Optimization services are used for and what information you need to know before creating a service.

At a minimum, service instance configuration requires a **Domain**, a **Path** to the media files, and an **Origin Server Resolution** scheme. You can further define control by assigning a configured **Virtual Player** and **Cache Tuning Policy**. The service instance is referenced via the URL in the HTTP request to Media Flow Controller.

For example, if you are serving content through Media Flow Controller for media under the following directories from your origin library:

- `example.com/videos/trg`
- `example.com/videos/UGC`
- `example.com/videos/premiumcontent`

You can create three service websites: **TRG**, **UGC**, and **Premium**, each with a different set of delivery policies.



NOTE: See the *Juniper Networks Media Flow Controller Administrators Guide* for detailed information about configuring and deploying transparent proxy services.

- Related Documentation**
- [Creating Network Optimization Services on page 19](#)
 - [Service Design Overview on page 4](#)
 - *Provisioning Services Overview*
 - *Media Flow Activate Overview*
 - *Understanding Media Flow Controller Management with Media Flow Activate*
 - *Quick Reference to Tasks in Media Flow Activate*

HTTP Reverse Proxy Services Overview

This topic describes what HTTP reverse proxy services are used for and what information you need to know before creating a service.

When you configure Media Flow Controller for HTTP reverse proxy services, it reduces network and CPU load on the origin server by serving previously retrieved content, which enhances user experience by decreasing latency.

The website is referenced via the URL in the HTTP request to Media Flow Controller.



NOTE: See the *Juniper Networks Media Flow Controller Administrators Guide* for detailed information about reverse proxy deployments.

Related Documentation

- [Creating HTTP Reverse Proxy Services on page 30](#)
- [Provisioning Services Overview](#)
- [Creating Reverse Proxy Service XML Files for Import on page 42](#)
- [Creating Reverse Proxy Service XML Files for Export on page 41](#)
- [Purging Content from Media Flow Controller Devices on page 14](#)
- [Actions on Services on page 15](#)
- [Quick Reference to Tasks in Media Flow Activate](#)

Content Ingest Services Overview

With the content ingest service (crawler), you can preload content from origin servers at predefined time intervals. The crawler searches parent node—that is, Media Staging Servers (MSSs) or other origin servers—for content that can be downloaded to Media Flow Controllers. (MSSs are repositories where service providers store permanent content.) The content on parent nodes are then crawled by edge nodes running the standard Media Flow Controller software.

The crawler discovers the requested URLs at the origin server, downloads the discovered URL content from the origin server, and caches the content on Media Flow Controllers.

Related Documentation

- [Creating Content Ingest Services on page 51](#)

PART 2

Administration

- [Managing Provisioned Service Templates with Media Flow Activate on page 11](#)

CHAPTER 2

Managing Provisioned Service Templates with Media Flow Activate

- [Managing Provisioned Services on page 11](#)
- [Purging Content from Media Flow Controller Devices on page 14](#)
- [Actions on Services on page 15](#)

Managing Provisioned Services

Purpose View the status of all devices provisioned with a service on the basis of the status of the provisioned service.

Action To view the status of a provisioned service:

1. Select a service from the **Service Design** workspace (that is, from the **Network Optimization Services** page, **HTTP Reverse Proxy Services** page, or the **Content Ingest Services** page).
2. On the **Actions** list, select **Manage Provisioned Devices**. The **Manage Provisioned Devices** page is displayed, showing each Media Flow Controller provisioned with the service you selected in Step 1.

Meaning The **Manage Provisioned Devices** page displays the following status information for the selected service:

- **Name**—Media Flow Controllers provisioned with the service
- **IP Address**—IP addresses of the Media Flow Controllers provisioned with the service
- **Resource pool**—Resource pools associated with the Media Flow Controllers for the specific service (reverse proxy)



NOTE: If the service is not associated with a user-defined resource pool, the service is associated with the “Global” resource pool, by default. For more information about resource pools, see *Resource Pools Overview*.

- **Service Status**—Whether the service is active or inactive

- **Managed Status**—Whether the device inventory information in the Junos Space database matches the current configuration information on the Media Flow Controller device. The "Managed Status" of the device can be one of the following:
 - **In Sync**—Indicates that the Media Flow Controller device configuration and the configuration information in the Junos Space database are in sync. It is recommended that you provision the services, only when the configurations are in sync.
 - **Out of Sync**—Indicates that the Media Flow Controller device configuration and the configuration information in the Junos Space database are out of sync. This usually happens when configurations are made to the device but are not yet committed. Wait till the configurations are in sync before you provision a service to a device.
 - **Synchronizing**—Indicates that the resync job has begun. The "Managed Status" of the device changes to "In Sync" after the resync job has completed.
 - **Sync Failed**—Indicates that resynchronization has failed.

To resolve this issue, try resynchronizing the managed device by following the steps mentioned in the "Resynchronizing Managed Devices" section of the *Junos Space Network Application Platform User Guide*. If resynchronization does not rectify the issue, you must delete the device and rediscover it.

- **Device Status**—Whether the device is Up (discovered and functioning); or Down (not functioning)
- **Provisioning Status**—Whether the provisioning job is Successful (completed) or Failed (not completed) and the Provision Job identification number (Click the Provision job ID on the Job Management > Manage Jobs page for more details about that provisioning job.)

In the **Manage Provisioned Devices** page, you can sort the data and even choose what columns you want to display by:

- Mousing over a column and clicking the list.
- Selecting **Sort Ascending** or **Sort Descending** to sort the data in ascending or descending order.
- Selecting **Columns** and choosing the columns to display. By default, the following columns are not displayed on the **Manage Provisioned Devices** page: **Id**, **Configuration**, and **Ref**. Select these columns, if you want the information for these columns to be displayed, as well.

Use the **Search** option to display specific Media Flow Controllers by filtering using their names or tags.

From this page, you can select Media Flow Controllers and then select one of the following options:

- **Provision Again**—Provision the service again. Select this option, if the **Provisioning Status** is Failed, or if you have modified the service.
- **De-provision**—Remove the association of the device with this service. A confirmation dialog box is displayed; click **Ok** to complete the deletion. First, the service configurations

are deleted from the selected Media Flow Controllers; then the service association with the selected Media Flow Controllers is deleted from Media Flow Activate.

You may consider deprovisioning a service for the following reasons:

- When you no longer want to cache the content for a website and you want to remove the service completely from Media Flow Controller.
- When any of the Media Flow Controllers is in inconsistent state due to some error.
- **Activate**—Activate the service. A newly created service is inactive by default and you must explicitly activate it.

When a service is provisioned for the first time, the service is in the active state.

- **De-activate**—Deactivate the service. Media Flow Controller drains the connections when a service is deactivated. No new connections are accepted and no new requests are accepted in the current connections. Any existing traffic is brought to a graceful shutdown.

You can deactivate a service when you want to make numerous changes to the service configuration—for example, when you want to update the website and you do not want visitors to the website during the update.



.....
CAUTION: Deactivation of a service disrupts the service.
.....

- **Cancel**—Exit the **Manage Provisioned Devices** page; no changes are made.

See the *Juniper Networks Media Flow Controller Administrators Guide* for detailed information about provisioning the transparent or reverse proxy services (deployments).

Related Documentation

- [Service Design Overview on page 4](#)
- [Creating Network Optimization Services on page 19](#)
- [Creating HTTP Reverse Proxy Services on page 30](#)
- [Creating Content Ingest Services on page 51](#)
- [Provisioning Services Overview](#)
- [Media Flow Activate Overview](#)
- [Quick Reference to Tasks in Media Flow Activate](#)

Purging Content from Media Flow Controller Devices

You can purge all or specific content, which you no longer need to cache, from selected Media Flow Controller devices.

To purge content from selected Media Flow Controller devices:

1. From the left navigation panel, click the plus sign (+) adjacent to **Service Design**.
2. Click **HTTP Reverse Proxy Services**. The **HTTP Reverse Proxy Services** page is displayed.
3. Select the services for which you want to purge content. From the **Actions** list, select **Purge Content**. The **Purge Content** configuration page is displayed.
4. In the **Content to Purge** area, select one of the following options:

- **All**—Purge all content.
- **Directories**—Enter a regular expression. Enter a URL until the folder name. For example, <http://abc.com/videos>.

You may choose to purge directories that exactly match this value or at least contain a part of this value.

- **Files**—Enter a regular expression. Enter a URL that includes the filename. For example, <http://abc.com/videos/hello.flv>.

You may choose to purge files that exactly match this value or at least contain a part of this value.

- Select the **Soft Purge** check box. You can select this option to revalidate the content with the origin server instead of deleting content from Media Flow Controllers. Select this option only if you have selected the **All** option.
5. In the **Select Devices** area, select the Media Flow Controllers.
 6. Click one of the following buttons:
 - **Delete**—Proceed with the purge and close the page. The specified content from the selected Media Flow Controllers is purged.



CAUTION: The content is not purged if you have configured a query string or a header in the HTTP reverse proxy service to be matched in the incoming request URL.

- **Cancel**—Cancel the purge and close the page.

