Cisco UCS Manager XML API Method Descriptions

This chapter includes the following sections:

aaaChangeSelfPassword

The `aaaChangeSelfPassword` method changes the user's own password. The user supplies the old password for authentication, the new password, and a confirmation of the new password. If the user is authenticated successfully with the old password, the new password becomes effective.

Note Cisco UCS Manager Release 4.1(3a) onwards, users can reset their expired password using "null" in cookie.

Request Syntax

```
<xs:element name="aaaChangeSelfPassword" type="aaaChangeSelfPassword"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaChangeSelfPassword" mixed="true">
            <xs:attribute name="inUserName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inOldPassword">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inNewPassword">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inConfirmNewPassword">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaChangeSelfPassword" type="aaaChangeSelfPassword"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaChangeSelfPassword" mixed="true">
            <xs:attribute name="outStatus">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="success"/>
                         <xs:enumeration value="failure"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Example 1

Request

```
<aaaChangeSelfPassword
    cookie="<real_cookie>"
    inUserName="admin"
    inOldPassword="Nbv12345"
    inNewPassword="Mbv12345"
    inConfirmNewPassword="Mbv12345" />
```

Response

```
<aaaChangeSelfPassword
    cookie="<real_cookie>"
    response="yes"
    outStatus="success">
</aaaChangeSelfPassword>
```

Example 2

The following example shows how to reset the expired password.

Request

```
<aaaChangeSelfPassword
    cookie="null"
    inUserName="admin"
    inOldPassword="Nbv12345"
    inNewPassword="Mbv12345"
    inConfirmNewPassword="Mbv12345" />
```

Response

```
<aaaChangeSelfPassword
    cookie="<real_cookie>"
    response="yes"
    outStatus="success">
</aaaChangeSelfPassword>
```

aaaCheckComputeAuthToken

The `aaaCheckComputeAuthToken` method gets details on the specified token, such as the user name (who generated this token) and the user’s privileges and locales.

Request Syntax

```
<xs:element name="aaaCheckComputeAuthToken" type="aaaCheckComputeAuthToken"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaCheckComputeAuthToken" mixed="true">
            <xs:attribute name="inUser">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inToken" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaCheckComputeAuthToken" type="aaaCheckComputeAuthToken"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaCheckComputeAuthToken" mixed="true">
            <xs:attribute name="outAllow">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>

            <xs:attribute name="outRemote">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outAuthUser">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outLocales" type="xs:string"/>
            <xs:attribute name="outPriv">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:pattern
value="((ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equipment|ls-con
fig-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security|ls-config|
ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod-config|ls-
server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-security|ls-netwo
rk-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-network|ls-ext-ac
cess|fault),){0,35}(ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equip
ment|ls-config-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security
|ls-config|ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod
-config|ls-server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-securi
ty|ls-network-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-networ
k|ls-ext-access|fault){0,1}"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
 </xs:complexType>
```

Examples

Request

```
<aaaCheckComputeAuthToken
    cookie="<real_cookie>"
    inToken="04541875309302299687211"
    inUser="admin"/>
```

Response

```
<aaaCheckComputeAuthToken
    cookie="<real_cookie>"
    response="yes"
    outAllow="yes"
    outRemote="no"
    outAuthUser="admin"
    outLocales=""
    outPriv="admin,read-only">
</aaaCheckComputeAuthToken>
```

aaaCheckComputeExtAccess

The `aaaCheckComputeExtAccess` method validates whether a specified user has access to the server specified with the inDn parameter.

Request Syntax

```
<xs:element name="aaaCheckComputeExtAccess" type="aaaCheckComputeExtAccess"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaCheckComputeExtAccess" mixed="true">
            <xs:attribute name="inDn" type="referenceObject"/>
            <xs:attribute name="inUser">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaCheckComputeExtAccess" type="aaaCheckComputeExtAccess"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaCheckComputeExtAccess" mixed="true">
            <xs:attribute name="outAllow">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
 </xs:complexType>
```

Examples

Request

```
<aaaCheckComputeExtAccess
    cookie="<real_cookie>"
    inDn="sys/Chassis-1/blaade-2"
    inUser="gopis"/>
```

Response

```
<aaaCheckComputeExtAccess
    cookie="<real_cookie>"
    response="yes"
    outAllow="no">
</aaaCheckComputeExtAccess>
```

aaaGetNComputeAuthTokenByDn

The `aaaGetNComputeAuthTokenByDn` method returns the authentication tokens for `TokenLogin` to a particular server specified by DN.

Request Syntax

```
<xs:element name="aaaGetNComputeAuthTokenByDn" type="aaaGetNComputeAuthTokenByDn"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaGetNComputeAuthTokenByDn" mixed="true">
 <xs:attribute name="inDn">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inNumberOf" type="xs:unsignedByte"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaGetNComputeAuthTokenByDn" type="aaaGetNComputeAuthTokenByDn"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaGetNComputeAuthTokenByDn" mixed="true">
            <xs:attribute name="outUser">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outTokens" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaGetNComputeAuthTokenByDn
    cookie="<real_cookie>"
    inDn="sys/chassis-1/blade-2"
    inNumberOf="5"/>
```

Response

```
<aaaGetNComputeAuthTokenByDn
    cookie="<real_cookie>"
    response="yes"
    outUser="__computeToken__"
    outTokens="3550599419521612726721 1,9359555190852706023245 1,1176997309605730159399 1,527
    29538672765491844031,73106643969990280919791">
</aaaGetNComputeAuthTokenByDn>
```

aaaGetComputeAuthTokens

The `aaaGetComputeAuthTokens` method returns authentication tokens that are used to launch the KVM. This generates two temporary authentication tokens that are valid for 60 seconds. The first is the KVM user name and the second token is the password. Using the authorization tokens as credentials, you can access the URL from where you can download the Java Network Launch Protocol (JNLP) file. You can download the JNLP file from the URL and launch it to start a KVM session.

Note

- You cannot obtain tokens if the vKVM option is disabled on the CIMC.
- You must have user or admin privileges to the CIMC to obtain the authentication tokens. Users with read-only privileges will not be able to obtain the tokens.
- The authorization tokens expire is 60 seconds; you cannot use the tokens after 60 seconds to access the URL. If you try to access after 60 seconds, the login fails and you get a authentication failure or timeout message.

Request Syntax

