Using an XML Document to Configure the CSS

The CSS Content Application Program Interface (API) feature allows you to use a network management workstation to make web-based configuration changes to the CSS using Extensible Markup Language (XML) documents. XML is a powerful tool that can be used to automatically configure a CSS using all of the CLI commands included in the CSS software, such as to specify server weight and load, to configure load balancing across a group of servers, or to configure content rules to restrict access to a group of directories or files on the servers.

XML code loads a series of CLI commands into the CSS without the need to respond to the prompts, similar to operating in expert mode. As the CSS administrator, plan which type of changes you want to implement and the consequences of these changes as they are performed.

After you create the XML document, you publish (upload) the XML file to the Hypertext Transfer Protocol (HTTP) server embedded in the CSS using an HTTP PUT method.

This chapter contains the following major sections:

Creating XML Code

When developing XML code for the Content API to issue CLI commands, adhere to the following guidelines. You can use any text editor for creating the XML code. The maximum number of characters for each tag set is 300.

1. Include the following line as the first line in the XML file:

   ```
   <?xml version="1.0" standalone="yes"?>
   ```

2. Enclose the CLI commands within the <action></action> tag set. For example:

   ```
   <action>add service MyServiceName</action>
   <action>vip address 10.2.3.4</action>
   ```

   Note A nested **script play** command (to execute a script line by line from the CLI) is not allowed in an XML file. This restriction is enforced because the actual execution of the XML tag set is performed within a **script play** command

3. If special characters are required in an XML configuration, be aware of the following considerations:

   - The CSS supports the inclusion of the ~, !, @, #, $, ^, &, *, (, and) characters in XML content. All other special characters (such as <, >, and %) are not supported in an XML configuration.
   - Special characters can be included in a service, owner, or content name provided that the name is included in the content of the XML element and the name is enclosed within an <action></action> tag set. For example, `<action>service My@#Service</action>`.
   - If special characters do form part of an XML tag, specifically an attribute, these characters are not supported (for example, `<service name = My@#Service>`. In this case, the command request may be rejected or the special characters may be discarded. You must enclose the special characters within an <action></action> tag set, as described above.

4. Pay attention to mode hierarchy of the CLI commands in the XML file. Each mode has its own set of commands. Many of the modes have commands allowing you to access other related modes. If you enter a series of commands in the improper mode hierarchy, this will result in an XML file that fails to execute properly.

   As an example, the following commands configure an access control list (ACL):

```
<?xml version=”1.0” standalone=”yes” ?>
<config>
    <action>acl 98</action>
        <action>clause 10 permit any any dest any</action>
    <action>apply circuit-(VLAN3)</action>
</config>
```

   In another example, the following commands configure a CSS Ethernet interface:

```
<?xml version=”1.0” standalone=”yes” ?>
<config>
    <action>interface ethernet-6</action>
    <action>bridge vlan 3</action>
    <action>circuit VLAN3</action>
        <action>ip address 10.10.104.1/16</action>
</config>
```

5. Pay attention to the allowable CLI command conventions for syntax and variable argument in the XML file. If you enter an invalid or incomplete command, this will result in an XML file that fails to execute properly.

Note For overview information on the CLI commands you can use in global configuration mode and its subordinate modes, refer to the *Cisco Content Services Switch Command Reference*, Chapter 2, “CLI Commands.”

XML Document Example

The following example is a complete XML document. The XML document creates three services, an owner, and a content rule, and assigns one of the newly created services to the content rule.

```
<?xml version="1.0" standalone="yes"?>
<config>
    <service name="router">
        <ip_address>10.0.3.1</ip_address>
        <action>active</action>
    </service>
    <service name="sname2">
        <ip_address>10.0.3.2</ip_address>
        <weight>4</weight>
        <action>active</action>
    </service>
    <service name="sname3">
        <ip_address>10.0.3.3</ip_address>
        <weight>5</weight>
        <protocol>udp</protocol>
        <action>suspend</action>
    </service>
    <service name="nick">
        <ip_address>10.0.3.93</ip_address>
        <action>active</action>
    </service>
    <owner name="test">
    <content name="rule">
        <vip_address>10.0.3.100</vip_address>
        <protocol>udp</protocol>
        <port>8080</port>
        <add_service>nick</add_service>
        <action>active</action>
    </content>
    </owner>
</config>
```

Allowing the Transfer of XML Configuration Files on the CSS

For client applications to publish XML configuration files on the CSS, they must upload the files to the CSS over HTTP connections. By default, the CSS denies HTTP connections for the transfer of XML configuration files. You can configure the CSS to allow the transfer of these files through either an unsecure HTTP connection or a secure HTTPS SSL connection. You cannot configure access for both secure and unsecure connections.

