APPENDIX E

Creating Service Rule Files

This appendix describes the Service Rule file used by a delivery service to specify the service rules for all the SEs in a delivery service. This appendix consists of the following topics:

- Introduction, page E-1
- Service Rule File Structure and Syntax, page E-4
- Rule Actions for Web Engine, page E-12
- Rule Actions for Flash Media Streaming, page E-26
- URL Signing Key in the Service Rule File, page E-18
- Service Rule File Example, page E-31

Note The Service Rule file is only supported for the Web Engine and Flash Media Streaming. Windows Media Streaming and Movie Streamer should continue to configure service rules by device. For more information, see the "Configuring Service Rules" section on page 4-21. For the Web Engine and Flash Media Streaming, the Service Rule file must be used if service rules are to be configured.

The Authorization Service must be enabled on all SEs participating in a delivery service that uses the Service Rule Configuration. The Authorization Service is enabled by default. For more information, see the "Configuring the Authorization Service" section on page 4-28.

When Geo/IP and service rules are configured by way of XML configuration files that are associated with a delivery service, each client request goes through the following processing order:

1. SE bypass (this is used for multi-tiered SEs), no configuration is required
2. Service rules
3. Geo/IP Network element
4. Geo/IP Geo element

For information about Geo/IP, see Appendix D, "Creating Geo/IP Files."

Introduction

The Service Rule file is an XML file used to specify the service rules for all the SEs in a delivery service. Just the same as configuring service rules for each SE, the Service Rule file allows you to specify a set of rules, each clearly identified by an action and a pattern, for all the SEs in a delivery service. Subsequently, for every incoming request, if a pattern for a rule matches the given request, the corresponding action for that rule is taken. You do not need to enable Service Rules on each SE for the Web Engine and Flash Media Streaming, just create a Service Rule file, upload it to the VDS, and assign it to the delivery service.

Note In a Service Rule File, you can define multiple PatternListGrps.

It is not recommended that you use a single Service Rule File for several Delivery Services, since a single request belongs to a specified delivery service. It will cost more CPU circles while going though all the PatternListGrps and all the other rules defined later, that is not applied to the delivery service.