```
<xs:element name="aaaGetComputeAuthTokens" type="aaaGetComputeAuthTokens"
substitutionGroup="externalMethod"/>
     <xs:complexType name="aaaGetComputeAuthTokens" mixed="true">
         <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
         <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaGetComputeAuthTokens" type="aaaGetComputeAuthTokens"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaGetComputeAuthTokens" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="outTokens">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:complexType>
```

Examples

Request:

```
aaaGetComputeAuthTokens
cookie="<real_cookie>" />
```

Response:

```
<aaaGetComputeAuthTokens cookie="<real_cookie>" outTokens="1804289383,846930886"
response="yes"> </aaaGetComputeAuthTokens>
```

aaaKeepAlive

The `aaaKeepAlive` method keeps the session active until the default session time expires, using the same cookie after the method call.

Request Syntax

```
<xs:element name="aaaKeepAlive" type="aaaKeepAlive" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaKeepAlive" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaKeepAlive" type="aaaKeepAlive" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaKeepAlive" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaKeepAlive
    cookie="<real_cookie>" />
```

Response

```
<aaaKeepAlive
    cookie="<real_cookie>"
    commCookie="11/15/0/2969"
    srcExtSys="10.193.33.109"
    destExtSys="10.193.33.109"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes">
</aaaKeepAlive>
```

aaaLogin

The aaaLogin method is the login process and is required to begin a session. This action establishes the HTTP (or HTTPS) session between the client and Cisco UCS.

Note When the password expiry feature is enabled, the aaaLogin API indicates the expiry of password in the XML API response.

Request Syntax

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogin" mixed="true">
            <xs:attribute name="inName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inPassword">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogin" mixed="true">
            <xs:attribute name="outCookie" type="xs:string"/>
            <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
            <xs:attribute name="outPriv">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:pattern
value="((ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equipment|ls-con
fig-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security|ls-config|
ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod-config|ls-
server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-security|ls-netwo
rk-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-network|ls-ext-ac
cess|fault),){0,35}(ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equip
ment|ls-config-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security
|ls-config|ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod
-config|ls-server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-securi
ty|ls-network-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-networ
k|ls-ext-access|fault){0,1}"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outSessionId">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="32"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outVersion" type="xs:string"/
            <xs:attribute name="outname" type="xs:string"/>
            <xs:attribute name="outPasswdExpiryStatus" type="xs:string"/>
                <xs:enumeration value="None"/>
                <xs:enumeration value="Warning"/>
                <xs:enumeration value="Expired"/>
            <xs:attribute name="outPasswdExpiryDuration" type="xs:unsignedInt"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaLogin
    inName="admin"
    inPassword="Nbv12345"/>
```

Response

```
<aaaLogin
    cookie=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin,read-only"
    outDomains=""
    outChannel="noencssl"
    outEvtChannel="noencssl"
    outSessionId="web_41246_A"
    outVersion="1.4(0.61490)">
    outName="username"
    outPasswdExpiryStatus="None"
    outPasswdExpiryDuration="1"
</aaaLogin>
```

aaaLogout

The `aaaLogout` method is a process to close a web session by passing the session cookie as input. It is not automatic; the user has to explicitly invoke the `aaaLogout` method to terminate the session.

Request Syntax

```
<xs:element name="aaaLogout" type="aaaLogout" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogout" mixed="true">
            <xs:attribute name="inCookie" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaLogout" type="aaaLogout" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogout" mixed="true">
            <xs:attribute name="outStatus">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="success"/>
                         <xs:enumeration value="failure"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaLogout
    inCookie="<real_cookie>"/>
```

Response

```
<aaaLogout
    cookie=""
    response="yes"
    outStatus="success">
</aaaLogout>
```

aaaRefresh

The aaaRefresh method keeps sessions active (within the default session time frame) by user activity. There is a default of 600 seconds that counts down when inactivity begins. If the 600 seconds expire, Cisco UCS enters a sleep mode. It requires signing back in, which restarts the countdown. It continues using the same session ID.

Note

- Using this method expires the previous cookie and issues a new cookie.
- When the password expiry feature is enabled, the aaaRefreshAPI indicates the expiry of password in the XML API response.

Request Syntax

```
<xs:element name="aaaRefresh" type="aaaRefresh" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaRefresh" mixed="true">
            <xs:attribute name="inName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inPassword">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inCookie" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaRefresh" type="aaaRefresh" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaRefresh" mixed="true">
            <xs:attribute name="outCookie" type="xs:string"/>
            <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
            <xs:attribute name="outPriv">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:pattern
value="((ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equipment|ls-con
fig-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security|ls-config|
ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod-config|ls-
server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-security|ls-netwo
rk-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-network|ls-ext-ac
cess|fault),){0,35}(ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equip
ment|ls-config-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security
|ls-config|ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod
-config|ls-server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-securi
ty|ls-network-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-networ
k|ls-ext-access|fault){0,1}"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="outname" type="xs:string"/>
            <xs:attribute name="outPasswdExpiryStatus‟ type="xs:string">
                <xs:enumeration value="None"/>
                <xs:enumeration value="Warning"/>
                <xs:enumeration value="Expired"/>
            <xs:attribute name="outPasswdExpiryDuration‟ type="xs:unsignedInt"/>
        </xs:complexType>
```

Examples

Request

```
<aaaRefresh
    cookie="<real_cookie>"
    inName="admin"
    inPassword="Nbv12345"
    inCookie="<real_cookie>"/>
```

Response

```
<aaaRefresh
    cookie="<real_cookie>"
    commCookie="" srcExtSys="0.0.0.0"
    destExtSys="0.0.0.0"
    srcSvc=""
    destSvc=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="7200"
    outPriv="admin"
    outDomains=""
    outChannel="fullssl"
    outEvtChannel="fullssl">
    outName="username"
    outPasswdExpiryStatus="None"
    outPasswdExpiryDuration="1"
</aaaRefresh>
```

aaaTokenLogin

The `aaaTokenLogin` method allows access to the user based on the token passed. These tokens authenticate the user instead of using the password to allow access to the system. Tokens are generated by `aaaGetNComputeAuthToken` method.