Note Because the CSS can process large configurations, the CSS processes only two concurrent XML configuration uploads over secure connections. If a third upload is attempted, it may not succeed. The CSS closes the connection and sends the following message to the client:

```
status 503 Service Unavailable
```

To allow the transfer of XML configuration files through:

- An unsecure connection, use the global configuration **no restrict xml** command. For example, enter:

  ```
  (config)# no restrict xml
  ```

  The CSS listens on port 80 for an unsecure connection request.

  To reset the default behavior to deny the transfer of XML configuration files through an unsecure connection, use the global configuration **restrict xml** command. For example, enter:

  ```
  (config)# restrict xml
  ```

- A secure SSL connection, use the global configuration **no restrict secure-xml** command. For example, enter:

  ```
  (config)# no restrict secure-xml
  ```

  The CSS listens on port 443 for a secure connection request. The client application can use SSL v2/3 or v3. However, the CSS performs all negotiations using SSL v3. The CSS requires a Secure Management license key to negotiate a secure connection using SSL strong encryption. Without the key, the CSS uses SSL weak encryption.

  To reset the default behavior to deny the transfer of XML configuration files through a secure connection, use the global configuration **restrict secure-xml** command. For example, enter:

  ```
  (config)# restrict secure-xml
  ```

Parsing the XML Code

After you complete the XML file, parse the code to ensure that it is syntactically correct. One easy way to parse XML code is to open the XML file directly from Microsoft Internet Explorer. Syntax errors are flagged automatically when the file is loaded. If an error occurs, review your XML code and correct all syntax errors.

Publishing the XML Code to the CSS

The completed XML file is remotely published (uploaded) to the HTTP server in the CSS from the external network management workstation by using an HTTP PUT method. The HTTP PUT method uses the IP address of the CSS as the destination URL where you want to publish the XML file.

Note When you configure TACACS+ on a CSS, the CSS does not authorize scripts through the TACACS+ server. Because the CSS transforms all XML commands into scripts, the CSS also does not authorize XML commands through the TACACS+ server.

When you publish XML files to the HTTP server in the CSS, the CSS requires a valid username and password as part of the user authentication process. The username must have SuperUser privileges to be able to add XML files to the CSS.

Ensure that the CLI commands in the XML document do not have an impact on the interface configuration through which the XML file transfer process occurs (for example, including the command **no ip addr**, which identifies the IP address of the CSS receiving the XML file). If this occurs, you will disconnect the workstation performing the XML file transfer.

Because the CSS can process large configurations, the CSS processes only two concurrent XML configuration uploads over secure connections. If a third upload is attempted, it may not succeed. The CSS closes the connection and sends the following message to the client:

```
status 503 Service Unavailable
```

Software is available to simplify the process of publishing XML files to the CSS HTTP server. These software packages offer a simple method to publish files to a web server. This software uses the unsecure HTTP or secure HTTPS protocol to publish files and requires no special software on the web server side of the connection.

Note An error code in the publishing process usually means that **no restrict secure-xml** or **no restrict xml** commands have not been issued on the CSS prior to publishing the XML file. See the "Allowing the Transfer of XML Configuration Files on the CSS" section for details.

Testing the Output of the XML Code

Test the output of the XML code by reviewing the running configuration of the CSS. After the XML has been successfully published to the CSS, Telnet to the switch and enter the **show running-config** command to verify that the XML changes have properly occurred. If the XML changes are incorrect or missing, republish the XML code to the CSS as described in the "Publishing the XML Code to the CSS" section.