Converting Old Service Rules to New Service Rules

The following example shows the commands for configuring a service rule that performs a URL rewrite using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action rewrite pattern-list 1
SE (config)# rule pattern-list 1 url-regsub http://.*.rfqdn2.cds.cisco.com/(.*)
http://$1
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
     <Revision>1.0</Revision>
     <CustomerName>MOD</CustomerName>
     <Rule_Patterns>
          <PatternListGrp id = "grp1">
               <UrlRegex>.rfqdn2.cds.cisco.com/</UrlRegex>
          </PatternListGrp>
     </Rule_Patterns>
     <Rule_Actions>
        <Rule_Allow matchGroup = "grp1" protocol = "http" />
        <Rule_UrlRewrite matchGroup = "grp1" protocol = "http" regsub = "http://.*.rfqdn2.cds.cisco.com/(.*)"
rewrite-url = "http://$1" />
     </Rule_Actions>
</CDSRules>
```

Table E-1 shows the mapping between a service rule pattern command and an XML pattern.

Table E-1 Mapping Service Rule Patterns—CLI Format to XML Format

Pattern Type	CLI Pattern	XML Pattern
Domain	rule pattern-list 1 domain rfqdn.cds.com	<PatternListGrp id = "1"> <Domain>rfqdn.cds.com</Domain> </PatternListGrp>
SrcIp	rule pattern-list 1 src-ip 1.1.1.1 255.255.255.0	<PatternListGrp id = "1"> <SrcIp>1.1.1.1</SrcIp> </PatternListGrp>
UrlRegex	rule pattern-list 2 url-regex http:\/\/.*.svc01.cdn.t-online.de\/web[0-9]+\/streaming\/CDN_testprovider_2\/streaming\/.*	<PatternListGrp id = "2"> <UrlRegex> http:\/\/.*.svc01.cdn.t- online.de\/web[0-9]+\/streaming\/CDN_testprovider_2\/streaming\/.* </UrlRegex> </PatternListGrp>

The pattern type header-field is not supported in the Service Rule file.

Table E-2 shows the mapping between a service rule action command and an XML action.

Table E-2 Mapping Service Rule Actions—CLI Format to XML Format

Action Type	CLI Action	XML Action
Allow	rule action allow pattern-list 1 protocol http	<Rule_Allow matchGroup = "1" protocol = "http" />
Block	rule action block pattern-list 2 protocol http	<Rule_Block matchGroup = "2" protocol = "http" />
Validate	rule action validate-url-signature error-redirect-url "http://wwwin.cisco.com" pattern-list 1 protocol http	<Rule_Validate matchGroup = "1" protocol = "http" error-redirect-url = "http://wwwin.cisco.com" exclude-validation = "all" />

UrlRewrite	rule pattern-list 3 url-regsub http://(.*.)cdsis.com/(.*.)mp4(.*.) http://customer.com/%29%28(.*.) http://$1$2$3mp4 rule action rewrite pattern-list 3 protocol http	<Rule_UrlRewrite matchGroup = "3"
		protocol = "http"
		regsub = "http://(.*.)cdsis.com/(.*.)mp4(.*.) http://customer.com/%29%28(.*.)"
		rewrite-url = " http://$1$2$3mp4" />

Adding a Service Rule File to the VDS

The Service Rule files can be created using any ASCII text-editing tool. The Service Rule file are registered to the VDS by using the Authorization File Registration page. For more information see the "Authorization File Registration" section on page 6-15. When the file has been registered, you can assign it to a delivery service through the Authorization Plugins page. For more information, see the "Authorization Plugins" section on page 5-27.

Service Rule File Structure and Syntax

The XML Schema file describes and dictates the content of the XML file. The CDSRules.xsd file contains the XML schema. To view or download a copy of the CDSRules.xsd file, see the "Viewing or Downloading XML Schema Files" section on page 6-24.

Table E-3 defines the Service Rule file elements. For more information on the rule actions supported by Web Engine, see the "Rule Actions for Web Engine" section on page E-12. For more information on the rule actions supported by Flash Media Streaming, see the "Rule Actions for Flash Media Streaming" section on page E-26.

Table E-3 Service Rule File Elements

Element	Subelements	Attributes	Description
CDSRule	Revision		Optional. Revision number to specify the version of this file.
	CustomerName		Optional. Customer name associated with this file.
	ApplyAllTier		Required for the Rule_UrlResolve rule action.
	Rule_Patterns		Patterns to match for a specified action. There can be only one Rule_Patterns element for a Service Rule file.
	Rule_Actions		Action to take when a pattern is matched. There can be only one Rule_Actions element for a Service Rule file.
ApplyAllTier			Valid values for the ApplyAllTier element are "yes" or "no." The ApplyAllTier element has the following effect: • If the ApplyAllTier is set to "yes," the Rule_UrlResolve rule action is applied to all SEs in the delivery service. The ApplyAllTier element must be set to "yes" for the Rule_UrlResolve to work properly. For more information, see the "URL Resolve" section on page E-12. • If the ApplyAllTier is set to "no" or if it is absent, and Rule_UrlResolve is included in the Service Rule file, the Rule_UrlResolve does not work properly. • If the ApplyAllTier is set to "no" or if it is absent, and Rule_UrlResolve is not included, the Service Rule file is only applied to the edge tier. • If the Service Rule file needs to be applied to the Content Acquirer (root SE), then ApplyAllTier must be set to "yes."
Rule_Patterns	PatternListGrp		Marks the beginning and ending of all the defined patterns in this file.
PatternListGrp	Domain SrcIp UrlRegex	id	The PatternListGrp *id* attribute is used to identify the pattern list group and can be up to 128 alphanumeric characters. **Note** Currently, the Header element is not supported.

Table E-3 Service Rule File Elements (continued)

Element	Subelements	Attributes	Description
Domain			The Domain element is used to match the domain name in the URL or the host header against a regular expression. For more information, see the Table 4-12 on page 4-23. **Note** When VOD (prefetch/caching) and live streaming share the same content origin, and the Service Rules XML file is configured to validate the signed URL where the domain must match the Service Routing Domain Name, make sure to create rule patterns for the URL validation to match both the Service Routing Domain Name and the Origin Server FQDN. Additionally, when the URL is signed, exclude the domain from the signature. See the "Running a Python URL Signing Script" section on page H-11 for more information. The URL validation must not include the domain for validation (use the **exclude-domain** option for the *exclude-validate* attribute of the Rule_Validate element).
SrcIp			Matches the source IP address of the request. The SrcIP pattern requires the IP address be specified in the classless inter-domain routing (CIDR) format. The Service Rule XML file validation fails if the IP address is not in CIDR format.
UrlRegex			Matches the URL against a regular expression. The match is case sensitive. The following example covers both uppercase and lowercase expressions of MP4 files: <UrlRegex> http://.*.cdsis.com/.*.[mM][pP]4 </UrlRegex> **Note** The VDS-IS system uses GNU regular expressions.
Rule_Actions	Rule_Allow Rule_Block Rule_Validate Rule_UrlRewrite Rule_NoCache Rule_UrlRedirect Rule_UrlResolve Rule_UrlGenerateSign Rule_ForceRevalidate Rule_SwfFileValidate Rule_Dscp Rule_SetAction		For information about the rule action processing, see the "Rule Action Processing" section on page E-11.

Table E-3 Service Rule File Elements (continued)

Element	Subelements	Attributes	Description
Rule_Allow		matchGroup protocol	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be one or more of the following: http, rtmp, rtmpe, rtmpt, and rtmpte.
Rule_Block		matchGroup protocol	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be one or more of the following: http, rtmp, rtmpe, rtmpt, and rtmpte.
Rule_Validate		matchGroup protocol error-redirect-url exclude-validation key public-key symmetric-key	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be one or more of the following: http, rtmp, rtmpe, rtmpt, and rtmpte. The *error-redirect-url* attribute value is the URL that clients are redirected to if they fail validation. The *exclude-validation* attribute is optional and can be one of the following values: client-ip, expiry-time, exclude-domain, or all. The *exclude-validation client-ip* attribute instructs the SEs to ignore the client's IP address when processing the validation of the signed URL. The *exclude-validation expiry-time* attribute instructs the SEs to ignore the expiry time that normally limits access to the content when the expiry time has occurred. The *exclude-validation exclude-domain* attribute instructs the SEs to ignore the domain in the URL when processing the validation of the signed URL. The *exclude-validation all* attribute instructs the SEs to ignore both the client IP address and the content expiration time when processing the validation of the signed URL. The key, public-key, and symmetric-key attributes are described in the "URL Signing Key in the Service Rule File" section on page E-18.

Table E-3 Service Rule File Elements (continued)

Element	Subelements	Attributes	Description
Rule_UrlRewrite		matchGroup protocol rewrite-url regsub	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be http. **Note** Only http is supported as the *protocol* attribute value All other values have no affect. The *rewrite-url* attribute value is the URL used to rewrite the original request. The *regsub* attribute value is the regular expression the request URL must match to be replaced with the *rewrite-Url* attribute value. The regsub attribute value must be an exact match of the string you want to replace in the request URL. **Note** The regsub attribute supports regular expressions, but only one substitution is supported per Rule_UrlRewrite. Multiple substitutions for the same Rule_UrlRewrite are not supported.
Rule_NoCache		matchGroup protocol	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be http. **Note** Only http is supported as the *protocol* attribute value All other values have no affect.
Rule_UrlRedirect		matchGroup protocol redirect-url	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be http. The *redirect-url* attribute value is the URL to redirect the request to. **Note** Only http is supported as the *protocol* attribute value All other values have no affect.

Table E-3 Service Rule File Elements (continued)

Element	Subelements	Attributes	Description
Rule_UrlResolve		matchGroup protocol	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be http. **Note** Only http is supported as the protocol attribute value. All other values have no affect.
	SourceUrl (required)	regsub rewrite-url	The *regsub* attribute value is the regular expression the request URL must match to be replaced with the *rewrite-Url* attribute value. The regsub attribute value must be an exact match of the string you want to replace in the request URL. **Note** The regsub attribute supports regular expressions, but only one substitution can be defined. Multiple substitutions are not supported. The *rewrite-url* attribute value is the URL used to rewrite the original request.
	StorageUrl (required)	regsub rewrite-url	The *regsub* attribute value is the regular expression the request URL must match to be replaced with the *rewrite-Url* attribute value. The regsub attribute value must be an exact match of the string you want to replace in the request URL. **Note** The regsub attribute supports regular expressions, but only one substitution can be defined. Multiple substitutions are not supported. The *rewrite-url* attribute value is the URL used to rewrite the original request.

Table E-3 Service Rule File Elements (continued)

Element	Subelements	Attributes	Description
Rule_UrlGenerateSign		matchGroup protocol key-id-owner key-id-number timeout-in-sec key private-key symmetric-key	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be http. The *key-id*-owner attribute value is the ID number for the owner of the encryption key. Valid entry is 1 if the key is defined in the Service Rule XML file. Valid entries are from 1 to 32 if the key is defined in the URL Signing page or by using the **url-signature** command. The *key-id*-number attribute value is the encryption key ID number. Valid entry is 1 if the key is defined in the Service Rule XML file. Valid entries are from 1 to 16 if the key is defined in the URL Signing page or by using the **url-signature** command. The *timeout-in-sec* attribute value is the time interval to wait before expiring the signed URL. The default is 30 seconds. **Note** Only http is supported as the *protocol* attribute value All other values have no affect. The key, private-key, and symmetric-key attributes are described in the "URL Signing Key in the Service Rule File" section on page E-18.
Rule_ForceRevalidate		matchGroup protocol	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be http. **Note** Only http is supported as the *protocol* attribute value All other values have no affect.
Rule_SwfFileValidate		matchGroup protocol	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be one or more of the following: rtmp, rtmpe, rtmpt, and rtmpte.
Rule_Dscp		matchGroup protocol dscp-bits	The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be one or more of the following: rtmp, rtmpe, rtmpt, and rtmpte. The *dscp-bits* attribute value ranges from 0 to 63. Absence of the tag in the rules xml file shall assume default DSCP value to 0.
Rule_SetAction	SetParameter SetRewrite SetExecute		The Rule_SetAction is used for Session-Based Encryption and Session Tracking. For more information, see Appendix F, "ABR Session-Based Encryption and Session Tracking."

All specified attributes for the Rule_Actions subelements are required, except the exclude-validation attribute, which is optional.

Pattern Matching

Before any pattern matches are checked, the protocol is checked. If the protocol of the incoming request does not match the protocols specified for the rule action, the action is not taken. If a pattern for a rule matches the given request, the corresponding action for that rule is taken.

Boolean AND Function

When a PatternListGrp is specified for an action, it implies an AND of all the patterns within the group. All patterns specified in that group must be matched for the action to take place. In the following example, both patterns in grp1 must be matched for the action to be taken.

```
<Rule_Patterns>
                <PatternListGrp id = "grp1">
                    <Domain>fmsvod.com</Domain>
                     <uRLregex>clouds</UrlRegex>
                </PatternListGrp>
        </Rule_Patterns>
```

Boolean OR Function

When the matchGroup id attributes are separated by a comma, it implies an OR of all the patterns. The action is taken when either of the patternListGrp elements are matched. In the following example, the pattern of either grp1 or grp2 is considered a match.

```
<Rule_Patterns>
                <PatternListGrp id = "grp1">
                    <Domain>fmsvod.com</Domain>
                </PatternListGrp>
        </Rule_Patterns>
<Rule_Patterns>
                <PatternListGrp id = "grp2">
                     <uRLregex>clouds</UrlRegex>
                </PatternListGrp>
        </Rule_Patterns>

 <Rule_Actions>
                   <Rule_Block matchGroup = "grp1,grp2" protocol = "rtmp"  />
        </Rule_Actions>
```

In the following example, multiple protocols are specified for the same rule by including the protocols separated by a comma as a value of the protocol attribute:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Capricious</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <Domain>fmsvod.com</Domain>
                </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                <Rule_Validate matchGroup = "grp1" protocol = "rtmpe,rtmpte"
                 error-redirect-url="http://www.cisco.com"/>
        </Rule_Actions>
    </CDSRules>
```

Rule Action Processing

The rules are processed in the same order they are listed in the Rule_Actions element.

Multiple Rule_Actions can be configured; for example, there can be a Rule_Allow followed by a Rule_Block followed by a Rule_UrlRewrite and so on. The Rule_Actions can be in any order and the processing of the rules is determined by the order they are listed in the Service Rule XML file.

The maximum number of rule actions allows is 100. If the number of rule actions exceeds 100, then the Service Rule XML file validation fails.

Note For RTSP, rules processing uses the Rule daemon and not the Authsvr process; therefore, the authsvr statistics (**show statistics authsvr delivery-service-id** <*delivery service ID*> **rules**) are not incremented. For HTTP, if ApplyAllTier is set to "no," statistics are incremented only on the edge SE, not the Content Acquirer (root SE).