Request Syntax

```
<xs:element name="aaaTokenLogin" type="aaaTokenLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaTokenLogin" mixed="true">
            <xs:attribute name="inName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inToken">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaTokenLogin" type="aaaTokenLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaTokenLogin" mixed="true">
            <xs:attribute name="outCookie" type="xs:string"/>
            <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
            <xs:attribute name="outPriv">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:pattern
value="((ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equipment|ls-con
fig-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security|ls-config|
ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod-config|ls-
server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-security|ls-netwo
rk-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-network|ls-ext-ac
cess|fault),){0,35}(ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equip
ment|ls-config-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security
|ls-config|ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod
-config|ls-server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-securi
ty|ls-network-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-networ
k|ls-ext-access|fault){0,1}"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outSessionId">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="32"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outVersion" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaTokenLogin
    inName="admin"
    inToken="80278502964410805791351" />
```

Response

```
<aaaTokenLogin cookie=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin,read-only"
    outDomains=""
    outChannel="noencssl"
    outEvtChannel="noencssl"
    outSessionId="web_49374_A"
    outVersion="1.4(0.61490)">
</aaaTokenLogin>
```

aaaTokenRefresh

The aaaTokenRefresh method refreshes the current TokenLogin session.

Note When the password expiry feature is enabled and of if the user password is expired, the aaaTokenRefresh API indicates the expiry of password in the failure XML API response.

Request Syntax

```
<xs:element name="aaaTokenRefresh" type="aaaTokenRefresh"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaTokenRefresh" mixed="true">
            <xs:attribute name="inName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inCookie" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="aaaTokenRefresh" type="aaaTokenRefresh"
substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaTokenRefresh" mixed="true">
            <xs:attribute name="outCookie" type="xs:string"/>
            <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
            <xs:attribute name="outPriv">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:pattern
value="((ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equipment|ls-con
fig-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security|ls-config|
ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod-config|ls-
server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-security|ls-netwo
rk-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-network|ls-ext-ac
cess|fault),){0,35}(ext-lan-policy|pn-maintenance|ls-security-policy|pod-security|pn-equip
ment|ls-config-policy|ext-san-policy|ls-security|aaa|power-mgmt|read-only|ext-lan-security
|ls-config|ls-server-policy|pod-qos|pn-policy|ls-storage-policy|admin|ext-san-security|pod
-config|ls-server|ext-lan-qos|ls-storage|ls-qos-policy|operations|ext-lan-config|pn-securi
ty|ls-network-policy|pod-policy|ext-san-qos|ls-qos|ls-server-oper|ext-san-config|ls-networ
k|ls-ext-access|fault){0,1}"/>
                </xs:restriction>
            </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaTokenRefresh
    inName="admin"
    inCookie="<real_cookie>" />
```

Response

```
<aaaTokenRefresh
    cookie=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin,read-only"
    outDomains=""
    outChannel="noencssl"
    outEvtChannel="noencssl">
</aaaTokenRefresh>
```

`configCheckConformance`

The `configCheckConformance` method checks if the given distributable (firmware package) can be used against the running Cisco UCS Manager version.

Request Syntax

```
<xs:element name="configCheckConformance" type="configCheckConformance"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configCheckConformance" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configCheckConformance" type="configCheckConformance"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configCheckConformance" mixed="true">
            <xs:all>
                <xs:element name="outConfDns" type="dnSet" minOccurs="0"/>
                <xs:element name="outToResetDns" type="dnSet" minOccurs="0"/>
                <xs:element name="outNonConfDns" type="dnSet" minOccurs="0"/>
                <xs:element name="outInProgressDns" type="dnSet" minOccurs="0"/>
                <xs:element name="outNonUpgradableDns" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<configCheckConformance
    dn="sys/fw-catalogue/distrib-ucs-k9-bundle-b-series.2.0.0.528.gbin"
    cookie="<real_cookie>">
</configCheckConformance>
```

Response

```
<configCheckConformance
    dn="sys/fw-catalogue/distrib-ucs-k9-bundle-b-series.2.0.0.528.gbin"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfDns>
       <dn value="sys/chassis-1/blade-5/mgmt/fw-system"/>
       <dn value="sys/chassis-1/blade-5/bios/fw-boot-loader"/>
       <dn value="sys/chassis-1/blade-3/boardController/mgmt/fw-system"/>
    </outConfDns>
    <outToResetDns>
    </outToResetDns>
    <outNonConfDns>
       <dn value="sys/chassis-1/blade-1/mgmt/fw-system"/>
```

```
        <dn value="sys/chassis-1/blade-3/mgmt/fw-system"/>
        <dn value="sys/chassis-1/blade-1/bios/fw-boot-loader"/>
        <dn value="sys/chassis-1/blade-3/bios/fw-boot-loader"/>
        <dn value="sys/chassis-1/blade-3/adaptor-1/mgmt/fw-system"/>
        <dn value="sys/chassis-1/blade-1/adaptor-2/mgmt/fw-system"/>
        <dn value="sys/chassis-1/blade-1/adaptor-1/mgmt/fw-system"/>
        <dn value="sys/chassis-1/blade-3/adaptor-2/mgmt/fw-system"/>
        <dn value="sys/chassis-1/blade-5/adaptor-1/mgmt/fw-system"/>
        <dn value="sys/chassis-1/blade-3/board/storage-SAS-1/fw-system"/>
    </outNonConfDns>
    <outInProgressDns>
    </outInProgressDns>
    <outNonUpgradableDns>
    </outNonUpgradableDns>
</configCheckConformance>
```

configCheckFirmwareUpdatable

The `configCheckFirmwareUpdatable` method checks if firmware in certain components can be updated or activated. The method is triggered every time a user initiates an update or activate process.