Related Documentation

- [HTTP Reverse Proxy Services Overview on page 8](#)
- [Service Design Overview on page 4](#)
- [Provisioning Services Overview](#)
- [Media Flow Activate Overview](#)

- *Quick Reference to Tasks in Media Flow Activate*

Actions on Services

From the **Content Ingest Services**, **Network Optimization Services**, or **Rproxy Services** page, you can perform some or all of the following actions on services by clicking the links on the **Actions** list. You have to select the services before performing any actions on them:

- **Modify Service**—Click this link to modify all configuration settings other than the service name.

You can modify only one service at a time.

- **Delete Service(s)**—Click this link to delete one or more services. Deletion removes the services totally from Media Flow Activate.

If the service is provisioned, deprovision the service from all the devices that are running the service before deleting the service. For more information about deprovisioning a service, see [“Managing Provisioned Services” on page 11](#).

- **Copy Service**—Click this link to create a duplicate of the service with a different name. When you select this action, you are prompted for a name for the new service.
- **Export Service**—Click this link to export a created service as an XML file.

You can use your preferred XML editor to make the necessary changes to the XML file and later import it back to Media Flow Activate. You can then reprovision the service to Media Flow Controller with the updated configuration. However, if you want to import this configuration to another Media Flow Activate server, you have to remove the service ID information before the import.

- **Import Service**—Click this link to import a service as an XML file and provision the service on to Media Flow Controllers.
- **Manage Provisioned Devices**—Click this link to view the status of all devices provisioned with a service and, if needed, perform further actions, such as reprovisioning, deprovisioning, activating, or deactivating the service on specific Media Flow Controllers. For more information about managing the services that are provisioned, see [“Managing Provisioned Services” on page 11](#).
- **Tag Service**—Click this link to tag the services.
- **Untag Service**—Click this link to untag the selected tags from the specific service.
- **View Tags**—Click this link to view the tags associated with a specific service.
- **Purge Content**—Click this link to purge all or specific content, which you no longer need to cache, from selected Media Flow Controller devices. For more information about purging content, see [“Purging Content from Media Flow Controller Devices” on page 14](#).
- **Show Service Details**—Click this link to view the configuration of the specific service.

Related Documentation

- [Network Optimization Services Overview on page 7](#)
- [HTTP Reverse Proxy Services Overview on page 8](#)

- [Content Ingest Services Overview on page 8](#)
- [Service Design Overview on page 4](#)
- *Media Flow Activate Overview*
- *Quick Reference to Tasks in Media Flow Activate*

PART 3

Configuration

- [Creating Service Templates for Media Flow Controllers with Media Flow Activate on page 19](#)

CHAPTER 3

Creating Service Templates for Media Flow Controllers with Media Flow Activate

- [Creating Network Optimization Services on page 19](#)
- [Creating Network Optimization Service XML Files for Import on page 26](#)
- [Creating HTTP Reverse Proxy Services on page 30](#)
- [Creating Reverse Proxy Service XML Files for Export on page 41](#)
- [Creating Reverse Proxy Service XML Files for Import on page 42](#)
- [Creating Content Ingest Services on page 51](#)

Creating Network Optimization Services

Use the **Service Design** workspace to create content delivery services. You create these services for the types of media you deliver.

Before you begin configuring a **Network Optimization** website media delivery service, you need the following information:

- The regular expression (regex) for the **Domain (Regex)** that is used in requests for the media for this service; see [Table 3 on page 25](#) for example domain regex values. You also need the file **Path** to the media in that domain that you expect to deliver.
- The **Precedence** to give this service among all the configured services. The higher the **Precedence** value, the lower the precedence.
- For cache misses (requests for media not in the cache), how you want the service to obtain the requested media.
- The **Virtual Player** to manage video delivery policies for that site. This is needed when the site being cached serves HTTP video. A single virtual player can be used for multiple sites. See *Creating Virtual Players* to create virtual players.
- The **Cache Tuning Policy** to fine-tune the cache settings based on the network conditions and caching requirements. See *Creating Cache-Tuning Policies* to create cache-tuning policies.

Create a separate **Network Optimization** website delivery service for each media repository that you have.

To configure Network Optimization services on the **Service Design** workspace:

1. From the left navigation panel, click the plus sign (+) adjacent to **Service Design**.
2. Click **Network Opt Services**. The **Network Optimization Services** page is displayed.
3. Click **Add Service Instance**. The **Add Service Instance** page is displayed.

Figure 1: Create Network Optimization Service—General Tab

Add Service Instance

General | Dynamic URIs

Name: !

Description:

Domain: ! ☐ Regex

Order of serving client request: Last in first out ▼

Ignore all object correlation validators: ☐

Custom cache-control header:

Path: /(all files) ☐ Regex

Precedence: 0 ▲▼

Origin Server Resolution: Follow Host Header ▼ ☒ Use Client IP

Tunnel All: ☐

Virtual Player Name: ▼

Cache Tuning Policy: ▼

Policy Script: ▼

Revalidate always: ☐ Revalidation Mode: ☐ Offline ☐ Inline

OK Cancel

4. On the **General** tab, specify the following information:
 - a. **Name** for the service and **Description** (optional) of the service.
 - b. **Domain**—Enter the defined fully qualified domain name (FQDN) that is matched with the incoming HOST header to identify the request. This field is mandatory.

Select **Regex** for Media Flow Controller to treat the entry in the **Domain** field as a regex entry. Enclose all regex entries in double quotation marks—for example, a regex for www.example.com plus example.com could be: “**^.*\example\.com**”. The regex is written against the absolute path portion of the URL. For example, given the following URL: http://abc.com:8080/index.html, /index.html would be the absolute path portion. See [Table 3 on page 25](#) for more examples of regex entries.
 - c. **Order of serving client request**—Select the policy that determines the order in which the objects are ingested from the RAM cache into the disk cache when handling bulk requests.

To avoid clients waiting for data while Media Flow Controller ingests data into the disk cache, the data serving path and the ingestion path are decoupled in Media Flow Controller. While objects are being served from the RAM cache, the ingestion process in Media Flow Controller picks these objects from the RAM cache and ingests them into the disk cache. The order in which the ingestion process picks these objects for ingestion can be tuned:

- **Last in first out**—(Default) The object served last is ingested into the disk cache first. Because the object is selected for ingestion immediately after it has been served to the client, the probability of finding the object in the RAM cache is high, which prevents refetching of the object from the origin server.
- **First in first out**—The object served first is ingested into the disk cache first. Here, the objects are ingested into the disk cache in the order in which they have been served to the clients from the RAM cache. However, this feature may introduce additional fetches from the origin server if the objects are evicted from the RAM cache before they are ingested into the disk cache.

Recommendations: For caching adaptive bit-rate streaming videos (such as Microsoft Smooth Streaming videos or Apple HTTP Live Streaming videos), it is recommended that you set the configuration to **First in first out**. For Network Optimization service deployments, it is recommended that you set the configuration to **Last in first out**.

Example for “First in first out”: Consider that in a given instant (say, at t_0 seconds), Media Flow Controller receives 100 requests for 100 objects. After Media Flow Controller fetches these objects from the origin server, the order in which these objects are ingested into disk cache is determined by whether Media Flow Controller is configured for **First in first out** or **Last in first out**. If it is configured for **First in first out**, then it ingests these objects into disk cache starting from the first object that was served to the client, followed by the second object, and so on. At t_1 seconds, if Media Flow Controller receives another 100 new requests and if it has cached only 80 objects at the end of t_0 seconds, the cacheable disk queue becomes 120 objects and it starts caching from the eighty-first object. At t_2 seconds, if Media Flow Controller receives another 100 new requests and if it has cached only 160 objects at the end of t_1 seconds, the cacheable disk queue now becomes 140 objects and it starts caching from the hundred and sixty-first object.

- d. **Ignore all object correlation validators**—Select this check box for Media Flow Controller to ignore the object validation fields (such as ETag and Last Modified headers) in the origin server response when it validates an object.

When the ETag or the Last Modified header is different from the previously cached origin server response, by default, Media Flow Controller assumes that the object is of a different version and deletes the cached object and replaces it with the modified object. If you select the **Ignore all object correlation validators** check box, Media Flow Controller ignores any changes to these headers in the origin server response and serves the previously cached object, provided that it has not expired. However, if the object has expired, Media Flow Controller deletes the existing cached object, caches and serves the object fetched from the origin server.

Example: Media Flow Controller uses the ETag header or the Last Modified header in the origin server response to validate whether the object that is currently cached is of the same version as that in the origin server. If the object has been modified in the origin server, then the ETag header value or the Last Modified header value sent by the origin server is different from the value that was previously sent by the origin server. If the values are different, Media Flow Controller deletes the existing cached content, fetches the modified content from the origin server, and caches it.

At the time of caching the origin server response, Media Flow Controller associates a version number with each of the object that is being cached. This version number is generated from the ETag header value of the origin server response, provided that the ETag header is present; otherwise, it is generated from the Last Modified header value, provided that the Last Modified header is present. If both are absent, then Media Flow Controller generates a random version number. Though this is a good mechanism for validating the freshness of the object, you may find that in CDN deployments, the objects might not be cached at all in Media Flow Controller when the client is making byte-range requests.