Only the following rule actions are allowed to have multiple entries:

- Rule_Rewrite
- Rule_UrlResolve
- Rule_UrlGenerateSign

All other rule actions can only have a single entry. If there are multiple entries of the same Rule_Actions subelement, the last entry with a matched condition is the rule that is applied.

The following list describes the Rule_Actions processing:

- When a Rule_Allow pattern is matched, the request is allowed, and if there are subsequent rules, the next Rule_Actions is processed. If the condition is not matched, the request is denied and no further rule processing is performed.
- When a Rule_Block pattern is matched, the request is blocked and the Rule_Actions processing does not continue.
- When a Rule_Validate pattern is matched, the request is validated and if the validation is successful, the Rule_Actions processing continues to the next rule configured. If the validation fails, the request is not validated and the Rule_Actions processing stops. For more information about rule processing for Rule_Validate, see the "Service Rule Action Order for Rule_Validate and Rule_UrlGenerateSign" section on page E-22.
- Whether a Rule_UrlRewrite pattern is matched or not, rule processing continues to the next configured rule. If the Rule_UrlRewrite pattern is matched, the request is rewritten. If the Rule_UrlRewrite pattern is not matched, the request is not rewritten.
- Whether a Rule_NoCache pattern is matched or not, rule processing continues to the next configured rule. Rule_NoCache action just determines whether to cache the content on the SE or not, provided further rule processing results in the request being allowed. If the Rule_NoCache pattern is matched, the content is not cached on the SE. If the Rule_NoCache pattern is not matched, the content is cached on the SE.
- Whether a Rule_UrlRedirect pattern is matched or not, rule processing continues to the next configured rule. If the Rule_UrlRedirect pattern is matched, the request is redirected. If the Rule_UrlRedirect pattern is not matched, the request is not redirected.
- Whether a Rule_UrlResolve pattern is matched or not, rule processing continues to the next configured rule. Rule_UrlResolve action maps the incoming URL to a Source and Storage URL Source URL. If the Rule_UrlResolve pattern is not matched, the mapping does not occur.

- When a Rule_UrlGenerateSign pattern is matched, a generated URL signature is returned to the client as part of the ASX response for Windows Media Streaming live programs, and processing continues to the next configured rule. For more information about the rule process for the Rule_UrlGenerateSign rule action, see the "Windows Media Streaming ASX Files with URL Signing" section on page E-20.
- Whether a Rule_ForceReValidate pattern is matched or not, rule processing continues to the next configured rule. Rule_ForceReValidate action enables the Web Engine to take the appropriate revalidation action. If the Rule_ForceReValidate pattern is not matched, the revalidation action is not taken.
- Whether a Rule_SwfFileValidate is matched or not, rule processing continues to the next configured rule. Rule_SwfFileValidate action enables Flash Media Streaming to perform SWF file validation. If the Rule_SwfFileValidate pattern is not matched, the SWF file is not validated.

Rule Actions for Web Engine

The service rule actions for allow, block, URL signature validation, URL rewrite, and no cache are described at the beginning of the Creating Service Rule Files appendix. This section provides details on the following rule actions:

- URL Resolve
- URL Redirect
- Force Revalidation
- URL Generate Signature

As well as information on converting Windows Media Streaming service rules for generate-url-signature and validate-url-signature ("Converting Old Windows Media Streaming Service Rules for URL Signing and Validation" section on page E-25.

Multiple Rule Actions in Web Engine

It is important to note that the Web Engine only applies one of the following rule actions, in the following order:

1. Rule_UrlRedirect
2. Rule_UrlResolve
3. Rule_UrlRewrite

If more than one rule action is returned from the Authorization Server, only the one with the higher priority is chosen.

URL Resolve

In many content delivery cases, URLs are not just used as unique identifiers of the content, but they are also used to transfer specialized information from the client to the Origin Servers (for example, client IP address es and special tags for video identification) in the form of query strings.

The URL Resolve rule action (Rule_UrlResolve) provides a way to take a client's incoming URL (known as the Intercept URL) and resolve it into other URLs that can be used for caching (known as the Storage URL) and ingesting the content (known as the Source URL).

Note The default behavior of the Web Engine is to cache the content when the request URL has a query string, which results in multiple copies of the same content being stored. The Rule_NoCache rule action in the Service Rule file offers a way to not cache content with query strings; however, this meant the content was served by way of bypass (downloaded from the Origin Server directly), which resulted in more connections to the Origin Server. With the Rule_UrlResolve rule action, the Storage URL provides a way to address any URL uniqueness that complicates caching, so long as the uniqueness can be removed by parsing the URL and replacing parts of the URL with regular expressions.

Table E-4 describes the URLs used in Rule_UrlResolve and the CDS-Domain header.

Note URL Resolve Rule does not work when ABR Session Tracking is enabled. For more information on HLS Session Tracking, see the Appendix F, "ABR Session-Based Encryption and Session Tracking."

Table E-4 Components of the URL Resolve

Component	Description
Intercept URL (required)	Incoming URL from the client or downstream proxy. This is the URL that the client uses to send a request for content. This URL has the domain name that matches the service routing fully-qualified domain name (RFQDN). The Service Router, or other device in the DNS plane, can redirect and resolve the request to a device that is part of the service routing domain and that serves the content. The Intercept URL is seen by the SE in the following form: `http://SE-HOST-NAME.se.Service_Routing_Domain_Name/Content_Path`
Storage URL (required)	Translated URL used by the Web Engine for storage-related operations. This is the URL used to store and locate the content on the SEs. Typically, this is the same as the Source URL. However, this URL can be any configured regular expression. The Storage URL has the following form: `http://origin_server/Content_Path` **Note** Required subelement of the Rule_UrlResolve rule action. **Note** Cisco recommends that the domain should be either the Origin Server fully-qualified domain name (OFQDN) or the RFQDN of the delivery service. When the domain of the Storage URL is configured to be something other than the OFQDN of the delivery service to which the request belongs, dynamically cached content is not deleted from the SE when the SE is unassigned from the delivery service. Content deletion only happens through the eviction process or by using the **clear content** command.

Table E-4 Components of the URL Resolve (continued)

Component	Description
Source URL (required)	Translated URL used by the Web Engine to ingest content. This is the URL used to ingest content from the Origin server. Normally, the domain name of the incoming URL is replaced with the OFQDN. However, this URL can be any configured regular expression. The Source URL has the following format: `http://origin_server/Content_Path` **Note** Configuring the Source URL domain to be the RFQDN of a delivery service causes the request to be rejected, because the RFQDN of a delivery service most likely resolves to the SR, which could result in loops in the system. **Note** Required subelement of the Rule_UrlResolve rule action.
CDS-Domain	RFQDN of the delivery service to which the Intercept URL from the client belongs. The CDS-Domain header is sent from the downstream SE to the upstream SE. To ensure consistency in locating the SEs that are participating in the delivery service, the RFQDN is used in the URL sent from the edge SE to the middle-tiered SEs when a cache-miss occurs at the edge SE. When the middle-tiered SEs see the CDS-Domain header in the request, it replaces the domain in the “Intercept URL” (which is the edge SE’s Storage URL) with the RFQDN. **Note** The CDS-Domain header is always sent, whether URL Resolve is configured or not. If this header is received from an end client, the request is rejected.