For example, if a user tries to update the firmware version of an endpoint for which a firmware policy is specified as part of a service profile (either a host firmware pack or management firmware pack), the operation is disallowed. This method performs the validation.

Request Syntax

```
<xs:element name="configCheckFirmwareUpdatable" type="configCheckFirmwareUpdatable"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configCheckFirmwareUpdatable" mixed="true">
            <xs:all>
                <xs:element name="inUpdatableDns" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configCheckFirmwareUpdatable" type="configCheckFirmwareUpdatable"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configCheckFirmwareUpdatable" mixed="true">
            <xs:all>
                <xs:element name="outPassDns" type="dnSet" minOccurs="0"/>
                <xs:element name="outFailDns" type="dnSet" minOccurs="0"/>
                <xs:element name="outInvalidDns" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<configCheckFirmwareUpdatable
    cookie="<real_cookie>">
    <inUpdatableDns>
       <dn value="sys/chassis-1/blade-5/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-5/adaptor-2/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-2/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-2/adaptor-1/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-1/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-3/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-3/adaptor-2/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-1/adaptor-1/mgmt/fw-updatable"/>
    </inUpdatableDns>
</configCheckFirmwareUpdatable>
```

Response

```
<configCheckFirmwareUpdatable
    cookie="<real_cookie>"
    response="yes">
    <outPassDns>
       <dn value="sys/chassis-1/blade-1/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/bla de-2/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-3/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-5/mgmt/fw-updatable"/>
    </outPassDns>
    <outFailDns>
       <dn value="sys/chassis-1/blade-5/adaptor-2/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-2/adaptor-1/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade- 1/adaptor-1/mgmt/fw-updatable"/>
       <dn value="sys/chassis-1/blade-3/adaptor-2/mgmt/fw-updatable"/>
    </outFailDns>
    <outInvalidDns>
    </outInvalidDns>
</configCheckFirmwareUpdatable>
```

configConfFiltered

The `configConfFiltered` method limits data and activity according to the configured policies.

Request Syntax

```
<xs:element name="configConfFiltered" type="configConfFiltered"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfFiltered" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
                <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
```

```
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:enumeration value="no"/>
                                <xs:enumeration value="yes"/>
                            </xs:restriction>
                        </xs:simpleType>
                   </xs:union>
               </xs:simpleType>
           </xs:attribute>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="classId" type="namingClassId"/>
       </xs:complexType>
```

Response Syntax

```
<xs:element name="configConfFiltered" type="configConfFiltered"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfFiltered" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Examples

Request

```
<configConfFiltered
      cookie="<real_cookie>"
      inHierarchical="false"
      classId="orgOrg">
      <inFilter>
            <eq class="orgOrg"
            property="name"
            value="root" />
      </inFilter>
</configConfFiltered>
```

Response

```
<configConfFiltered
    cookie="<real_cookie>"
    commCookie="5/15/0/617"
    srcExtSys="10.193.33.206"
    destExtSys="10.193.33.206"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes"
    classId="orgOrg">
    <outConfigs>
       <orgDatacenter
```

```
        descr="HR (Human Resources- new Descr)"
        dn="org-root/org-Cisco/org-HR"
        fltAggr="0"
        level="2"
        name="HR"
        status="modified"/>
    </outConfigs>
</configConfFiltered>
```

configConfMo

The `configConfMo` method configures the specified managed object in a single subtree (for example, DN).

Request Syntax

```
<xs:element name="configConfMo" type="configConfMo" substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMo" mixed="true">
            <xs:all>
                <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configConfMo" type="configConfMo" substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMo" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<configConfMo
    dn=""
    cookie="<real_cookie>"
    inHierarchical="false">
    <inConfig>
       <aaaLdapEp
          attribute="CiscoAvPair"
          basedn="dc=pasadena,dc=cisco,dc=com"
          descr=""
          dn="sys/ldap-ext"
          filter="sAMAccountName=$userid"
          retries="1"
          status="modified"
          timeout="30"/>
    </inConfig>
</configConfMo>
```

Response

```
<configConfMo
    dn=""
    cookie="<real_cookie>"
    commCookie="11/15/0/28"
    srcExtSys="10.193.33.101"
    destExtSys="10.193.33.101"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes">
    <outConfig>
       <aaaLdapEp
          attribute="CiscoAvPair"
          basedn="dc=pasadena,dc=cisco,dc=com"
          childAction="deleteNonPresent"
          descr=""
          dn="sys/ldap-ext"
          filter="sAMAccountName=$userid"
          fsmDescr=""
          fsmPrev="updateEpSuccess"
          fsmProgr="100"
          fsmStageDescr=""
          fsmStamp="2010-11-22T23:41:01.826"
          fsmStatus="nop"
          fsmTry="0"
          intId="10027"
          name=""
          retries="1"
          status="modified"
          timeout="30"/>
    </outConfig>
</configConfMo>
```

configConfMoGroup

The `configConfMoGroup` method configures groups of managed objects based upon the configured policies.

Request Syntax

```
<xs:element name="configConfMoGroup" type="configConfMoGroup"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMoGroup" mixed="true">
            <xs:all>
                <xs:element name="inDns" type="dnSet" minOccurs="0"/>
                <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configConfMoGroup" type="configConfMoGroup"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMoGroup" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Note The descr property of `orgDataCenter` (under `org-root/org-Cisco` and `org-root/org-Soda`) is modified. Because the descr property is not implicit, it can be modified. If implicit, the modification does not apply and a new `orgDataCenter` is created.

Request

```
<configConfMoGroup
    cookie="<real_cookie>"
    inHierarchical="false">
    <inDns>
       <dn value="org-root/org-Cisco" />
       <dn value="org-root/org-Soda" />
    </inDns>
    <inConfig>
       <orgDatacenter dn="org-HR" descr="HR (Human Resources)"/>
    </inConfig>
</configConfMoGroup>
```

Response

```
<configConfMoGroup
    cookie="<real_cookie>"
    commCookie="5/15/0/600"
    srcExtSys="10.193.33.206"
    destExtSys="10.193.33.206"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outConfigs>
       <orgDatacenter
          descr="HR (Human Resources)"
          dn="org-root/org-Soda/org-HR"
          fltAggr="0"
          level="2"
          name="HR"
          status="modified"/>
       <orgDatacenter
          descr="HR (Human Resources)"
          dn="org-root/org-Cisco/org-HR"
          fltAggr="0"
          level="2"
          name="HR"
          status="modified"/>
    </outConfigs>
</configConfMoGroup>
```

configConfMos

The `configConfMos` method configures managed objects in multiple subtrees using DNs.