Scenario 1: When you have a load balancer set up, where multiple origin servers might respond to byte-range requests, it is possible that the ETag header values are different in the responses sent by each of these origin servers. Consider a client requesting a 10-MB object in byte-range requests of 2 MB. For the first request of 0 to 2 MB, if there is a cache miss, then Media Flow Controller forwards this request to the origin server. If origin server 1 (OS1) responds to this request, Media Flow Controller caches this response and also generates a version number from the ETag header value of the response, and associates it with the object. For the next request of 2 to 4 MB, upon a cache miss, if origin server 2 (OS2) responds, it is likely that the ETag header sent by this origin server might be different. At the time of caching, Media Flow Controller assumes that this is a different version of the object and deletes the previously cached 0 to 2 MB. Media Flow Controller tunnels the 2- to 4-MB response because it cannot cache this content without caching the previous 0 to 2 MB. The remaining byte-range requests are also tunneled to the client. Here, Media Flow Controller does not cache the entire 10-MB object. This is one scenario where even though the response is cacheable, Media Flow Controller does not cache the response.

Scenario 2: Assume the origin server does not send the ETag or the Last Modified header in its response to the byte-range requests. Consider a client is requesting a 10-MB object in byte-range requests of 2 MB. When Media Flow Controller caches the first request of 0 to 2 MB, it generates a random version number and associates it with the cached object. This is because there is no ETag or Last Modified header in the origin server response to generate the version number. When Media Flow Controller tries to cache the next request of 2 to 4 MB, it generates another random version number to associate with the object. At the time of caching, Media Flow Controller compares the version numbers of 0 to 2 MB and 2 to 4 MB of the object. Because the version numbers are different, Media Flow Controller deletes the cached 0 to 2 MB and tunnels the 2 to 4 MB of the object directly to the client. The remaining byte-range requests are also tunneled to the client.

- e. **Custom cache-control header**—Specify a custom cache-control header name in the origin server response to be used for determining cache expiry. Assume that you have configured “xyz” for this field. If the origin server response contains the “xyz” header, then Media Flow Controller sets the expiry time of the object to the value contained in this header. If the xyz header value is 2400, then Media Flow Controller serves the object from its cache for the next 40 minutes from the time of caching. After this duration, the object is considered expired.

Example: Some origin servers do not send the expiry information of an object to intermediate proxies using standard HTTP headers, such as the Expires header or the Cache-control:max-age header. Instead, they send it by using custom headers. Media Flow Controller expects the object expiry information from these standard HTTP headers and when these headers are not present in the origin server’s response, it automatically assumes that the origin server has not sent this information. Media Flow Controller then tries to apply the expiry value configured in the **cache-age default** or **cache-age content-type any** CLIs (that is, the values you have configured for **Default Cache Age** or **Cache Age Override** in the Cache tuning policy). However, for Media Flow Controller to use the expiry information present in the custom header of the origin server’s response, you must specify the custom header name in the **Custom cache-control header** field. Media Flow Controller now detects that this custom header contains the expiry information for the object and it proceeds to set the object expiry time to this value.

For example, consider that the origin server sends a custom header, “x-cache-control:max-age=3600.” For Media Flow Controller to detect that the custom header, “x-cache-control,” in the origin server’s response contains the expiry information of the object, you need to set the value of the **Custom cache-control header** field to “x-cache-control.” Media Flow Controller then serves the object from its cache for the next one hour from the time of caching.

- f. **Path**—Enter the actual file path where the media files are located. The default path is “/” (slash), which refers to all files in the configured **Domain**.

Select **Regex** for Media Flow Controller to treat the entry in the **Path** field as a regex entry.

- g. **Precedence**—Select a value to break ties between services defined with the same **Domain**. The lower the number, the higher the preference for that service; 0 is the default and highest.

- h. **Origin Server Resolution**—Select an option to specify how Media Flow Controller determines and accesses the origin server for cache misses; choose one of the following options:

- **Follow Host Header**—Use the host header of the incoming request as the origin server.
- **Destination IP**—Use the destination IP address of the incoming request as the origin server.

- i. **Use Client IP**—(Default) Select this check box to use the client IP address in place of the origin server’s IP address in the request.

- j. **Tunnel All**—Select this check box to tunnel all incoming client requests directly to the origin server without further processing. By default, this feature is disabled.

When you enable this feature, all caching-related configuration is ignored and the requests are directly tunneled to the origin servers.

- k. **Virtual Player Name**—Select a defined virtual player; see *Creating Virtual Players* to create a virtual player.

- l. **Cache Tuning Policy**—Select a defined policy; see *Creating Cache-Tuning Policies* to create a cache-tuning policy.

- m. **Policy Script**—Associate a policy script from a list of previously added policy scripts.

You typically associate a policy script when you want to have a fine granular control over the various features of Media Flow Controller at runtime.

Make sure that the policy script file is available on the local system from where Junos Space is accessed.

- n. **Revalidate always**—Select this check box for Media Flow Controller to revalidate its cached object for every client request.

When Media Flow Controller receives a request from the client and this feature is enabled, Media Flow Controller first serves the object from its cache if the object has not expired. However, in parallel, Media Flow Controller sends a revalidation request to the origin server to determine whether the cached object has been modified in the origin server or not. Depending on the origin server's response, if the object has not been modified, Media Flow Controller updates only the expiry time of the object; otherwise, Media Flow Controller deletes the existing cached object and caches the latest modified object. This is the default behavior—that is, the objects are revalidated offline. However, to revalidate the objects before serving them to the clients, set **Revalidation Mode** to **Inline**.

- o. From the **Dynamic URIs** tab, configure parameters using which Media Flow Controller can identify dynamic URIs and ensure the delivery of the correct content. An incoming request having a URI that matches the configured regex value (specified in the **URL to Match** field) is matched to a cache index string (specified in the **Map to** field). For more information about URI remapping, see "Understanding Dynamic URI Mapping."



NOTE: **URL to Match** value must always be configured along with **Map To**. **Tunnel Unmatched** can be selected only if the **URL to Match** and **Map To** values are provided.

- **URL to Match**—Configure a regex expression to match the request URL. The `<regex>` has a maximum character limit of 1024 characters (including NULL); if the URI exceeds this limit, the request is tunneled. PCRE regex is not allowed; whereas GNU regex is allowed. The default value is NULL.

Example: `/get/[^/]+/[^/]+/[^/]+/(.*)`

- **Map to**—Configure a map-string value (a string to map or rewrite the URL portion of the request) when a match is found. The <map_string> has a maximum character limit of 2048 characters (including NULL). The default value is NULL.

Example: /\$1

- **Tunnel Unmatched**—Select the check box to tunnel the request when no regex match is found.

5. Click **Ok** or **Cancel**. **Ok** instantiates the values you set and closes the page; **Cancel** closes the page without making any changes.



NOTE: To add a service website, go to the **Service Design > Network Opt Services** page. On the **Actions** list, click **Import Service** (this link is active only when you have not selected any services from this page). You use a defined XML file to import a service website.

Table 3: Example: Domain Regex Values

Regex	Matches
"www.example.com example.com"	www.example.com
".*example.com"	example.com
"^[a-f,0-9]{8}\\. (origin\\. cdn\\.)?cms\\. example\\. com:80\$"	abcdef02.origin.cms.example.com:80 abcdef23.cdn.cms.example.com:80 abcdef09.cms.example.com:80
"^cms[0-9]{3}\\. (dc2 qcg7)\\. example\\. com:80\$"	cms123.dc2.example.com:80 cms079.qcg7.example.com:80
"^orig. (sv1 qcg1 qcg5)\\. example\\. com:80\$"	orig.sv1.example.com:80 orig.qcg1.example.com:80 orig.qcg5.example.com:80
"^(cms[0-9]{3}).*(qcg[0-9]+ sv1 ch1 dc2 af1)\\. example\\. com:80\$"	cms123.x.y.qcg0.example.com:80 cms257.x.y.qcg01.example.com:80 cms379.x.y.sv1.example.com:80 cms222.x.y.ch1.example.com:80 cms876.x.y.dc2.example.com:80 cms343.x.y.af1.example.com:80



NOTE: See the *Juniper Networks Media Flow Controller Administrators Guide* for detailed information about configuring and deploying transparent proxy services.

Related Documentation

- [Media Flow Activate Overview](#)
- [Understanding Media Flow Controller Management with Media Flow Activate](#)
- [Service Design Overview on page 4](#)
- [Network Optimization Services Overview on page 7](#)
- [Creating Network Optimization Service XML Files for Import on page 26](#)
- [Quick Reference to Tasks in Media Flow Activate](#)

Creating Network Optimization Service XML Files for Import

Use the following XML structure to create transparent proxy forms that can be imported to Media Flow Activate. These forms are then available for inclusion in a website service provisioned to selected Media Flow Controllers.