Example 1 Example for One-Tiered VDS

This section provides an example of the Service Rule XML file with the Rule_UrlResolve rule action. The following parameters are used in the example:

- RFQDN—ott.c.awebsite.com
- OFQDN—cds.c.awebsite.com
- Origin Server—cache12.awebsite.com

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
<Revision>1.0</Revision>
<CustomerName>DMZ1</CustomerName>
    <Rule_Patterns>
        <PatternListGrp id = "grp1">
          <UrlRegex>ott\.c\.awebsite\.com/.*\?params=(.*)</UrlRegex>
       </PatternListGrp>
    </Rule_Patterns>
    <Rule_Actions>
       <Rule_UrlResolve matchGroup="grp1" protocol="http">
           <SourceUrl regsub="http://(.*)ott\.c\.awebsite\.com/(.*\?)(params=)(.*)"
                      rewrite-url="http://$1$2$3"/>
           <StorageUrl regsub="http://.*\.c\.awebsite\.com/.*(id=[0-9a-zA-Z]*)"
                     rewrite-url="http://cds.c.awebsite.com/$1"/>
       </Rule_UrlResolve>
    </Rule_Actions>
</CDSRules>
```

The bold portion shows the regular expressions used to translate the Intercept URL into the Storage URL and the Source URL. The URL Resolve process for this example is as follows:

1. The client URL request (incoming URL) might be as follows:

```
http://ott.c.awebsite.com/cache12.awebsite.com/xaa?params=sparams=id&&ip=1.2.3.4&id=ab
cd
```

2. After Service Router redirection, the URL request arrives at the edge SE in the Intercept URL form as follows:

```
http://se1.se.ott.c.awebsite.com/cache12.awebsite.com/xaa?params=sparams=id&ip=1.2.3.4
&id=abcd
```

3. After the Rule_UrlResolve action, the following Source URL and Storage URLs are created:

   Storage URL: `http://cds.c.awebsite.com/xaa?id=abcd`

   Source URL: `http://cache12.awebsite.com/xaa?sparams=id&ip=1.2.3.4&id=abcd`

The following rules apply for URL Resolve:

- Only http is supported as the protocol attribute value, and only for VOD (prefetched, dynamic, and hybrid content), live, and adaptive bit rate (ABR). MP3 is not supported.
- Client headers (such as cookies, accept, and so on) are not forwarded to the origin server.
- If the Source URL belongs to another delivery service, processing continues to use the original delivery service.

Example 2 Example for VDS with Two or More Tiers

The basic example in the Example 1 on page E-14 assumes a VDS with only one tier (root location). For systems with two tiers or more, there needs to be at least two Rule_UrlResolve rules per delivery service:

- One to translate the Intercept URL coming from the SR to the edge tier.

```
http://se1.se.ott.c.awebsite.com/cache12.c.awebsite.com/videoplayback?params=sparams=i
d&ip=1.2.3.4&id=abcd
```

- Another to translate the Intercept URL coming from the edge tier to the middle tiers, which in this case, is actually the Source URL from the edge tier.

```
http://cache12.c.awebsite.com/videoplayback?sparams=id&ip=1.2.3.4&id=abcd
```

It is clear that the Intercept URL coming into the middle tiers does not match pattern grp1 in Example 1 on page E-14. A second Rule_UrlResolve rule action is required. The pattern for grp2 in the following Service Rule file example matches the Intercept URL coming into the middle tiers and will be translated into the same Source URL and Storage URL as the edge tier:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
<Revision>1.0</Revision>
<CustomerName>DMZ1</CustomerName>
<ApplyAllTier>yes</ApplyAllTier>
    <Rule_Patterns>
        <PatternListGrp id = "grp1">
          <UrlRegex>ott\.c\.awebsite\.com/.*\?params=</UrlRegex>
       </PatternListGrp>

 <PatternListGrp id = "grp2">
          <UrlRegex> cache12.c.awebsite.com\/.*\?sparams=</UrlRegex>
 </PatternListGrp>
    </Rule_Patterns>
    <Rule_Actions>
       <Rule_UrlResolve matchGroup="grp1" protocol="http">
           <SourceUrl regsub="http://.*ott\.c\.awebsite\.com/(.*\?)params=(.*)"
                    rewrite-url="http://$1$2"/>
           <StorageUrl regsub="http://.*\.c\.awebsite\.com/(.*\?).*(id=[0-9a-zA-Z]*)"
                     rewrite-url="http://cds.c.awebsite.com/$1$2"/>
       </Rule_UrlResolve>

<Rule_UrlResolve matchGroup="grp2" protocol="http">
           <SourceUrl regsub="http://(.*)"
                      rewrite-url="http://$1"/>
           <StorageUrl regsub="http://.*\.c\.awebsite\.com/(.*\?).*(id=[0-9a-zA-Z]*)"
                     rewrite-url="http://cds.c.awebsite.com/$1$2"/>
</Rule_UrlResolve >
    </Rule_Actions>
</CDSRules>
```

Additionally, this Service Rule file must be applied to every tier in the VDS to create the correct Source URL and Storage URL at each tier. The ApplyAllTier is a new Service Rule element that ensures the Service Rule file is applied to all tiers of the delivery service.

Note The ApplyAllTier element must be set to yes for the Rule_UrlResolve to work properly.

URL Rewrite and URL Resolve

URL Rewrite and URL Resolve have the following differences:

- URL Resolve (Rule_UrlResolve) allows the configuration of separate Source and Storage URLs for a given incoming URL; URL Rewrite (Rule_UrlRewrite) allows the Intercept URL to be modified and the modified URL is used for both the Source URL and Storage URL.
- Rule processing is different. In the case of Rule_UrlRewrite, if the domain of the rewritten URL maps to a new delivery service, that delivery service is used to process the request. In the case of Rule_UrlResolve, even if the domain of the Source URL maps to another delivery service, the original delivery service is used to process the request.

Monitoring

Use the following commands to monitor the URL Resolve:

- **show statistics web-engine detail**
- **show cache content**
- **show cache-router routes web-engine URL**

The following new tokens have been added to the Web Engine custom log formats:

- %g—Storage URL
- %G—Source URL

The Web Engine Ingest log has a new field called CDS-Domain which has the CDS-Domain header being sent to the upstream SEs.

The **show statistics web-engine** command has a neVDSw counter, Authorization Resolve, which keeps track of the number of URL Resolve hits.

The web-engine-error-logs has a log entry of the Storage URL and Source URL. The log entry is identified by the WEUrl*is string.

URL Redirect

The URL Redirect (Rule_UrlRedirect) rule action is supported in the Service Rule XML file for the Web Engine. Following is an example of the Rule_UrlRedirect rule action:

```
<Rule_UrlRedirect matchGroup = "grp4" protocol = "http" redirect-url = "http://www.google.com" />
```

Whether a Rule_UrlRedirect pattern is matched or not, rule processing continues to the next configured rule. If the Rule_UrlRedirect pattern is matched, the request is redirected. If the Rule_UrlRedirect pattern is not matched, the request is not redirected.

The **show statistics web-engine** command has a new counter, Authorization Redirect, which keeps track of the number of URL Redirect hits.

Force Revalidation

The Force Revalidation (Rule_ForceReValidate) action rule forces revalidation of cached content. The freshness of content algorithm and the comparison between the Origin Server expiry time with the max age value are ignored if this rule action is invoked.

If the Rule_ForceReValidate rule action is configured as part of Service Rule file, the Authorization Server responds to the Web Engine with the Rule_ForceReValidate directive. This enables the Web Engine to take appropriate revalidation action.

Following is an example of the Service Rule file with the Rule_ForceReValidate rule action:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
<Revision>1.0</Revision>
<CustomerName>Cisco</CustomerName>
    <Rule_Patterns>
       <PatternListGrp id = "grp1">
          <Domain>demo.cdsis.com</Domain>
       </PatternListGrp>
    </Rule_Patterns>
    <Rule_Actions>
       <Rule_ForceReValidate matchGroup = "grp1" protocol = "http"  />
    </Rule_Actions>
</CDSRules>
```

Whether a Rule_ForceReValidate pattern is matched or not, rule processing continues to the next configured rule. Rule_ForceReValidate action enables the Web Engine to take the appropriate revalidation action. If the Rule_ForceReValidate pattern is not matched, the revalidation action is not taken.

The **show statistics web-engine** command has a new counter, Authorization Force Revalidate, which keeps track of the number of forced revalidation hits.

URL Generate Signature

The URL Generate Signature (Rule_UrlGenerateSign) rule action is supported in the Service Rule XML file for the Web Engine. The Rule_UrlGenerateSign is a rule action for generating the URL signatures in the Windows Media metafile (ASX file) response associated with prefetched content, based on the SE configuration for the URL signature and this rule action.

The Windows Media player receives the ASX file containing the signed URL, parses it, and sends out the request again with the signed URL. The SE receives the signed URL and performs the URL validation with the internally signed URL. If the validation is successful, the content is served to the client.

The Rule_UrlGenerateSign has the following attributes:

- matchGroup—Attribute value is the list of PatternListGrp *id* attributes
- protocol—Attribute value must be http
- key-id-owner—Attribute value is the ID number for the owner of the encryption key. Valid entry is 1 if the key is defined in the Service Rule XML file. Valid entries are from 1 to 32 if the key is defined in the URL Signing page or by using the **url-signature** command.
- key-id-number—Attribute value is the encryption key ID number. Valid entry is1 if the key is defined in the Service Rule XML file. Valid entries are from 1 to 16 if the key is defined in the URL Signing page or by using the **url-signature** command.
- timeout-in-sec—Attribute value is the time interval to wait before expiring the signed URL. The default is 30 seconds.
- key—Unique URL signature key that is up to 16 characters. For symmetric key URL validation.
- private-key—URL where the private key file is located. For asymmetric key URL validation.
- symmetric-key—Key (16 bytes) used for AES encryption of the signed URL. For asymmetric key URL validation.

Note Only http is supported as the *protocol* attribute value. All other values have no affect.

Following is an example of the Service Rule file with the Rule_UrlGenerateSign rule action and the URL signing key defined in the URL Signing page:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
   <Revision>1.0</Revision>
   <CustomerName>Cisco</CustomerName>
   <ApplyAllTier>yes</ApplyAllTier>
    <Rule_Patterns>
      <PatternListGrp id = "grp1">
         <Domain>cisco.co</Domain >
      </PatternListGrp>
   </Rule_Patterns>

   <Rule_Actions>
      <Rule_UrlGenerateSign matchGroup = "grp1" protocol = "http" key-id-owner="1"
key-id-number="2" timeout-in-sec="30"/>
   </Rule_Actions>
</CDSRules>
```