Request Syntax

```
<xs:element name="configConfMos" type="configConfMos" substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMos" mixed="true">
            <xs:all>
                <xs:element name="inConfigs" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_2">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configConfMos" type="configConfMos" substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMos" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_5">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
       <pair key="org-root/logprof-default">
          <policyLogProfile dn="org-root/logprof-default"
          name="default"
          level="debug1"
          size="10000000"
          backupCount="4"/>
       </pair>

 <!-- Update Controller Device Profile -->
       <pair key="org-root/controller-profile-default">
            <policyControllerDeviceProfile
               dn="org-root/controller-profile-default"
               adminState="enabled">
               .
                <commDnsProvider hostip="171.70.168.183" order="1"/>
                <commDnsProvider hostip="171.68.226.120" order="2"/>
                <commDnsProvider hostip="64.102.6.247"   order="3"/>
            </policyControllerDeviceProfile>
        </pair>
    </inConfigs>
</configConfMos>
```

Response

```
<configConfMos
    cookie="<real_cookie>"
    commCookie="7/15/0/1a74"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes">
    <outConfigs>
       <pair key="org-root/logprof-default">
          <policyLogProfile
             adminState="enabled"
             backupCount="4"
             descr="the log level for every process"
             dn="org-root/logprof-default"
             intId="10065"
             level="debug1"
             name="default"
             size="10000000"/>
       </pair>
       <pair key="org-root/controller-profile-default">
          .
          ./>
       </pair>
    </outConfigs>
</configConfMos>
```

configConfRename

The `configConfRename` method is used to rename a service or chassis profile.

Request Syntax

```
<xs:element name="configConfRename" type="configConfRename"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configConfRename" mixed="true">
           <xs:attribute name="inNewName" type="xs:string"/>
           <xs:attribute name="inHierarchical">
                <xs:simpleType>
                     <xs:union memberTypes="xs:boolean">
                          <xs:simpleType>
                               <xs:restriction base="xs:string">
                                    <xs:enumeration value="no"/>
                                    <xs:enumeration value="yes"/>
                               </xs:restriction>
                          </xs:simpleType>
                     </xs:union>
                </xs:simpleType>
           </xs:attribute>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
```

```
          <xs:attribute name="dn" type="referenceObject"/>
      </xs:complexType>
```

Response Syntax

```
<xs:element name="configConfRename" type="configConfRename"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configConfRename" mixed="true">
           <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
           </xs:all>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="errorCode" type="xs:unsignedInt"/>
           <xs:attribute name="errorDescr" type="xs:string"/>
           <xs:attribute name="invocationResult" type="xs:string"/>
           <xs:attribute name="dn" type="referenceObject"/>
      </xs:complexType>
```

Examples

Request

```
<configConfRename
dn="org-root/cp-TestnewName"
cookie="<real_cookie>"
inNewName="testCP">
</configConfRename>
```

Response

```
<configConfRename dn="org-root/cp-TestnewName"
cookie="<real_cookie>" response="yes">
<outConfig>
<equipmentChassisProfile assignState="unassigned" assocState="unassociated"
chassisDn="" chassisFwPolicyName="" configQualifier="" configState="not-applied"
descr="" diskZoningPolicyName="" dn="org-root/cp-testCP" fltAggr="0" fsmDescr=""
fsmFlags="" fsmPrev="ConfigureSuccess" fsmProgr="100" fsmRmtInvErrCode="none"
fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="2016-08-31T21:33:09.959"
fsmStatus="nop" fsmTry="0" intId="6090789" maintPolicyName="" name="testCP"
operChassisFwPolicyName="org-root/fw-chassis-pack-default"
operDiskZoningPolicyName="org-root/disk-zoning-policy-default"
operMaintPolicyName="org-root/chassis-profile-maint-default" operSrcTemplName=""
operState="unassociated" owner="management" policyLevel="0" policyOwner="local"
propAcl="0" resolveRemote="yes" srcTemplName="" status="created" type="instance"
usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
</outConfig>
</configConfRename>
```

configCountClass

The `configCountClass` method is used to know the number of instances of a class with or without a filter.

Request Syntax

```
<xs:element name="configCountClass" type="configCountClass"
substitutionGroup="externalMethod"/>
```

```
      <xs:complexType name="configCountClass" mixed="true">
           <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
           </xs:all>
           <xs:attribute name="inHierarchical">
                <xs:simpleType>
                     <xs:union memberTypes="xs:boolean">
                          <xs:simpleType>
                               <xs:restriction base="xs:string">
                                    <xs:enumeration value="no"/>
                                    <xs:enumeration value="yes"/>
                               </xs:restriction>
                          </xs:simpleType>
                     </xs:union>
                </xs:simpleType>
           </xs:attribute>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="classId" type="namingClassId"/>
      </xs:complexType>
```

Response Syntax

```
<xs:element name="configCountClass" type="configCountClass"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configCountClass" mixed="true">
           <xs:attribute name="outCount" type="xs:unsignedLong"/>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="errorCode" type="xs:unsignedInt"/>
           <xs:attribute name="errorDescr" type="xs:string"/>
           <xs:attribute name="invocationResult" type="xs:string"/>
           <xs:attribute name="classId" type="namingClassId"/>
      </xs:complexType>
```

Examples

Request

```
<configCountClass
classId="computeItem"
cookie="<real cookie>">
 </configCountClass>
```

Response

```
<configCountClass cookie="<real cookie>"
response="yes"
classId="computeItem"
outCount="2">
</configCountClass>
```