Service Network Optimization “Content Direct” XML Schema

When you customize the following XML structure for your needs, give the **contentDirectDefinition** element a **name** and **description**, then modify the options and elements. These parameters are described in [Table 4 on page 27](#). See the following simple XML schema for transparent proxy service. For information about a schema with a cache-tuning policy and virtual player configuration, see *Sample XML Schema for Network Optimization and Reverse Proxy Services*.

```

Simple XML Schema for Transparent Proxy Service
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<contentDirectDefinitionCatalog>
  <contentDirectDefinitions>
    <contentDirectDefinition>
      <name>site1</name>
      <description>test</description>
      <properties>
        <domain>google</domain>
        <isDomainRegex>>false</isDomainRegex>
        <match>
          <path>
            <allFiles>>true</allFiles>
          </path>
        </match>
        <precedence>1</precedence>
        <clientRequest>
          <cacheIndex>
            <urlToMatch>urlToMatch</urlToMatch>
            <mapTo>mapto</mapTo>
            <tunnelUnMatched>>false</tunnelUnMatched>
          </cacheIndex>
        </clientRequest>
      </properties>
    </contentDirectDefinition>
  </contentDirectDefinitions>
</contentDirectDefinitionCatalog>

```

```

</clientRequest>
<originServerResolution>USE_DESTINATION_ADDR</originServerResolution>
<useClientIP>true</useClientIP>
</properties>
<tunnelAll>true</tunnelAll>
<revalidateCacheHit>>false</revalidateCacheHit>
</contentDirectDefinition>
</contentDirectDefinitions>
</contentDirectDefinitionCatalog>

```

Table 4: contentDirectDefinition Options and Elements

Option	Option or Sub-Element	Option or Sub Sub-Element	Description	CLI Equivalent
name	none	none	Enter a name for the contentDirectDefinition element.	namespace <name>
domain	none	none	Enter the name of the domain against which incoming requests are referenced; media files are located in this domain.	namespace <name> domain regex <regex>
match	path	allFiles	Specify that all files in the given domain may be requested. Alternatively, use the specific files option.	namespace <name> match uri regex <regex>
		specificFiles	Specify a regex path to the files that are requested. Alternatively, use the all files option.	namespace <name> match uri <uri-prefix> /
precedence	none	none	The lower the value, the higher the precedence for this namespace, should there be a conflict. See “Understanding Precedence” in “ Service Design Overview ” on page 4 for more information about using the “precedence” parameter.	namespace <name> match uri regex <regex> [precedence <number>]

Table 4: contentDirectDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub Sub-Element	Description	CLI Equivalent
clientRequest	cacheIndex	urlToMatch	Configure a regex expression to match the request URL. The regex value for url to match has a maximum character limit of 1024 characters (including NULL); if the URI exceeds this limit, the request is tunneled. Only one url to match expression per service is allowed. No PCRE regex is allowed; only GNU regex is allowed	namespace <name> delivery protocol http client-request cache-index url-match <regex> map-to <map_string> no-match-tunnel
		mapTo	Configure a string to map, or rewrite, the URL portion of the request when a match is found. The mapTo value has a maximum character limit of 2048 characters (including NULL).	
		tunnelUnMatched	Tunnel the request when no regex match for url to match is found.	
originServerResolution	USE_HOST_HEADER	none	Use the host header of the incoming request as the origin server (in case of a cache miss). Optionally, set use client IP to use the client IP address in place of the origin server's IP address in the request.	namespace <name> origin-server http follow header <header>
	USE_DESTINATION_ADDR	none	Use the destination IP address of the incoming request as the origin server. Optionally, set use client IP to use the client IP address in place of the origin server's IP address in the request.	

Table 4: contentDirectDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub Sub-Element	Description	CLI Equivalent
useClientIP	none	none	Use the client IP address in place of the origin server's IP address in the request.	namespace <name> origin-server http follow header <header> [use-client-ip] or namespace <name> origin-server http follow dest-ip [use-client-ip]
tunnelAll	none	none	Tunnel all incoming client requests directly to the origin server without further processing.	namespace <name> delivery protocol http client-request tunnel-all
revalidateCacheHit	none	none	Set requested objects in cache to always trigger a timestamp revalidation. When this parameter is enabled, performance is impacted because every transaction is revalidated.	namespace <name> delivery protocol http client-request cache-hit action revalidate-always

Some **url to match** regex examples:

- Regex **url to match** example with no substring address, only \$0 returned:

```
/opt/nkn/bin/nknregex -m -e '/videoplayback\?.*\&id=[^\&]+.*' -d
'/videoplayback?xxx&id=1xxuuu'

match[0]: /videoplayback?xxx&id=1xxuuu
```
- Regex **url to match** example with two substrings denoted, \$0, \$1, and \$2 returned:

```
/opt/nkn/bin/nknregex -m -e '(/videoplayback\?.*)\&id=([^\&]+).*' -d
'/videoplayback?xxx&id=1xxuuu'

match[0]: /videoplayback?xxx&id=1xxuuu match[1]: /videoplayback?xxx match[2]:
1xxuuu
```
- Regex **url to match** example with two substrings denoted, \$0, \$1, and \$2 returned:

```
/opt/nkn/bin/nknregex -m -e '(/videoplayback\?.*)\&id=([^\&]+).*' -d
'/videoplayback?xxx&id=1xxuuu'

match[0]: /videoplayback?xxx&id=1xxuuu match[1]: /videoplayback?xxx match[2]:
1xxuuu
```

Some **map to** string examples:

- Using regex example 3:

`/abc/$1/$2 => /abc//videoplayback?xxx/1xxuuu`

- Using regex example 3:

`/XXX$0/$1/$2 => /XXX/videoplayback?xxx&id=1xxuuu//videoplayback?xxx/1xxuuu`

- Using regex example 3:

`$$$1/$2 => $/videoplayback?xxx/1xxuuu`

- Using regex example 3:

`$$$1$$ => $/videoplayback?xxx$`

See these topics for additional information:

- *Sample XML Schema for Network Optimization and Reverse Proxy Services*
- [Creating Network Optimization Services on page 19](#)
- [Service Design Overview on page 4](#)
- *Media Flow Activate Overview*
- *Understanding Media Flow Controller Management with Media Flow Activate*
- *Quick Reference to Tasks in Media Flow Activate*

See the *Juniper Networks Media Flow Controller Administrator's Guide* for Media Flow Controller CLI command descriptions.

After you have customized and saved a Transparent Proxy XML configuration file, go to the Service Design > Network Opt Services page. On the **Actions** list, click **Import Service**. From the **Import Service** dialog box, navigate to the XML file you created and click **Ok**. The Transparent Proxy configuration provided in the XML file is made available to Media Flow Activate and can be included in service designs for provisioning to selected Media Flow Controllers through the **Service Provisioning** workspace.

Creating HTTP Reverse Proxy Services

Use the **Service Design** workspace to create content delivery services. You create these services for the types of media you are delivering.

Before you begin configuring an **HTTP Reverse Proxy** website media delivery service, you need the following:

- Regular expression for **Domain (Regex)** that is used in requests for the media for this service (See [Table 3 on page 25](#) for example domain regex values.)
- Criterion to be matched with the incoming URL to identify the request. This criterion can be "URI Path, Header name, or Query String."
- **Precedence** you want to give this service among all the configured services (for the same domain)

- For cache misses (requests for media not in the cache), how you want the service to obtain the requested media (by configuring the origin server settings).
- **Virtual Player** you use to manage video delivery policies for that site (See *Creating Virtual Players* to create virtual players.)
- **Cache Tuning Policy** you use to fine-tune the cache settings based on the network conditions and caching requirements (See *Creating Cache-Tuning Policies* to create cache-tuning policies.)
- **Policy Script** you use to customize the various features in Media Flow Controller at runtime
- Order in which the objects must be ingested from RAM cache to disk cache
- Dynamic URI parameters, which Media Flow Controller uses to identify dynamic URIs and deliver correct content
- Consider whether to:
 - Authenticate the origin servers before downloading the content
 - Monitor the HTTP and HTTPS transactions handled by Media Flow Controller
 - Classify the traffic by setting the DiffServ code point (DSCP) value
 - Pin the objects to the cache
 - Modify request and response headers
 - Ignore the validation headers when revalidating an object
 - Use a custom header to determine cache expiration
 - Send a “503 Service Unavailable” response when Media Flow Controller is unable to service a request because the resource pool concurrent session limit has been reached or exceeded

To configure HTTP reverse proxy services on the **Service Design** workspace:

1. From the left navigation pane, click the plus sign (+) adjacent to **Service Design**.
2. Click the plus sign (+) adjacent to **HTTP Reverse Proxy Services**.
3. Click **Add Service Instance**. The **Add Service Instance** page is displayed.
4. On the **General** tab, specify the following information:

Figure 2: Create HTTP Reverse Proxy Service—General Tab

Add Service Instance

General | Design Elements | Object Handling | Dynamic URI | Origin Headers | Client Headers

Service Name: !

Description:

Domain: ! ☐ Regex

Order of serving client request: Last in first out

Match Conditions

☐ Regex

☒ URI Path:

☐ Query String Match Name: Value:

☐ Header Match Name: Value:

Precedence:

Origin Server

☒ FQDN ☐ Origin map

Server Name: !

Port: !

Secure Authentication: ☐

OK Cancel

- **Service Name**—Enter the name of the service or website that you want to configure, which must be unique.
- **Description**—(Optional) Enter a description of the service or website.
- **Domain**—Enter the defined fully qualified domain name (FQDN) or regular expression that is matched with the incoming HOST header to identify the request.

Select **Regex** if you want the MFA to treat any entries in the **Domain** field as regex entries. Enclose all regex entries in double quotation marks.