URL Signing Key in the Service Rule File

The Service Rule XML file supports URL signing configuration of symmetric and asymmetric keys. Additionally, URL signature validation is supported for all protocol engines, except Movie Streamer, and URL signature generation is supported for Windows Media Streaming live requests (.asx).

URL signing can still be configured for each SE by using the URL Signing page to specify the key parameters. If there are no key parameters specified in the Service Rule XML file, the SE settings are used. For more information on SE configuration, see the "Configuring URL Signing Key" section on page 4-27.

For information on converting Windows Media Streaming service rules for URL signature generation and validation with URL signing parameters, see the "Converting Old Windows Media Streaming Service Rules for URL Signing and Validation" section on page E-25.

The following new attributes have been added to the Rule_Validate element:

- key—Unique URL signature key that is up to 16 characters. For symmetric key URL validation.
- public-key—URL where the public key file is located. For asymmetric key URL validation.
- symmetric-key—Key (16 bytes) used for AES encryption of the signed URL. For asymmetric key URL validation.

The following new attributes have been added to the Rule_UrlGenerateSign:

- key—Unique URL signature key that is up to 16 characters. For symmetric key URL validation.
- private-key—URL where the private key file is located. For asymmetric key URL validation.
- symmetric-key—Key (16 bytes) used for AES encryption of the signed URL. For asymmetric key URL validation.

The key ID owner and key ID number fields apply to the per-device configuration of URL Signing (**Devices > Devices > Service Control > URL Signing**). For compatibility, key ID owner and key ID number are required for the Rule_UrlGenerateSign action and are set to 1 when the URL signing key is specified in the Service Rule XML file. If the URL signing key is specified by using the URL Signing page or the **url-signature** command for each SE, the UrlGenerateSign action will find the key by the key-id-owner and key-id-number specified in the Rule_UrlGenerateSign action, and the Rule_Validate action will find the key by the KO (key-id-owner) and KN (key-id-number).

The following rules apply for Rule_Validate and Rule_UrlGenerateSign actions:

- key-id-owner and key-id-number are required attributes for the UrlGenerateSign action
- Only http is supported as the protocol attribute value for Rule_UrlGenerateSign; all other values have no affect.
- Rule_Validate supports http, rtsp, and rtmp as the protocol attribute value.

Following is an example of the Service Rule XML file configured with a symmetric key (also known as shared secret):

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
  <Revision>1.0</Revision>
  <CustomerName>Cisco</CustomerName>
  <ApplyAllTier>yes</ApplyAllTier>
  <Rule_Patterns>
    <PatternListGrp id = "grp1">
      <Domain>cds.cisco.com</Domain >
    </PatternListGrp>
  </Rule_Patterns>
<Rule_Actions>
    <Rule_UrlGenerateSign matchGroup="grp1" protocol="http" key-id-owner="1"
key-id-number="1" key="cisco123" timeout-in-sec="50" />
    <Rule_Validate matchGroup="grp1" key="cisco123" protocol="all"
error-redirect-url="http://wwwin.cisco.com" />
  </Rule_Actions>
</CDSRules>
```

Following is an example of the Service Rule XML file configured with an asymmetric key (also known as public key):

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
  <Revision>1.0</Revision>
  <CustomerName>Cisco</CustomerName>
  <ApplyAllTier>yes</ApplyAllTier>
  <Rule_Patterns>
    <PatternListGrp id = "grp1">
      <Domain>cds.cisco.com</Domain >
    </PatternListGrp>
  </Rule_Patterns>
<Rule_Actions>
    <Rule_UrlGenerateSign matchGroup="grp1" protocol="http" key-id-owner="1"
key-id-number="1" private-key="http://10.74.61.69/vod/private_key.txt"
symmetric-key="ciscociscociscoc" timeout-in-sec="50" />
    <Rule_Validate matchGroup="grp1" public-key="http://10.74.61.69/vod/public_key.txt"
symmetric-key="ciscociscociscoc" protocol="all"
error-redirect-url="http://wwwin.cisco.com" />
  </Rule_Actions>
</CDSRules>
```

Windows Media Streaming ASX Files with URL Signing

The Windows Media Streaming ASX Files with URL Signing feature uses the Rule_UrlGenerateSign rule action in the Service Rule file.

When the playback URL for a Windows Media Streaming live program has an ASX extension, the Content Abstraction Layer (CAL) returns metadata with an ASX file generated that contains both an HTTP URL and an RTSP URL for playback of the live program. These two URLs should be signed so that subsequent requests to playback the live program can be validated by the SE.

The Rule_UrlGenerateSign Rule_Action provides the ability to internally generate URL signatures using Version 2 of the URL signing script (SHA-1 encryption, protocol removed from beginning of the URL, and domain name not included). When the signed URL is sent back to the client as part of the ASX response, the domain name received from the client is added back in.

ASX File Request Flow

The request flow is as follows:

1. Client requests an ASX file.
2. A Service Rule XML file is configured for the delivery service that contains the new Rule_Action, Rule_UrlGenerateSign. The Rule_UrlGenerateSign Rule_Action element requires the following attribute values: Key Owner, Key Number, and timeout. If the timeout attribute value is not specified, the default value of 30 seconds is used. The range for the timeout value is from 0 to 50 seconds.
3. If the pattern for Rule_UrlGenerateSign is matched, the URL signature is generated by the SE using Version 2 of the URL signing script and the attribute values specified for the Rule_UrlGenerateSign element.

   Internally signed URLs will have IS=1. The IS=0 string is for legacy support with some VDSVDS components that use both internal and external signing mechanisms.

Both the HTTP and RTSP signed URLs are contained in the ASX file. The signed URL that is used is determined by which protocol (HTTP or RTSP) is allowed or disallowed in the Windows Media Streaming configuration.

Note If Windows Media Streaming is disabled, a 500 internal server message is sent to the client. The ASX file is not generated if Windows Media Streaming is disabled.

4. The client receives the ASX file with the signed URL. The player parses the ASX file and sends out the request again with the signed URL. The SE receives the signed URL and validates it. If the validation succeeds, the client is served the content.

The Service Rule XML file has to be created and uploaded through the CDSM GUI, then assign to the delivery service.

Rule_UrlGenerateSign Configuration Example for Two Delivery Services and One Origin Server

As previously mentioned, the Rule_UrlGenerateSign rule action works with files that have the .asx extension, which are requests for Windows Media Streaming live content. The .asx request is first handled by the Web Engine, which treats it as a VOD request.

The following example describes how to configure the Service Rule XML file for two delivery services (one live and one VOD) and one Origin server. The two delivery services, wmt-live and wmt-vod, hare the same content origin server that has an RFQDN of cds.cisco.com.