`configEstimateConfMos`

The `configEstimateConfMos` method

Request Syntax

```
<xs:element name="configEstimateConfMos" type="configEstimateConfMos"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configEstimateConfMos" mixed="true">
           <xs:all>
                <xs:element name="inConfigs" type="configMap" minOccurs="0">
                     <xs:unique name="unique_map_key_2">
                          <xs:selector xpath="pair"/>
                          <xs:field xpath="@key"/>
                     </xs:unique>
                </xs:element>
           </xs:all>
           <xs:attribute name="inHierarchical">
                <xs:simpleType>
                     <xs:union memberTypes="xs:boolean">
                          <xs:simpleType>
                               <xs:restriction base="xs:string">
                                    <xs:enumeration value="no"/>
                                    <xs:enumeration value="yes"/>
                               </xs:restriction>
                          </xs:simpleType>
                     </xs:union>
                </xs:simpleType>
           </xs:attribute>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
      </xs:complexType>
```

Response Syntax

```
<xs:element name="configEstimateConfMos" type="configEstimateConfMos"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configEstimateConfMos" mixed="true">
           <xs:all>
                <xs:element name="outConfigs" type="configMap" minOccurs="0">
                     <xs:unique name="unique_map_key_6">
                          <xs:selector xpath="pair"/>
                          <xs:field xpath="@key"/>
                     </xs:unique>
                </xs:element>
           </xs:all>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="errorCode" type="xs:unsignedInt"/>
           <xs:attribute name="errorDescr" type="xs:string"/>
           <xs:attribute name="invocationResult" type="xs:string"/>
      </xs:complexType>
```

Examples

Request

Response

configEstimateImpact

The `configEstimateImpact` method estimates the impact of a set of managed objects modifications in terms of disruption of running services. For example, modifying the UUID pool used by an updating template might require rebooting servers associated to service profiles instantiated from the template.

User can estimate the impact of a change set by passing the set to the method and inspecting the output parameters. Output parameters are a set of affected service profiles (before and after the changes) and the corresponding ack object for each service profile.

Ack objects contain the following information:

- Whether the changes are disruptive (for example, require reboot of the server associated to the service profile).
- Summary of changes.
- When changes are applied (immediately, after user ack, during scheduled occurrence of a maintenance window).
- Date and time at which such changes were made and by whom.

Cisco UCS returns the ack objects before and after the changes are applied. This information helps determine whether some changes were already pending on the service profile. This condition can occur when maintenance policies are used.

The parameters are defined as:

- configs—Set of changes to be evaluated (add, delete, or modify managed objects).
- affected—Affected service profiles after the changes have been applied (not hierarchical).
- oldAffected—Affected service profiles before applying changes (not hierarchical).
- ackables—Content of the ack object associated to the service profiles, after applying the changes.
- oldAckables—Content of the ack object associated to the service profiles, before applying the changes.

Request Syntax

```
<xs:element name="configEstimateImpact" type="configEstimateImpact"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configEstimateImpact" mixed="true">
            <xs:all>
                <xs:element name="inConfigs" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_3">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configEstimateImpact" type="configEstimateImpact"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configEstimateImpact" mixed="true">
            <xs:all>
                <xs:element name="outAckables" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_6">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
                <xs:element name="outOldAckables" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_7">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
                <xs:element name="outAffected" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_8">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
                <xs:element name="outOldAffected" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_9">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<configEstimateImpact
    cookie="<real_cookie>">
    <inConfigs>
       <pair key="org-root/ls-template-3">
          <lsServer
             agentPolicyName=""
             biosProfileName=""
             bootPolicyName=""
             descr=""
             dn="org-root/ls-template-3"
             dynamicConPolicyName=""
             extIPState="none"
             hostFwPolicyName=""
             identPoolName="default"
             localDiskPolicyName=""
             maintPolicyName=""
             mgmtAccessPolicyName=""
             mgmtFwPolicyName=""
```

```
            name="template-3"
            powerPolicyName="default"
            scrubPolicyName=""
            solPolicyName=""
            srcTemplName=""
            statsPolicyName="default"
            status="created"
            type="updating-template"
            usrLbl=""
            uuid="derived"
            vconProfileName=""/>
      </pair>
    </inConfigs>
</configEstimateImpact>
```

Response

```
<configEstimateImpact
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outAckables>
    </outAckables>
    <outOldAckables>
    </outOldAckables>
    <outAffected>
       <pair key="org-root/ls-template-3">
          <lsServer
            agentPolicyName=""
            assignState="unassigned"
            assocState="unassociated"
            biosProfileName=""
            bootPolicyName=""
            configQualifier=""
            configState="not-applied"
            descr=""
            dn="org-root/ls-template-3"
            dynamicConPolicyName=""
            extIPState="none"
            fltAggr="0"
            hostFwPolicyName=""
            identPoolName="default"
            intId="52359"
            localDiskPolicyName=""
            maintPolicyName=""
            mgmtAccessPolicyName=""
            mgmtFwPolicyName=""
            name="template-3"
            operBiosProfileName=""
            operBootPolicyName="org-root/boot-policy-default"
            operDynamicConPolicyName=""
            operHostFwPolicyName=""
            operIdentPoolName="org-root/uuid-pool-default"
            operLocalDiskPolicyName="org-root/local-disk-config-default"
            operMaintPolicyName="org-root/maint-default"
            operMgmtAccessPolicyName=""
            operMgmtFwPolicyName=""
            operPowerPolicyName="org-root/power-policy-default"
            operScrubPolicyName="org-root/scrub-default"
            operSolPolicyName=""
            operSrcTemplName=""
            operState="unassociated"
            operStatsPolicyName="org-root/thr-policy-default"
            operVconProfileName=""
            owner="management"
            pnDn=""
            powerPolicyName="default"
            scrubPolicyName=""
            solPolicyName=""
            srcTemplName=""
            statsPolicyName="default"
            status="created"
            type="updating-template"
            usrLbl=""
            uuid="derived"
            uuidSuffix="0000-000000000000"
            vconProfileName=""/>
      </pair>
    </outAffected>
    <outOldAffected>
      <pair key="org-root/ls-template-3">
         <lsServer
          .
         ./>
      </pair>
    </outOldAffected>
</configEstimateImpact>
```

configFindDependencies

The `configFindDependencies` method returns the device policy details for a specified policy.