- **Order of serving client request**—Enter the policy that determines the order in which the objects are ingested from RAM cache into disk cache when Media Flow Controller handles bulk requests.

To ensure that clients do not need to wait for data while Media Flow Controller ingests data into disk cache, the data serving path and the ingestion path are decoupled in Media Flow Controller. During the time that objects are being served from RAM cache, the ingestion process in Media Flow Controller picks these objects from RAM cache and ingests them into disk cache. The order in which the ingestion process picks these objects for ingestion can be tuned:

- **Last in first out**—(Default) The object served last is ingested into disk cache first. Because the object is selected for ingestion immediately after it is served to the

client, the probability of finding the object in RAM cache is high. This prevents the object from being refetched from the origin server.

- **First in first out**—The object served first is ingested into disk cache first. The objects are ingested into disk cache in the order in which they are served to the clients from RAM cache. However, this feature may introduce additional fetches from the origin server if the objects are evicted from RAM cache before they are ingested into disk cache.



NOTE: Recommendations: For caching adaptive bit rate streaming videos (such as Microsoft Smooth Streaming videos or Apple HTTP Live Streaming videos), it is recommended that you set the configuration to **First in first out**. For Network Optimization service deployments, it is recommended that you set the configuration to **Last in first out**.

Example for “First in first out”: Consider that in a given instant (for example, at t_0 seconds), Media Flow Controller receives 100 requests for 100 objects. After Media Flow Controller fetches these objects from the origin server, the order in which these objects are ingested into disk cache is determined by whether Media Flow Controller is configured for **First in first out** or **Last in first out**. If it is configured for **First in first out**, then it ingests these objects into disk cache starting from the first object that was served to the client, followed by the second object, and so on. At t_1 seconds, if Media Flow Controller receives another 100 new requests and if it has cached only 80 objects at the end of t_0 seconds, the cacheable disk queue becomes 120 objects and it starts caching from the eighty-first object. At t_2 seconds, if Media Flow Controller receives another 100 new requests and if it has cached only 160 objects at the end of t_1 seconds, the cacheable disk queue now becomes 140 objects and it starts caching from the hundred and sixty-first object.

- In the **Match Conditions** area, specify the criteria to be matched with the incoming URL to identify the request. The request is serviced only when there is a match. It provides a finer control over the requests that are serviced by Media Flow Controller.

When there is a match, the request is attached to the specific service and the requested object is delivered as per the configured service parameters. If no matching services are found (which can mean that the requested object may not be in the cache), the requests are terminated.

This feature is useful in scenarios where you have multiple services defined for a website and you want to direct the request to the service that is most relevant for the request (such as where you might have configured the virtual player and other parameters that are most suitable to cater to that specific content).

For example, you might have created two reverse proxy services for the domain www.cnn.com to cater to videos and images. Assume that you have created the service *cnnv*, which caches the videos under the *cnn/videos* directory structure. You can specify an additional match criterion of */videos* in the *URI Path* field in the Match Conditions area, so that any incoming request such as www.cnn.com/videos/roundtheworld.flv is attached to the “cnnv” service and is processed by this service.

Specify at least one of the following match conditions:

Select **Regex** if you want Media Flow Controller to treat the entries in the following fields as regex entries.

- **URI Path**—Enter a path to be matched in the incoming request URL. You can enter a string or a regular expression. For example, “/videos.” By default, this option is selected and has a value of “/”, which means that any incoming request with the configured domain name followed by a “/” is matched to this namespace. For example, if the domain name configured for a namespace is “www.yahoo.com,” any incoming request URL containing the string “www.yahoo.com/” is matched and served by this namespace.
- **Query String Match Name**—Enter a query string parameter (name-value pair) to be matched in the incoming request URL.

For example, if you enter “video_id” and “1”, this service is associated with an incoming URL request such as www.cnn.com/videos/get_video?video_id=1.

- **Header Match Name**—Enter a header to be matched in the incoming request.
- **Precedence**—When there are multiple services defined for the same domain, specify a number to determine which service must be given higher precedence. The lower the number, the higher the preference for that service; 0 is the default and the highest.

For example, you might have configured two services: one that stores any generic video and one that stores videos that are specific to the year 2011 (the directory structure in Media Flow Controller would be something like /videos and videos/2011). If the incoming request is for a video created in 2011, because it matches both these services, the service that has the higher precedence is associated with the request. In this case, the service that stores the 2011 videos must be configured with the higher precedence.

- In the **Origin Server** area, you configure the origin servers that Media Flow Controller must access to fetch content upon a cache miss. You must select either FQDN or Origin Map.
 - Select **FQDN** to configure a single origin server:
 - **Server Name**—Enter the IP address or the server name of the origin server.
 - **Port**—Enter the port number through which the server listens for requests. Usually, because the server listens at port 80, you can enter **80** in this field. However, if you want to authenticate the origin servers before downloading the content from the origin servers, enter **443**. For more information about authenticating origin servers, see the information provided for the **Secure Authentication** check box.
 - Select **Origin map** to configure multiple HTTP origin servers.

Select an origin map from the **Name** list. For more information about creating origin maps, see *Creating Consistent Hash Maps* and *Creating Escalation Maps*.

- Select the **Secure Authentication** check box to authenticate the origin servers before downloading content from the origin servers. If you select this check box, Media Flow Controller encrypts the authentication messages and fetches content from the origin servers by using HTTPS. If you do not select this check box, Media Flow Controller does not authenticate the origin servers and fetches content in plain text by using HTTP.

You use this feature when you want to authenticate and download content from origin servers that reside within nontrusted domains.

For the **Secure Authentication** feature to work in Media Flow Controller:

- Set the port to 443.

If you are using the FQDN option to associate an origin server with the service, enter 443 in the **Port** field. If you are associating an origin map with the service, select the origin map that is configured with port number 443 from the **Name** list.

- Configure the ciphers and CA certificates in Media Flow Controller.

For instructions to configure CA ciphers and CA certificates, see the “Setting Up Content Ingest Manager Security” section in the *Juniper Networks Media Flow Controller Administrator’s Guide*.

- Configure **Enable SSL Authentication** from **Media Flow Devices > Device Configuration** for Media Flow Controller.

When you enable the **Secure Authentication** feature:

- Media Flow Controller establishes a TCP connection on port 443 with the origin server.
- The SSL handshake is performed. During the handshake, the origin server sends its digital certificate to Media Flow Controller.
- Media Flow Controller uses the information sent by the server to authenticate the server. If the server is successfully authenticated, Media Flow Controller forwards the client request to the server.

5. On the **Design Elements** tab, specify the following information:

- **Virtual Player Name**—Select a previously defined virtual player with the service; see *Creating Virtual Players*. The virtual player configuration determines how the videos within the specified domain are delivered.
- **Cache Tuning Policy**—Select a previously defined policy with the service; for more information about creating a cache-tuning policy, see *Creating Cache-Tuning Policies*. The policy determines when to promote an object to various cache tiers.
- **Log Profile**—Select a log profile with the service from a list of previously created log profiles.

You associate a log profile with a service when you want to monitor the HTTP and HTTP transactions handled by that service. This information is captured on the log file that is configured in the log profile. Typically, the log file is stored in the LogExport

directory. For more information about access log profiles, see *Understanding Access Log Profiles*.



CAUTION: Though MFA supports creating any number of log profiles, the provisioning of the service fails if the log profile associated with the service exceeds the maximum number of log profiles that a Media Flow Controller can support (which is 32).

- **Policy Script**—Select a policy script from a list of previously added policy scripts.

You typically associate a policy script when you want a fine granular control over the various features of Media Flow Controller at runtime.

Make sure that the policy script file is available on the local system from where Junos Space is accessed.

6. On the **Object Handling** tab, specify the following information:

- **DSCP**—Specify a DSCP value so that Media Flow Controller sets the DSCP field of the IPv4 packets to this value for all HTTP responses sent from Media Flow Controller to the client for this service.

You can enter a value from 0 through 63. The default value is 0 (zero).



NOTE: DiffServ provides a mechanism to classify and mark packets belonging to a specific class. This mechanism proves helpful if you have configured a router to differentiate traffic on the basis of class, which can then be used to provide low latency to critical network traffic such as voice or streaming media, while providing a simple best-effort service to noncritical services such as Web traffic or file transfers.

- **Tunnel All**—Select this check box to tunnel all incoming client requests directly to the origin server without further processing. By default, this feature is disabled.

When you enable this feature, all caching-related configuration is ignored and the requests are tunneled directly to the origin servers.

- **Ignore all object correlation validators**—Select for Media Flow Controller to ignore the object validation fields (such as ETag and Last Modified headers) in the origin server response when it validates an object.

When the ETag or the Last Modified header is different from the previously cached origin server response, by default, Media Flow Controller assumes that the object is of a different version and deletes the cached object and replaces it with the modified object. If you select the **Ignore all object correlation validators** check box, Media Flow Controller ignores any changes to these headers in the origin server response and serves the previously cached object, provided that it has not expired. However, if the object has expired, Media Flow Controller deletes the existing cached object, caches the object fetched from the origin server, and serves it.