Create two Service Rule XML files:

- url_generate.xml—Assign this Service Rule file to the wmt-vod delivery service
- url_validate.xml—Assign this Service Rule file to the wmt-live delivery service

When the first request, http://cds.cisco.co/wmt-live.asx, comes in, the Rule_UrlGenerateSign rule in the url_generate.xml file generates a signed request in the reply. See the Example of url_generate.xml File section. Following is an example of the reply:

```
<ASX version="3">
  <Entry>
    <ref HREF="rtsp://cds.cisco.com/wmt-live?
SIGV=3&IS=1&KO=1&KN=1&US=sy1FVrgXxH4=9wWgxPK4fdO1b9ShREo4SqkojQAYndseOfn8cQf+5JdtpbRNy0eCS
dQ/ndXbhhYQSBXh3PMq04YG4umA/yDDMeB3TfhHSWQvkaDLLOjJa0xUYQ=="/>
    <ref HREF="http://cds.cisco.com/wmt-live?S
IGV=3&IS=1&KO=1&KN=1&US=sy1FVrgXxH4=9wWgxPK4fdO1b9ShREo4SqkojQAYndseOfn8cQf+5JdtpbRNy0eCSd
Q/ndXbhhYQSBXh3PMq04YG4umA/yDDMeB3TfhHSWQvkaDLLOjJa0xUYQ=="/>
  </Entry>
</ASX>
```

When the second request comes in:

```
rtsp://cds.cisco.com/wmt-live?=3&IS=1&KO=1&KN=1&US=sy1FVrgXxH4=9wWgxPK4fdO1b9ShREo4SqkojQA
YndseOfn8cQf+5JdtpbRNy0eCSdQ/ndXbhhYQSBXh3PMq04YG4umA/yDDMeB3TfhHSWQvkaDLLOjJa0xUYQ==
```

The Rule_Validate rule in the url_validate.xml file validates the request. See the Example of the url_validate.xml File.

Example of url_generate.xml File

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Cisco</CustomerName>
        <ApplyAllTier>yes</ApplyAllTier>
        <Rule_Patterns>
```

```
              <PatternListGrp id = "grp1">
                   <Domain>cds.cisco.com</Domain >
              </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                <Rule_UrlGenerateSign matchGroup="grp1" protocol="http" key-id-owner="1"
key-id-number="1" private-key="http://10.74.61.69/vod/private_key.txt"
symmetric-key="ciscociscociscoc" timeout-in-sec="50" />
        </Rule_Actions>
</CDSRules>
```

Example of the url_validate.xml File

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Cisco</CustomerName>
        <ApplyAllTier>yes</ApplyAllTier>
        <Rule_Patterns>
              <PatternListGrp id = "grp1">
                   <Domain>cds.cisco.com</Domain >
              </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                <Rule_Validate matchGroup="grp1"
public-key="http://10.74.61.69/vod/public_key.txt" symmetric-key="ciscociscociscoc"
protocol="all" error-redirect-url="http://wwwin.cisco.com" />
        </Rule_Actions>
</CDSRules>
```

Service Rule Action Order for Rule_Validate and Rule_UrlGenerateSign

The Rule_Actions processing is the same as described in "Rule Action Processing" section on page E-11; all Rule_Actions are processed in the same order as they are listed in the Rule_Actions element. However, for Rule_Validate and Rule_UrlGenerateSign, if the pattern is matched, and the URL validation or URL generation fails and there is a Rule_UrlRewrite or Rule_NoCache listed before, neither will be performed. Because the Rule_Validate or Rule_UrlGenerateSign process failed (validation or generation respectively), the authserver returns Action_Deny and the corresponding rule action (either Action_validate or Action_UrlGenerateSign). The Action_rewrite is not returned, nor is the action for Rule_NoCache if it is listed. This is true whenever Rule_Validate or Rule_UrlGenerateSign is listed, the pattern is matched, and the action fails (either URL validation or URL signing fails).

If either Rule_Validate or Rule_UrlGenerateSign is listed, the pattern is matched, and the action is successful, and if Rule_UrlRewrite is listed, then the Action_rewrite is returned and so is the Action_validate and Action_UrlGenerateSign (if all three rules are listed).

Service Rule Processing for Rule_Validate and Rule_UrlGenerateSign

This section describes the rule processing in general, and specifically addresses when Rule_UrlGenerateSign and Rule_Validate are included in the Rule_Actions.

Note Pattern match failure as described in this section means that none of the patternGrps specified as part of the matchGroup matched for a particular action.

Rule_Allow

If pattern match fails, the request is blocked and there is no further processing of the remaining rules.

If pattern match is successful, rule processing continues to the next rule action.

Rule_Block

If there is a pattern match for Rule_Block, the request is blocked and there is no further processing of the remaining rules.

If there is no pattern match for Rule_Block, rule processing continues to the next rule action.

Rule_UrlRewrite, Rule_NoCache, Rule_Validate. Rule_UrlGenerateSign—Pattern Match Failure Case

If pattern match fails, rule processing continues to the next rule action and there is no return value for the specified rule action. For example, if the rule action was Rule_Validate and the pattern match failed, there would be no URL validation performed on the request.

In the following XML example, because the pattern match failed for the action Rule_Validate, authserver does not return Action_validate. Because the Rule_UrlRewrite and Rule_UrlGenerateSign pattern matches were successful, authserver returns those actions in its response.

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
    <CustomerName>ATT</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <UrlRegex>asx</UrlRegex>
                </PatternListGrp>
                <PatternListGrp id = "grp2">
                                <UrlRegex>abcd</UrlRegex>
                </PatternListGrp>
        </Rule_Patterns>
        <Rule_Actions>
        <Rule_UrlGenerateSign matchGroup = "grp1" key-id-owner = "1" key-id-number = "1"
timeout-in-sec = "30" protocol = "http" />
        <Rule_Validate matchGroup = "grp2"  error-redirect-url="http://4.0.1.6/index.html"
protocol = "http" />
        <Rule_UrlRewrite matchGroup = "grp1" protocol = "http" regsub = "DejaVu"
rewrite-url = "dummy" />
        </Rule_Actions>
</CDSRules>
```

Rule_UrlRewrite, Rule_No_Cache, Rule_Validate, Rule_UrlGenerateSign—Pattern Match Success Case

If pattern match is successful, the actions are processed as described in the following subsections:

- Rule_Validate, Rule_UrlGenerateSign—Validation Fails, Signing Fails, Configuration Failure
- Rule_UrlRewrite and Rule_NoCache—Rewrite Fails
- Rule_UrlRewrite, Rule_NoCache, Rule_Validate, Rule_UrlGenerateSign—Success

Rule_Validate, Rule_UrlGenerateSign—Validation Fails, Signing Fails, Configuration Failure

Rule_Validate and Rule_UrlGenerateSign have a higher priority than Rule_UrlRewrite or Rule_NoCache. If the pattern matches, but the function fails (URL validation fails, URL signing fails, or there is a configuration failure), there is no further processing of the rule actions and the request is denied.

authserver returns [Action_Deny + Action_validate] if validation/UrlSignature generation fails.

authserver returns [Action_Deny + Action_UrlGenerateSign] if UrlSignature generation fails.

Also, the value from previous actions is not returned in either case. For example, if Rule_UrlRewrite preceded Rule_UrlGenerateSign, and Rule_UrlRewrite was successful, but Rule_UrlGenerateSign failed, authserver does not return the value for Action_Rewrite. Similarly, if Rule_UrlRewrite preceded Rule_Validate, and Rule_UrlRewrite was successful, but Rule_Validate failed, authserver would not return the value for Action_Rewrite. The same logic that is described for Rule_UrlRewrite applies to Rule_NoCache as well.

The following XML example illustrates the above scenarios:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
    <CustomerName>ATT</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <UrlRegex>asx</UrlRegex>
                </PatternListGrp>
                <PatternListGrp id = "grp2">
                                <UrlRegex>abcd</UrlRegex>
                </PatternListGrp>
        </Rule_Patterns>
        <Rule_Actions>
        <Rule_UrlRewrite matchGroup = "grp1" protocol = "http" regsub = "DejaVu"
rewrite-url = "dummy" />
        <Rule_UrlGenerateSign matchGroup = "grp1" key-id-owner = "1" key-id-number = "1"
timeout-in-sec = "30" protocol = "http" />
       <Rule_Validate matchGroup = "grp2"  error-redirect-url="http://4.0.1.6/index.html"
protocol = "http" />
        </Rule_Actions>
</CDSRules>
```

Rule_UrlRewrite and Rule_NoCache—Rewrite Fails

Rule_UrlRewrite and Rule_NoCache have a lower priority than Rule_Validate and Rule_UrlGenerateSign. If the pattern matches, but the Rule_UrlRewrite or Rule_NoCache fails, authserver does not return Action_Deny and processing of remaining rules actions continues. If Rule_UrlRewrite fails, authserver does not return the value for Action_Rewrite. If Rule_NoCache fails, authserver does not return its value.

Rule_UrlRewrite, Rule_NoCache, Rule_Validate, Rule_UrlGenerateSign—Success

If the Rule_UrlRewrite action is successful, authserver response contains the Action_Rewrite and the new rewritten URL is sent. Processing of the remaining rules actions continues.

If the Rule_NoCache action is successful, authserver sends the instructions to not cache the content. Processing of the remaining rules actions continues.