Request Syntax

```
<xs:element name="configFindDependencies" type="configFindDependencies"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configFindDependencies" mixed="true">
            <xs:attribute name="inReturnConfigs">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configFindDependencies" type="configFindDependencies"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configFindDependencies" mixed="true">
        <xs:all>
            <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
        </xs:all>
        <xs:attribute name="outHasDep">
            <xs:simpleType>
                <xs:union memberTypes="xs:boolean">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:enumeration value="no"/>
                            <xs:enumeration value="yes"/>
                        </xs:restriction>
                    </xs:simpleType>
                </xs:union>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
        <xs:attribute name="dn" type="referenceObject"/>
    </xs:complexType>
```

Examples

Request

```
<configFindDependencies
    dn="org-root/fw-host-pack-host-pack-6625"
    cookie="<real_cookie>"
    inReturnConfigs="yes">
</configFindDependencies>
```

Response

```
<configFindDependencies
    dn="org-root/fw-host-pack-host-pack-6625"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr=""
    outHasDep="yes">
    <outConfigs>
       <lsServer
          agentPolicyName=""
          assignState="assigned"
          assocState="associated"
          biosProfileName=""
          bootPolicyName=""
          configQualifier=""
          configState="applied"
          descr=""
          dn="org-root/ls-service-profile-5"
          dynamicConPolicyName=""
          extIPState="none"
          fltAggr="0"
          fsmDescr=""
          fsmFlags=""
          fsmPrev="ConfigureSuccess"
          fsmProgr="100"
```

```
            fsmRmtInvErrCode="none"
            fsmRmtInvErrDescr=""
            fsmRmtInvRslt=""
            fsmStageDescr=""
            fsmStamp="2011-01-10T23:51:28.310"
            fsmStatus="nop"
            fsmTry="0"
            hostFwPolicyName="host-pack-6625"
            identPoolName=""
            intId="29191" localDiskPolicyName=""
            maintPolicyName=""
            mgmtAccessPolicyName=""
            mgmtFwPolicyName="m-firmware-1"
            name="service-profile-5"
            operBiosProfileName=""
            operBootPolicyName="org-root/boot-policy-default"
            operDynamicConPolicyName=""
            operHostFwPolicyName="org-root/fw-host-pack-host-pack-6625"
            operIdentPoolName="org-root/uuid-pool-default"
            operLocalDiskPolicyName="org-root/local-disk-config-default"
            operMaintPolicyName="org-root/maint-default"
            operMgmtAccessPolicyName=""
            operMgmtFwPolicyName="org-root/fw-mgmt-pack-m-firmware-1"
            operPowerPolicyName="org-root/power-policy-default"
            operScrubPolicyName="org-root/scrub-default"
            operSolPolicyName=""
            operSrcTemplName=""
            operState="ok"
            operStatsPolicyName="org-root/thr-policy-default"
            operVconProfileName=""
            owner="management"
            pnDn="sys/chassis-1/blade-5"
            powerPolicyName="default"
            scrubPolicyName=""
            solPolicyName=""
            srcTemplName=""
            statsPolicyName="default"
            type="instance"
            usrLbl=""
            uuid="derived"
            uuidSuffix="0000-000000000000"
            vconProfileName=""/>
    </outConfigs>
</configFindDependencies>
```

configFindDnsByClassId

The `configFindDnsByClassId` method finds distinguished names and returns them sorted by class ID.

Request Syntax

```
<xs:element name="configFindDnsByClassId" type="configFindDnsByClassId"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configFindDnsByClassId" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configFindDnsByClassId" type="configFindDnsByClassId"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configFindDnsByClassId" mixed="true">
            <xs:all>
                <xs:element name="outDns" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Examples

Request

```
<configFindDnsByClassId
    classId="computeItem"
    cookie="<real_cookie>" />
```

Response

```
<configFindDnsByClassId
    cookie="<real_cookie>"
    response="yes"
    classId="computeItem">
    <outDns>
       <dn value="sys/chassis-1/blade-7"/>
       <dn value="sys/chassis-1/blade-5"/>
       <dn value="sys/chassis-1/blade-3"/>
       <dn value="sys/chassis-1/blade-1"/>
    </outDns>
</configFindDnsByClassId>
```

`configFindHostPackDependencies`

The `configFindHostPackDependencies` method is used to find the policies or profiles referring to the host packs.

Request Syntax

```
<xs:element name="configFindHostPackDependencies"
type="configFindHostPackDependencies" substitutionGroup="externalMethod"/>
      <xs:complexType name="configFindHostPackDependencies" mixed="true">
           <xs:all>
                <xs:element name="inHostPackDns" type="dnSet" minOccurs="0"/>
```

```
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configFindHostPackDependencies"
type="configFindHostPackDependencies" substitutionGroup="externalMethod"/>
      <xs:complexType name="configFindHostPackDependencies" mixed="true">
           <xs:all>
                <xs:element name="outConfigSet" type="configSet" minOccurs="0"/>
           </xs:all>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="errorCode" type="xs:unsignedInt"/>
           <xs:attribute name="errorDescr" type="xs:string"/>
           <xs:attribute name="invocationResult" type="xs:string"/>
           <xs:attribute name="dn" type="referenceObject"/>
      </xs:complexType>
```

Examples

Request

```
<configFindHostPackDependencies
dn="sys/fw-catalogue/distrib-ucs-k9-bundle-b-series.2.0.0.528.gbin"
cookie="<real cookie>">
</configFindHostPackDependencies>
```

Response

```
<configFindHostPackDependencies
dn="sys/fw-catalogue/distrib-ucs-k9-bundle-b-series.2.0.0.528.gbin"
cookie="<real cookie>"
response="yes"> <outConfigSet>
</outConfigSet>
</configFindHostPackDependencies>
```

configFindPermitted

The `configFindPermitted` method finds the managed objects of a specific org and class ID.

Request Syntax