Example: Media Flow Controller uses the ETag header or the Last Modified header in the origin server response to validate whether the object that is currently cached is of the same version as that in the origin server. If the object has been modified in the origin server, then the ETag header value or the Last Modified header value sent by the origin server is different from the value that was previously sent by the origin server. If the values are different, Media Flow Controller deletes the existing cached content, fetches the modified content from the origin server, and caches it.

At the time of caching the origin server response, Media Flow Controller associates a version number with each object that is cached. This version number is generated from the ETag header value of the origin server response, provided that the ETag header is present; otherwise, from the Last Modified header value, provided that the Last Modified header is present. If both are absent, then Media Flow Controller generates a random version number. Though this is a good mechanism for validating the freshness of the object, you may find that in CDN deployments, the objects might not be cached at all in Media Flow Controller when the client is making byte-range requests.

Scenario 1: If load balancers are set up, where multiple origin servers might respond to byte-range requests, it is possible that the ETag header values are different in the responses sent by each of these origin servers. Consider a client requesting a 10-MB object in byte-range requests of 2 MB. For the first request of 0 to 2MB, if there is a cache miss, Media Flow Controller forwards this request to the origin server. If origin server 1 (OS1) responds to this request, Media Flow Controller caches this response and also generates a version number from the ETag header value of the response and associates it with the object. For the next request of 2 to 4 MB, upon a cache miss, if origin server 2 (OS2) responds, it is likely that the ETag header sent by this origin server might be different. At the time of caching, Media Flow Controller assumes that this is a different version of the object and deletes the previously cached 0 to 2 MB. Media Flow Controller tunnels the 2- to 4-MB response in addition because it cannot cache this content without caching the previous 0 to 2 MB. The remaining byte-range requests are also tunneled to the client. Media Flow Controller does not cache the entire 10-MB object. This is one scenario where even though the response is cacheable, Media Flow Controller does not cache the response.

Scenario 2: Assume that the origin server does not send the ETag or the Last Modified header in its response to the byte-range requests. In this scenario, consider that a client is requesting a 10-MB object in byte-range requests of 2 MB. When Media Flow Controller caches the first request of 0 to 2 MB, it generates a random version number and associates it with the cached object. This is because there is no ETag or Last Modified header in the origin server response to generate the version number. When Media Flow Controller tries to cache the next request of 2 to 4 MB, it generates another random version number to associate with the object. At the time of caching, Media Flow Controller compares the version numbers of 0 to 2 MB and 2 to 4 MB of the object. Because the version numbers are different, Media Flow Controller deletes the cached 0 to 2 MB and tunnels the 2 to 4 MB of the object directly to the client. The remaining byte-range requests are also tunneled to the client.

- **Custom cache-control header**—Enter a custom cache-control header name in the origin server response to be used for determining cache expiry. Assume that you

have configured “xyz” for this field. If the origin server response contains the “xyz” header, then Media Flow Controller sets the expiry time of the object to the value contained in this header. If the xyz header value is 2400, then Media Flow Controller serves the object from its cache for the next 40 minutes from the time of caching. After this duration, the object is considered expired.

Example: Some origin servers do not send the expiry information of an object to intermediate proxies by using the standard HTTP headers, such as the Expires header or the Cache-control:max-age header. Instead, they send it by using custom headers. Media Flow Controller expects the object expiry information from these standard HTTP headers. When these headers are not present in the origin server’s response, Media Flow Controller automatically assumes that the origin server has not sent this information. Media Flow Controller then tries to apply the expiry value configured in the **cache-age default** or **cache-age content-type any** CLIs (that is, the values you configured for **Default Cache Age** or **Cache Age Override** in the cache-tuning policy). However, for Media Flow Controller to use the expiry information present in the custom header of the origin server’s response, you must specify the custom header name in the **Custom cache-control header** field. Media Flow Controller now detects that this custom header contains the expiry information for the object and it proceeds to set the object expiry time to this value.

For example, consider that the origin server sends a custom header, “x-cache-control:max-age=3600.” For Media Flow Controller to detect that the custom header, “x-cache-control,” in the origin server’s response contains the expiry information of the object, you need to set the value of the **Custom cache-control header** field to “x-cache-control.” Media Flow Controller then serves the object from its cache for the next one hour from the time of caching.

- Select the **Cache Pinning** check box to activate the cache pinning options.
 - Select the **Enable Auto Pin** check box. The objects are automatically pinned to the cache.
 - **Pin Header Name**—Enter the header name for cache pinning.

Before you specify a value, it is necessary to know the header name that is sent from the origin server as a response to a request from the caching server. An object is pinned to the cache only when the header name that you configure in this field matches the response header from the origin server.

For example, if you configure the pin header name as *pinnable* in the service when the objects are retrieved from the origin servers, provided that the response headers from the origin servers contain the same term *pinnable*, then the objects are pinned to the cache.

- **Maximum Object Size**—Enter the maximum size of the object, in KB, that can be pinned to the cache. Any object beyond this size is not pinned. The value must be an integer between 0 and 4,294,967,295.
- **Maximum Cache Capacity**—Enter the maximum cache capacity, in GB, to be allocated for caching pinned objects for a specific service. After the capacity is

reached, the objects are no longer pinned to the cache. The value must be an integer between 0 and 4,294,967,295.

- **Use Validity Begin Header**—Enter a valid header whose timestamp is used to serve the request.

For example, consider that you set the value to the “VAL-BEGIN”. If the origin response consists of “VAL-BEGIN: Wed, 18 Dec 2010 04:58:08 GMT” header, then the object is pinned but served only after this time.

- **Maximum Cache Capacity (For All Objects)**—Select a cache tier (**SSD**, **SAS**, or **SATA**) and specify the maximum cache capacity, in MB, to be allocated for caching objects in the selected tier. It is recommended that you select the lowest tier for caching the objects. The value must be an integer between 0 and 31,457,280.
7. On the **Dynamic URI** tab, in the **Dynamic URI** area, configure parameters using which Media Flow Controller can identify dynamic URIs and ensure the delivery of the correct content. To accomplish this, an incoming request having a URI that matches the configured regex value (specified in the **URL to Match** field) is matched to a cache index string (specified in the **Map to** field).



NOTE: The URL to Match value must always be configured along with Map To. You can select Tunnel unmatched only if the URL to Match and Map To values are provided.

- **URL to Match**—Configure a regex expression to match the request URL. The <regex> has a maximum character limit of 1024 characters (including NULL); if the URI exceeds this limit, the request is tunneled. PCRE regex is not allowed, whereas GNU regex is allowed. The default value is NULL.

Example: /get/[^/]+/[^/]+/[^/]+/(.*)

- **Map to**—Configure a map-string value (a string to map or rewrite the URL portion of the request) when a match is found. The <map_string> has a maximum character limit of 2048 characters (including NULL). The default value is NULL.

Example: /\$1

- **Use Tunnel unmatched**—Select the check box to tunnel the request when no regex match is found.
8. On the **Origin Headers** tab, in the **Origin request** area, modify the headers in the requests sent from Media Flow Controller to origin servers.
- **Inherit Host Header**—If you select the check box, Media Flow Controller uses the incoming client request header in its request to the origin server.
 - **Set Host Header**—Upon a cache miss, while forwarding the client request to the origin server, Media Flow Controller by default replaces the host header value sent by the client to the origin server’s hostname or the IP address configured in the “Origin Server” area in the GUI. This action ensures that the client request is forwarded to the origin server for fetching the object. However, if you want to override this Media Flow Controller behavior and assign a static domain name or IP address

to the host header when the request is forwarded from Media Flow Controller to the origin server, you can set the static IP address or hostname in the **Set Host Header** field. Media Flow Controller uses the value provided in this field for setting the host header instead of using the origin server's IP address or hostname in its request to the origin server.

- **Add headers to origin request**—From this area, you can add custom headers to the requests forwarded by Media Flow Controller to the origin servers. You can add a maximum of four custom headers.
9. On the **Client Headers** tab, select the **Client response** check box, modify the headers in the responses sent from Media Flow Controller to clients.

An origin server's response to Media Flow Controller's cache-miss request includes headers as well as the requested object. At times, you might not want to forward the cached origin response headers to the client when serving a request. In such scenarios, you can specify what headers must be deleted and, if needed, what headers need to be added in Media Flow Controller responses to the clients. For example, you might want to remove the "Server" header that identifies the origin server from the Media Flow Controller response to the client.

On exceeding resource limits, add retry header to try after—Select this check box and configure the time period (in seconds) in which the clients may resend a request if Media Flow Controller is unable to process the current request because "resource-pool client session limit" is reached. If this parameter is configured, Media Flow Controller adds the "Retry-After:<N>" header in its "503 Service Unavailable" response to the client, so that the client can retry after N seconds.

The resource pool client session limit is derived from the resource pool configuration associated with the service. If the service is not bound to a specific resource pool, then the global resource pool configuration is used and the number of client requests handled by Media Flow Controller for that specific service is determined by this configuration.

Though new connections may be rejected because the resource pool session limits are reached, any existing connections for that service that are handled by Media Flow Controller are gracefully brought to completion.

You can enter a value from 0 through 86,400 seconds. The default value is 0 seconds—that is, the client can request the object immediately after it receives a 503 response from Media Flow Controller.