If Rule_Validate is successful, authserver response contains the Action_Validate.

If Rule_UrlGenerateSign is successful, authserver response contains Action_UrlGenerateSign.

Converting Old Windows Media Streaming Service Rules for URL Signing and Validation

This section provides examples of converting the generate-url-signature and validate-url-signature service rule actions for Windows Media Streaming to the Service Rule format.

Note All Windows Media Streaming per-device service rules configured for URL signature and validation must be converted to the per-delivery service Service Rule XML file. This change only applies to the generate-url-signature and validate-url-signature service rule actions for Windows Media Streaming. The other service rule actions (allow, block, no-cache, redirect, refresh, replace, and rewrite) still use the per-device service rule configuration for Windows Media Streaming.

Perform URL Signature Generation on Requests

The following example shows the commands for configuring a service rule that performs URL signature generation on requests from the domain wmtvod.com using the old mechanism:

```
SE (config)# url-signature key-id-owner 1 key-id-number 1 key cisco123
SE (config)# rule enable
SE (config)# rule action generate-url-signature key-id-owner 1 key-id-number 1
pattern-list 1 protocol http
SE (config)# rule pattern-list 1 domain wmtvod.com
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
   <Revision>1.0</Revision>
   <CustomerName>Cisco</CustomerName>
   <ApplyAllTier>yes</ApplyAllTier>
    <Rule_Patterns>
      <PatternListGrp id = "grp1">
         <Domain>wmtvod.com</Domain >
      </PatternListGrp>
   </Rule_Patterns>

   <Rule_Actions>
      <Rule_UrlGenerateSign matchGroup = "grp1" protocol = "http" key="cisco123
key-id-owner="1" key-id-number="2" timeout-in-sec="30"/>
   </Rule_Actions>
</CDSRules>
```

Perform URL Signature Validation on Requests

The following example shows the commands for configuring a service rule that performs URL signature validation on requests from the domain, wmtvod.com using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action validate-url-signature error-redirect-url www.cisco.com
pattern-list 1  protocol all
SE (config)# rule pattern-list 1 domain wmtvod.com
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
  <Revision>1.0</Revision>
  <CustomerName>Capricious</CustomerName>
  <Rule_Patterns>
```

```
    <PatternListGrp id = "grp1">
      <Domain>wmtvod.com</Domain>
    </PatternListGrp>
  </Rule_Patterns>
  <Rule_Actions>
    <Rule_Validate matchGroup = "grp1" protocol = "all" key="cisco123"
error-redirect-url="http://www.cisco.com"/>
  </Rule_Actions>
</CDSRules>
```

Note The Rule_Validate action can also be configured without the *key* attribute, if the key is defined for each SE by using the CDSM GUI URL Signing page or by using the **url-signature** command.

Rule Actions for Flash Media Streaming

Service rules for Flash Media Streaming are now configured using the Service Rule file. By associating the Service Rule file with a delivery service, all service rules defined in the file are applied to all SEs in the delivery service.

The following service rule actions are supported for Flash Media Streaming:

- Allow (Rule_Allow)
- Block (Rule_Block)
- URL signature validation (Rule_Validate)
- SWF file validation (Rule_SwfFileValidate)
- DSCP (Rule_Dscp)

Note Starting from Release 3.3, VDS-IS supports per session DSCP marking for Flash Media streaming, VOD, and Live.

Converting Old Flash Media Streaming Service Rules

The following example shows an example of each rule action for Flash Media Streaming using the old mechanism and the conversion to the Service Rule format.

Note Currently, the header field referrer is not supported.

Block Requests

The following example shows the commands for configuring a service rule that blocks RTMP requests from the domain fmsvod.com using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action block pattern-list 1  protocol rtmp
SE (config)# rule pattern-list 1 domain fmsvod.com
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
```

```
        <Revision>1.0</Revision>
        <CustomerName>Capricious</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <Domain>fmsvod.com</Domain>
                </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                   <Rule_Block matchGroup = "grp1" protocol = "rtmp"  />
        </Rule_Actions>
</CDSRules>
```

Allow Requests

The following example shows the commands for configuring a service rule that allows RTMP requests from the domain fmsvod.com using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action allow pattern-list 1  protocol rtmp
SE (config)# rule pattern-list 1 domain fmsvod.com
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Capricious</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <Domain>fmsvod.com</Domain>
                </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                   <Rule_Allow matchGroup = "grp1" protocol = "rtmp"  />
        </Rule_Actions>
</CDSRules>
```

Perform URL Signature Validation on Requests

The following example shows the commands for configuring a service rule that performs URL signature validation on requests from the domain fmsvod.com using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action validate-url-signature error-redirect-url www.cisco.com
pattern-list 1  protocol rtmp
SE (config)# rule pattern-list 1 domain fmsvod.com
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Capricious</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <Domain>fmsvod.com</Domain>
                </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                <Rule_Validate matchGroup = "grp1" protocol = "rtmp"
                 error-redirect-url="http://www.cisco.com"/>
        </Rule_Actions>
```

```
    </CDSRules>
```

Match on Regular Expression

Pattern matching can be performed on a regular expression instead of matching on the domain name in any of the Flash Media Streaming service rules. The following example shows the commands for configuring a service rule that allows RTMP requests that match the string “clouds” using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action allow pattern-list 1  protocol rtmp
SE (config)# rule pattern-list 1 url-regex clouds
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Capricious</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <UrlRegex>clouds</UrlRegex>
                </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                   <Rule_Allow matchGroup = "grp1" protocol = "rtmp"  />
        </Rule_Actions>
</CDSRules>
```

Match on Source IP address

Pattern matching can be performed on the source IP address instead of matching on the domain name in any of the Flash Media Streaming rules. The following example shows the commands for configuring a service rule that allows RTMP requests that match the source IP address 209.165.201.1 using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action allow pattern-list 1  protocol rtmp
SE (config)# rule pattern-list 1 src-ip 209.165.201.10 255.255.0.0
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
        <Revision>1.0</Revision>
        <CustomerName>Capricious</CustomerName>
        <Rule_Patterns>
                <PatternListGrp id = "grp1">
                        <SrcIp>209.165.201.10/16</SrcIp>
                </PatternListGrp>
        </Rule_Patterns>

        <Rule_Actions>
                   <Rule_Allow matchGroup = "grp1" protocol = "rtmp"  />
        </Rule_Actions>
</CDSRules>
```

Support for SWF Validation

Small Web Format (SWF) file validation is supported by using the Service Rule XML file. A client player generates the signature for an SWF file and the signature is sent to the Flash Media Streaming engine. The client SWF file is validated against the SWF file on the SE. If the subscriber edits the SWF file or uses a malicious SWF file, the signatures differ and the request is rejected.

SWF Validation Process

If SWF validation is required, the Authorization Server tells Flash Media Streaming whether SWF file validation needs to be performed or not for a particular delivery service. Flash Media Streaming then fetches and accesses the SWF file and uses it to validate the request.

The Authorization Server determines if the SWF file verification needs to be done or not based on the rules listed in Service Rule file.

Note The SWF file must be stored on the local disk of the SE. In a cache-miss case, the entire SWF file must be retrieved before SWF validation can continue.

An algorithm is used to generate a hash of the SWF file by using the file size of the original SWF file and the location.

If the Authorization Server says SWF verification is not required, a property is set telling Flash Media Streaming to bypass it.

The SWF validation is performed by comparing the hash generated by Flash Media Streaming with the hash sent by the client. The client-side hash is generated automatically by the Flash Media player when an RTMP connection is made.

If the hashes match, the SWF validation is successful and the request is allowed; if the hashes do not match, the SWF validation is not successful and the request is denied.

Note The SWF validation does not apply to interactive applications.

Web Engine revalidation should be enabled so that the latest SWF file is used. If Web Engine revalidation is not enabled, then an older SWF file may be used for validation for up to one hour after the entry in the cache of hashes has expired. Revalidation is enabled by default on the Web Engine.

If Authorization Server is disabled, SWF validation is always performed.

Interaction with Web Engine

When Web Engine receives a Flash Media Streaming request and SWF validation is enabled for the delivery service, the original SWF file must be on the local disk. If the file is not found in the /local/local1/swfs directory, Web Engine performs a lookup and the file is cached on the local disk. If the SWF file is found at the cached location, Web Engine performs a cache revalidation, if applicable. In a cache-miss case, the entire SWF file must be retrieved before SWF validation can continue. If the URL of the SWF file is an Origin Server fully-qualified domain name (OFQDN)-based URL or a Service Router fully-qualified domain name (RFQDN)-based URL, Web Engine caches the file to the local disk. If the URL is other than these two, Web Engine treats it as a proxy request and the Flash Media Streaming engine writes the file to disk at the /local/local1/swfs directory (the file is deleted after the request is processed).