Example: Assume that a service is bound to a resource pool with a concurrent session limit of 10,000. When Media Flow Controller receives the ten thousand and first request, by default, it rejects this request with a "503 Service Unavailable" response. With this response, the client simply detects that the server is currently unable to handle the request due to a temporary overloading or maintenance of the server. However, when you configure Media Flow Controller to send a "Retry-After:5" header in its "503 Service Unavailable" response, it instructs the client to retry after five seconds.

To delete and add headers, perform the following steps:

1. From the **Delete headers from client response** area, click **Add** and enter the name of the header that you want to delete from the Media Flow Controller response to the client. In this case, add "Server."
2. From the **Add headers for client response** area, add another header to replace the server that was deleted. In this case, the header name could be "Server" and the value could be "Customized web server."

You can add and delete a maximum of eight headers. By default, the "via" header is deleted from the Media Flow Controller response to the client. However, if you want to retain it, select the header and click **Remove** from the **Delete headers from client response** area.

10. Click **OK** or **Cancel**. **OK** instantiates the values you set; **Cancel** closes the page without making any changes.



NOTE: See the *Juniper Networks Media Flow Controller Administrators Guide* for detailed information about configuring a service (namespace).

Related Documentation

- [Provisioning Services Overview](#)
- [Purging Content from Media Flow Controller Devices on page 14](#)
- [Actions on Services on page 15](#)
- [Creating Reverse Proxy Service XML Files for Export on page 41](#)
- [Creating Reverse Proxy Service XML Files for Import on page 42](#)
- [HTTP Reverse Proxy Services Overview on page 8](#)
- [Quick Reference to Tasks in Media Flow Activate](#)

Creating Reverse Proxy Service XML Files for Export

Reverse Proxy Service XML Schema

You can export a created service as an XML file. You can use your preferred XML editor to make the necessary changes and later import the XML file back to Media Flow Activate. You can then reprovision the service on Media Flow Controller with the updated configuration. However, if you want to import this configuration on another MFA server, you have to remove the service ID information before the import.



CAUTION: Server map information cannot be imported on a different Media Flow Activate server because it contains the device ID information that is specific to the Media Flow Activate server.

After you have customized and saved a Reverse Proxy XML configuration file, go to the Service Design > HTTP Reverse Proxy Services page, and select the service, which you

had recently customized. From the **Actions** list, select **Export Service**. In the **Export Service** dialog box, navigate to the XML file you created and click **Yes**. Save the file to a suitable location. Because the filename is saved with the service ID rather than the service name, rename the file with a suitable name and save it with an xml extension for your later use.



NOTE: See the *Juniper Networks Media Flow Controller Administrator's Guide* and *Juniper Networks Media Flow Controller CLI Command Reference* for Media Flow Controller CLI command descriptions.

See these topics for additional information:

- [Creating HTTP Reverse Proxy Services on page 30](#)
- [Creating Reverse Proxy Service XML Files for Import on page 42](#)
- [HTTP Reverse Proxy Services Overview on page 8](#)
- [Service Design Overview on page 4](#)
- [Media Flow Activate Overview](#)
- [Quick Reference to Tasks in Media Flow Activate](#)

Creating Reverse Proxy Service XML Files for Import

Use the following XML structure to create reverse proxy forms that can be imported to Media Flow Activate. These forms are then available for inclusion in a website service that is provisioned to selected Media Flow Controllers.

After you have customized and saved a Reverse Proxy XML configuration file, go to the Service Design > HTTP Reverse Proxy Service page. On the **Actions** list, select **Import Service**. From the **Import Service** dialog box, navigate to the XML file that you created and click **Ok**. The Reverse Proxy configuration provided in the XML file is made available to Media Flow Activate and can be included in service designs for provisioning to selected Media Flow Controllers through the **Service Provisioning** workspace.

Reverse Proxy Service XML Schema

When you customize the following XML file structure for your needs, give the **rproxyDefinition** element a **name** and **description**, then modify the options and elements. These elements are described in [Table 5 on page 44](#). See the following simple XML schema for an HTTP reverse proxy service. For a complex XML schema, see *Sample XML Schema for Network Optimization and Reverse Proxy Services*.

```
Simple XML Schema for Reverse Proxy Service
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<rproxyDefinitionCatalog>
  <rproxyDefinitions>
    <rproxyDefinition>
      <name>site1</name>
      <description>Test RProxy</description>
      <properties>
        <domain>hotmail</domain>
```

```

<isDomainRegex>false</isDomainRegex>
<match>
  <queryStringMatch>
    <name>name</name>
    <value>value</value>
  </queryStringMatch>
</match>
<precedence>1</precedence>
<clientRequest>
  <cacheIndex>
    <urlToMatch>/google/cbsgaruer374/pics</urlToMatch>
    <mapTo>/google/pics</mapTo>
    <tunnelUnMatched>>false</tunnelUnMatched>
  </cacheIndex>
</clientRequest>
<originServer>
  <fqdn>
    <address>
      <IPAddress>10.21.23.45</IPAddress>
      <port>5078</port>
    </address>
  </fqdn>
  <useSecureMode>>true</useSecureMode>
</originServer>
<cachePinning>
  <enableAutoPin>>false</enableAutoPin>
  <pinHeaderName>TestPinHeaderName</pinHeaderName>
  <maxObjectSizeInKB>500</maxObjectSizeInKB>
  <maxCacheCapacityInGB>1500</maxCacheCapacityInGB>
  <validityBeginHdr>valheader22</validityBeginHdr>
</cachePinning>
<maximumCacheCapacity>
  <cacheTier>SAS</cacheTier>
  <sizeInMB>234</sizeInMB>
</maximumCacheCapacity>
<clientResponse>
  <headersToDelete>
    <value>h3</value>
    <value>h4</value>
  </headersToDelete>
  <headersToAdd>
    <nameValue>
      <name>h1</name>
      <value>v1</value>
    </nameValue>
    <nameValue>
      <name>h2</name>
      <value>v2</value>
    </nameValue>
  </headersToAdd>
  <dscp>44</dscp>
</clientResponse>
<originRequest>
  <inheritHostHeaderValue>>true</inheritHostHeaderValue>
  <headersToAdd>
    <nameValue>

```

```

        <name>h12</name>
        <value>v12</value>
      </nameValue>
      <nameValue>
        <name>h22</name>
        <value>v22</value>
      </nameValue>
    </headersToAdd>
    <setHostHeader>headerrVal</setHostHeader>
  </originRequest>
</properties>
<tunnelAll>true</tunnelAll>
</rproxyDefinition>
</rproxyDefinitions>
</rproxyDefinitionCatalog>

```

Table 5: rproxyDefinition Options and Elements

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
name	none	none	Enter a name for the rproxyDefinition element.	namespace <i><name></i>
domain	none	none	Enter the name of the domain against which incoming requests are referenced; the media files are located in this domain.	namespace <i><name></i> domain regex <i><regex></i>

Table 5: rproxyDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
match	regexPath	none	Specify a path to be matched in the incoming request URL.	namespace <name> match uri (regex <regex> <uri-prefix>) [precedence number]
	queryStringMatch	name	Specify a query-string parameter (name-value pair) to be matched in the incoming request URL.	namespace <name> match query-string (regex <regex> <name> (any <value>)) [precedence number]
		value		
	regexQueryStringMatch	none		
	headerMatch	name	Specify a header to be matched in the incoming request.	namespace <name> match header (regex <regex> <name> (any <value>)) [precedence number]
		value		
	regexHeaderMatch	none		
	path	allFiles	Specify that all files in the given domain may be requested. Alternatively, use the specific files option.	namespace <name> match uri regex <regex>
		specificFiles	Specify a regex path to the files that are requested. Alternatively, use the all files option.	namespace <name> match uri <uri-prefix> /
precedence	none	none	The lower the value, the higher the precedence for this namespace, should there be a conflict. See “Understanding Precedence” in “ Service Design Overview ” on page 4 on page 70 for more information about configuring the “precedence” parameter.	namespace <name> match uri (regex <regex> <uri-prefix>) [precedence number] or namespace <name> match query-string (regex <regex> <name> (any <value>)) [precedence number] or namespace <name> match header (regex <regex> <name> (any <value>)) [precedence number]

Table 5: rproxyDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
clientRequest	cacheIndex	urlToMatch	Configure a regex expression to match the request URL. The regex value for url to match has a maximum character limit of 1024 characters (including NULL); if the URI exceeds this limit, the request is tunneled. Only one url to match expression per service is allowed. No PCRE regular expression is allowed; only GNU regular expression is allowed.	namespace <name> delivery protocol http client-request cache-index url-match <regex> map-to <map_string> no-match-tunnel
		mapTo	Configure a string to map or rewrite the URL portion of the request when a match is found. The mapTo value has a maximum character limit of 2048 characters (including NULL).	
		tunnelUnMatched	Tunnel the request when no regex match for url to match is found.	