There are five possible successful responses from Web Engine: cache miss, cache hit, alien hit, pre-position, or proxy. In the first four cases, Flash Media Streaming reads the SWF file directly and adds it to the cache of hashes. In the proxy request case, the file is written to the /local/local1/swfs directory and deleted after the SWF validation is complete. The hash is not added to the cache of hashes in the proxy-request case.

Note If the SWF file is uploaded to individual SEs at the /local/local1/swfs directory, revalidation of the SWF file is not performed, which means that if the SWF file is modified, the new file has to be uploaded to the SEs again. This has to be done for every SE in the delivery service.

If Authorization Server is disabled, or if the SWF validation is not enabled, the SWF validation is also not performed on the locally uploaded files. The SWF Validation feature assumes that the SWF file is being requested from an HTTP location; therefore, if the SWF file is located on a personal computer with a path similar to “c:/Documents/,” the SWF validation rejects the request.

Service Rule File Example for SWF Validation

Following is an example of the SWF validation in the Service Rule XML file:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
   <Revision>1.0</Revision>
   <CustomerName>Company</CustomerName>
   <Rule_Patterns>
      <PatternListGrp id = "grp1">
         <Domain>demo.cdsis.com</Domain>
      </PatternListGrp>
   </Rule_Patterns>
   <Rule_Actions>
      <Rule_SwfFileValidate matchGroup = "grp1" protocol = "rtmp"  />
   </Rule_Actions>
</CDSRules>
```

The *matchGroup* attribute value is the list of PatternListGrp *id* attributes. The *protocol* attribute value must be rtmp, rtmpt, rtmpe, rtmpte, or all.

Note Multiple protocols can be specified for the same rule by including each protocol as a value of the protocol attribute in the form of a comma-separated string.

Whether a Rule_SwfFileValidate is matched or not, rule processing continues to the next configured rule. Rule_SwfFileValidate action enables Flash Media Streaming to perform SWF file validation. If the Rule_SwfFileValidate pattern is matched and the SWF file validation fails, then the request is rejected.

Support for DSCP Marking

The DSCP per delivery service requires to configure domain name in the rule file.The rule will match the matchGroup defined by a regex pattern or domain name and the attribute dscp-bits will be applied to the matching pattern. The attribute is the DSCP value ranging from 0 to 63. Absence of the tag in the rules xml file shall assume default DSCP value to 0.

Service Rule File Example for DSCP Marking

The following example shows the commands for configuring a service rule that allows RTMP requests that match the amsvod domain name using the old mechanism:

```
SE (config)# rule enable
SE (config)# rule action allow pattern-list 1 protocol rtmp
SE (config)# rule pattern-list 1 domain amsvod
```

The Service Rule XML file for the above rule is as follows:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
<Revision>1.0</Revision>
<CustomerName>BT</CustomerName>
<Rule_Patterns>
<PatternListGrp id = "grp1">
<Domain>amsvod.com</Domain>
</PatternListGrp>
</Rule_Patterns>
<Rule_Actions>
<Rule_Dscp matchGroup = "grp1" protocol = "all" dscp-bits = "20" />
</Rule_Actions>
</CDSRules>
```

Service Rule File Example

The following is an example of a Service Rule file:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
    <Revision>1.0</Revision>
    <CustomerName>Capricious</CustomerName>
    <Rule_Patterns>
        <PatternListGrp id = "grp1">
                <UrlRegex>videos</UrlRegex>
                <Domain>rfqdn.cds.com</Domain>
        </PatternListGrp>
        <PatternListGrp id = "grp2">
            <Domain>dummy.cds.com</Domain>
            <SrcIp>10.10.10.10</SrcIp>
        </PatternListGrp>
        <PatternListGrp id = "grp3">
            <SrcIp>10.21.148.231</SrcIp>
        </PatternListGrp>
        <PatternListGrp id = "grp5">
        <UrlRegex>/*</UrlRegex>
        </PatternListGrp>
        <PatternListGrp id = "grp6">
            <Domain>rfqdn.cds.com</Domain>
        </PatternListGrp>
    </Rule_Patterns>
    <Rule_Actions>
        <Rule_Allow matchGroup = "grp1,grp5" protocol = "http"  />
        <Rule_UrlRewrite matchGroup = "grp1" protocol = "http" regsub = "videos"
rewrite-url = "http://dummy.cds.com" />
        <Rule_Block matchGroup = "grp3" protocol = "http" />
        <Rule_Validate matchGroup = "grp5"  protocol = "http" error-redirect-url =
"http://wwwin.cisco.com" exclude-validation = "all" />
    </Rule_Actions>
```

```
</CDSRules>
```

Service Rule File for URL Validation and the Exclude-Validation Attribute

As part of the URL Signing feature, to validate signed URLs for the Web Engine, you must configure the Service Rule file for URL Validation. The exclude-validation attribute offers the option to exclude the client IP address , the expiry time, or both from the URL validation process. The following sections explain the different exclude validation options:

- Exclude Client IP address from URL Validation
- Exclude Expiry Time from URL Validation
- Exclude Both the Client IP address and the Expiry Time from URL Validation

Exclude Client IP address from URL Validation

While performing URL validation, the SE compares the IP address from which it received the request and the CIP field in the signed URL request. The client IP address is a required parameter and is displayed as the CIP field in the signed URL request. If you configure the exclude-validation attribute with the client-ip value in the Service Rule XML file, the URL validation process ignores the client IP address during the validation process.

Following is an example of the Service Rule XML file with the exclude-validation attribute set to client-ip:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
 <Revision>1.0</Revision>
 <CustomerName>ATT</CustomerName>
 <Rule_Patterns>
  <PatternListGrp id = "grp1">
   <Domain>iphone.com</Domain>
  </PatternListGrp>
 </Rule_Patterns>
 <Rule_Actions>
  <Rule_Validate matchGroup = "grp1" protocol="http" exclude-validation="client-ip"
error-redirect-url = "http://wwwin.cisco.com"/>
 </Rule_Actions>
</CDSRules>
```

Exclude Expiry Time from URL Validation

Without the exclude-validation expiry-time attribute, he generated URL would be valid only for a stipulated period of time mentioned at the time of signing. This is indicated in the ET field in the signed URL. The ET field value is generated with respect to the local time on the server used for signing. The expriy time relies on the synchronization of the devices; for more information, see the "Importance of Device Synchronization" section on page H-13.

On receiving the request, the URL validation process compares the time stamp on the SE with the time stamp in the ET field of the received request. If the time stamp on the request is less than the time stamp on the SE, the request is rejected because of the expiry time lapse.

To bypass the expiry time validation, use the exclude-validation attribute with the expiry-time value in the Service Rule XML file.

Following is an example of the Service Rule XML file with the exclude-validation attribute set to expiry-time:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
 <Revision>1.0</Revision>
 <CustomerName>ATT</CustomerName>
 <Rule_Patterns>
  <PatternListGrp id = "grp1">
   <Domain>iphone.com</Domain>
  </PatternListGrp>
 </Rule_Patterns>
 <Rule_Actions>
   <Rule_Validate matchGroup = "grp1" protocol="http" exclude-validation="expiry-time"
error-redirect-url = "http://wwwin.cisco.com"/>
 </Rule_Actions>
</CDSRules>
```

Exclude Both the Client IP address and the Expiry Time from URL Validation

The exclude-validation attribute with the all value excludes both the client-ip and the expiry-time from the URL validation process. Meaning the SE considers the request successful even if the request comes from a different client than what is mentioned in the signed URL and the expiry-time has lapsed.

Following is an example of the Service Rule XML file with the exclude-validation attribute set to all:

```
<CDSRules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="schema\CDSRules.xsd">
 <Revision>1.0</Revision>
 <CustomerName>ATT</CustomerName>
 <Rule_Patterns>
  <PatternListGrp id = "grp1">
   <Domain>iphone.com</Domain>
  </PatternListGrp>
 </Rule_Patterns>
 <Rule_Actions>
  <Rule_Validate matchGroup = "grp1" protocol="http" exclude-validation="all" error-redirect-url = "http://wwwin.cisco.com"/>
 </Rule_Actions>
</CDSRules>
```

Note The *exclude-validation exclude-domain* attribute instructs the SEs to ignore the domain in the URL when processing the validation of the signed URL.