Table 5: rproxyDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
originServer	address	IPAddress	Specify the origin server IP address that Media Flow Controller must access to fetch content upon a cache miss.	namespace <name> origin-server http <hostname/IP address>/<port>
		port	Specify the port through which the server listens for any requests.	
	originMap		Specify the name of the origin map to be used with this reverse proxy service configuration.	namespace <name> origin-server http server-map <map_name>
	useSecureMode	none	Authenticate the origin servers before downloading content from the origin servers.	namespace <name> delivery protocol http origin-request secure

Table 5: rproxyDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
cachePinning	enableAutoPin	none	Specify whether the objects should be automatically pinned to cache.	namespace <name> pinned-object auto-pin
	pinHeaderName	none	Specify the header name for pinning to cache.	namespace <name> pinned-object pin-header <name>
	maxObjectSizeInKB	none	Specify the maximum size of the object that can be pinned (in KB).	namespace <name> pinned-object max-obj-size KB
	maxCacheCapacityInGB	none	Specify the maximum size, in GB, that can be allocated for storing pinned objects for a namespace.	namespace <name> pinned-object cache-capacity GB
	validityBeginHdr	none	Specify whether to use the header timestamp to determine whether an object can be delivered when a request to the object has been received.	namespace <name> object validity-begin-header
maximumCacheCapacity	cacheTier	none	Specify a cache tier (SSD, SAS, or SATA) and specify the maximum cache capacity, in MB, to be allocated for caching objects in the selected tier. It is recommended that you select the lowest tier for caching the objects.	namespace <name> media-cache disk cache-tier [sas (free-block-threshold <number> group-read (enable disable) read-size <number>) sata (free-block-threshold <number> group-read (enable disable) read-size <number>) ssd (free-block-threshold <number> group-read (enable disable) read-size <number>)]
	sizeInMB	none		

Table 5: rproxyDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
clientResponse	headersToDelete	value	Enter the name of the header that should be deleted from the outgoing response—that is, from Media Flow Controller to the client.	namespace <name> delivery protocol http client-response header <name> [<value>] action delete
	headersToAdd	name and value	Enter the name of the header that should be added to the outgoing response—that is, from Media Flow Controller to the client.	namespace <name> delivery protocol http client-response header <name> [<value>] action add
	dscp	none	Media Flow Controller sets the DSCP field of the IPv4 packets by using the value specified for all the HTTP responses sent from Media Flow Controller to the client for this service.	namespace <name> delivery protocol http client-response dscp <number>

Table 5: rproxyDefinition Options and Elements (*continued*)

Option	Option or Sub-Element	Option or Sub-Element	Description	CLI Equivalent
originRequest	inheritHostHeaderValue	none	Allow Media Flow Controller to set the HOST: header in the Media Flow Controller-to-origin HTTP REQUEST to the value found in the HOST: header in the incoming URL.	namespace <name> delivery protocol http origin-request host-header inherit incoming-req (deny permit)
	headersToAdd	name and value	Enter the name of the header that should be added to the incoming request.	namespace <name> delivery protocol http origin-request header <name> [<value>] action add
	setHostHeader	none	Media Flow Controller uses the value provided for setting the host header instead of using the origin server's IP address or hostname in its request to the origin server.	namespace <name> delivery protocol http origin-request host-header set <header-value>
tunnelAll	none	none	Select to tunnel all incoming client requests directly to the origin server without further processing.	namespace <name> delivery protocol http client-request tunnel-all



NOTE: See the *Juniper Networks Media Flow Controller Administrator's Guide* and *Juniper Networks Media Flow Controller CLI Command Reference* for Media Flow Controller CLI command descriptions.

See these topics for additional information:

- [Sample XML Schema for Network Optimization and Reverse Proxy Services](#)
- [Provisioning Services Overview](#)
- [Actions on Services on page 15](#)
- [Creating HTTP Reverse Proxy Services on page 30](#)
- [Creating Reverse Proxy Service XML Files for Export on page 41](#)

- *Media Flow Activate Overview*
- *Quick Reference to Tasks in Media Flow Activate*

Creating Content Ingest Services

Use the **Service Design** workspace to create content delivery services. You create these services for the types of media you are delivering.

Before you begin configuring a **Content Ingest** media delivery service, you need the following information:

- Protocol
- Origin server
- Link depth
- Crawler schedule
- Content types

To configure content ingest services:

1. From the left navigation panel, click the plus sign (+) adjacent to **Service Design**.
2. Click the plus sign (+) adjacent to **Content Ingest Service**.
3. Click **Add Service Instance**. The **Add Service** page is displayed.
4. On the **Basic Properties** tab, specify the following information:
 - **Name**—Enter the name of the crawler instance.
 - **Description**—(Optional) Enter the description of the crawler instance.
 - **Protocol**—Enter the protocol to fetch content.
 - **Origin Server URL**—Enter the complete URL of the origin server including the domain name, port number, and path from where the content must be downloaded (for example, www.foo.com:8080/video/).

The origin for parent caches is the Media Staging Servers (MSSs) or third-party origin servers. The origin for edge caches is always the parent cache.

 - **Link Depth**—Enter an integer that specifies the level of directory or links that the crawler needs to follow to obtain the content. A selected link depth of 1 means that the crawler needs to follow the link or directory one level down.

You can enter a value from 0 through 10. The default value is 10.

 - In the Crawler Schedule area, specify the following information:
 - **Start Time**—Enter time in GMT at which the crawler instance should be started. The crawler begins crawling at the specified start time.
 - **Stop Time**—Enter time in GMT after which the new crawler instances are not started. If a crawling operation is in progress after the stop time is reached, the crawling operation is gracefully completed.

- **Refresh Interval**—Enter interval at which the crawler refreshes content from the origin server. Specify the time in minutes. If a stop time is not specified, the crawler automatically continues to crawl at the specified refresh intervals.
5. On the **Content Types** tab, you can set the crawl file types that the crawler must accept for the particular instance. The maximum number of extensions that can be added is 10. The default is to download or preload objects that match the extension.
 - Click **Add** to add the content types. The **Select File Extensions** dialog box is displayed.
 - In the **Available** pane of the dialog box, double-click the content types you want the crawler to accept. Each double-clicked content type moves to the **Selected** pane. Click **OK**. You are returned to the **Content Types** tab. You can view the selected file extensions in the **Accept Content Types** area.



NOTE: You can also add new content types. To add new content types, enter a content type in the **Add File Type** field and click **Add**.

- By default, **Preload** is set to **Yes**. If you do not want to preload the content, set this value to **No**.
- You can sort the data by clicking the arrow next to the column name.
- Click **Remove** to remove a selected content type.
- **Generate ASX file for every wmv file**—By default, this check box is selected, which means that whenever the crawler finds a .wmv file, a corresponding .asx file is auto-generated. Clear this check box if you do not intend to cache any .wmv files. All the applications that need to be delivered via MS-WMSP need the .asx file. The .asx file specifies the application domain and path pointing to the target video.



NOTE: The ASX file contains data about the actual media file (ASF file), containing video, audio, and so on. The purpose of an ASX file is to start the ASF file streaming.

6. Click **OK**, **Cancel**, or **Reset**. **OK** instantiates the values you set, **Cancel** closes the dialog box, and **Reset** returns all values to their defaults.

**Related
Documentation**

- [Content Ingest Services Overview on page 8](#)
- [Actions on Services on page 15](#)

PART 4

Index

- [Index on page 55](#)

Index

Symbols

#, comments in configuration statements.....	xi
(), in syntax descriptions.....	xi
< >, in syntax descriptions.....	x
[], in configuration statements.....	xi
{ }, in configuration statements.....	xi
(pipe), in syntax descriptions.....	xi

A

action	
service.....	15

B

braces, in configuration statements.....	xi
brackets	
angle, in syntax descriptions.....	x
square, in configuration statements.....	xi

C

comments, in configuration statements.....	xi
configuring	
Content Ingest services.....	51
HTTP Reverse Proxy service.....	30
XML file for export.....	41
XML file for import.....	42
Network Optimization service.....	19
XML file for import.....	26
content	
purging	14
Content Ingest services	
configuring.....	51
conventions	
text and syntax.....	x
curly braces, in configuration statements.....	xi
customer support.....	xi
contacting JTAC.....	xi

D

designing	
HTTP Reverse Proxy service.....	8
Network Optimization service.....	7

documentation	
comments on.....	xi
dynamic URI remapping	
understanding.....	4

E

exporting	
reverse proxy service XML file.....	41

F

font conventions.....	x
-----------------------	---

H

HTTP Reverse Proxy service	
about	8
configuring.....	30

I

importing	
Network Optimization service XML file.....	26
reverse proxy service XML file.....	42

M

managing	
provisioned sites.....	11
manuals	
comments on.....	xi
Media Flow Activate	
about HTTP Reverse Proxy service.....	8
about Network Optimization service.....	7
configuring	
HTTP Reverse Proxy service.....	30
Network Optimization service.....	19
configuring Content Ingest services.....	51
managing	
provisioned device.....	11
Network Optimization service XML file for	
import.....	26
purging content.....	14
reverse proxy service XML file for	
export.....	41
import.....	42
service actions.....	15
Service Design.....	4

N

Network Optimization service	
about	7
configuring.....	19

P

parentheses, in syntax descriptions.....	xi
Precedence	
using in Service Design.....	4
purging	
content.....	14

S

service	
actions	15
Service Design	
managing provisioned device.....	11
overview.....	4
understanding dynamic URI remapping.....	4
using Precedence.....	4
support, technical See technical support	
syntax conventions.....	x

T

technical support	
contacting JTAC.....	xi