```
<xs:element name="configFindPermitted" type="configFindPermitted"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configFindPermitted" mixed="true">
           <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
           </xs:all>
           <xs:attribute name="inClassId" type="namingClassId"/>
           <xs:attribute name="inHierarchical">
                <xs:simpleType>
                     <xs:union memberTypes="xs:boolean">
                          <xs:simpleType>
                               <xs:restriction base="xs:string">
                                    <xs:enumeration value="no"/>
                                    <xs:enumeration value="yes"/>
                               </xs:restriction>
                          </xs:simpleType>
                     </xs:union>
                </xs:simpleType>
           </xs:attribute>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="dn" type="referenceObject"/>
      </xs:complexType>
```

Response Syntax

```
<xs:element name="configFindPermitted" type="configFindPermitted"
substitutionGroup="externalMethod"/>
      <xs:complexType name="configFindPermitted" mixed="true">
           <xs:all>
                <xs:element name="outConfigSet" type="configSet" minOccurs="0"/>
           </xs:all>
           <xs:attribute name="cookie" type="xs:string"/>
           <xs:attribute name="response" type="YesOrNo"/>
           <xs:attribute name="errorCode" type="xs:unsignedInt"/>
           <xs:attribute name="errorDescr" type="xs:string"/>
           <xs:attribute name="invocationResult" type="xs:string"/>
           <xs:attribute name="dn" type="referenceObject"/>
      </xs:complexType>
```

Examples

Request

```
<configFindPermitted
dn="org-root"
cookie="<real cookie>">
</configFindPermitted>
```

Response

```
<configFindPermitted
dn="org-root"
cookie="real cookie" response="yes">
<outConfigSet>
</outConfigSet>
</configFindPermitted>
```

`configGetRemotePolicies`

The `configGetRemotePolicies` method finds the list of policies from the policy manager of UCS central.

Request Syntax

```
<xs:element name="configGetRemotePolicies" type="configGetRemotePolicies"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configGetRemotePolicies" mixed="true">
          <xs:all>
               <xs:element name="inPolicyDigests" type="configSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inContext" type="referenceObject"/>
          <xs:attribute name="inStimulusId" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configGetRemotePolicies" type="configGetRemotePolicies"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configGetRemotePolicies" mixed="true">
          <xs:all>
               <xs:element name="outPolicies" type="configSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="outStimulusId" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

```
<configGetRemotePolicies
cookie="<real cookie>">
</configGetRemotePolicies>
```

Response

```
<configGetRemotePolicies
cookie="<real cookie>"
response="yes"
outStimulusId="0">
<outPolicies> </outPolicies>
</configGetRemotePolicies>
```

`configInstallAllImpact`

The `configInstallAllImpact` method is used to display a confirmation message about the impacts while applying a policy.

Request Syntax

```
<xs:element name="configInstallAllImpact" type="configInstallAllImpact"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configInstallAllImpact" mixed="true">
          <xs:all>
               <xs:element name="inHostPackDns" type="dnSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inInfraPackVersion">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inBladePackVersion">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inRackPackVersion">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inMSeriesPackVersion">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configInstallAllImpact" type="configInstallAllImpact"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configInstallAllImpact" mixed="true">
          <xs:all>
               <xs:element name="outConfigSet" type="configSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

```
<configInstallAllImpact
cookie="<real cookie>">
</configInstallAllImpact>
```

Response

```
<configInstallAllImpact
dn="" cookie="<real cookie>"
response="yes"
errorCode="552"
invocationResult="service-unavailable"
errorDescr="Authorization required">
</configInstallAllImpact>
```

configMoChangeEvent

The `configMoChangeEvent` method provides event details from Cisco UCS as a result of event subscription. The status property indicates the action that caused the event (indicated by inEid) to be generated. This is a request sent from Cisco UCS to the subscribers. There is no response.

Request Syntax

```
<xs:element name="configMoChangeEvent" type="configMoChangeEvent"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configMoChangeEvent" mixed="true">
            <xs:all>
                <xs:element name="inConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inEid" type="xs:unsignedLong"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configMoChangeEvent" type="configMoChangeEvent"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configMoChangeEvent" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<configMoChangeEvent
    cookie="<real_cookie>"
    inEid="174712">
    <inConfig>
       <callhomeEp
          dn="call-home"
          fsmPrev="configCallhomeSetLocal"
          fsmStamp="2008-10-16T17:59:25"
          fsmTry="11"
          status="modified"/>
    </inConfig>
</configMoChangeEvent>

<configMoChangeEvent
    cookie="<real_cookie>"
    inEid="174713">
    <inConfig>
       <mgmtIf
          dn="sys/switch-A/mgmt/if-1"
          fsmPrev="SwMgmtOobIfConfigSwitch"
          fsmStamp="2008-10-16T17:59:25"
          fsmTry="9"
          status="modified"/>
    </inConfig>
</configMoChangeEvent>

<configMoChangeEvent
    cookie="<real_cookie>"
    inEid="174714">
    <inConfig>
       <eventRecord
          affected="sys/sysdebug/file-export"
          cause="transition"
          created="2008-10-16T17:59:25"
          descr="[FSM:STAGE:RETRY:8]: configuring automatic core file export service on
            local"
          dn="event-log/54344"
          id="54344"
          ind="state-transition"
          severity="info"
          status="created"
          trig="special"
          txId="24839"
          user="internal"/>
    </inConfig>
</configMoChangeEvent>
```

Response

There is no response to this method.

configRefreshIdentity

The `configRefreshIdentity` method

Request Syntax

```
<xs:element name="configRefreshIdentity" type="configRefreshIdentity"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configRefreshIdentity" mixed="true">
          <xs:attribute name="inIdType">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="mac"/>
                         <xs:enumeration value="wwnn"/>
                         <xs:enumeration value="wwpn"/>
                         <xs:enumeration value="uuid"/>
                         <xs:enumeration value="vlan"/>
                         <xs:enumeration value="ipV4"/>
                         <xs:enumeration value="ipV6"/>
                         <xs:enumeration value="iqn"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
```

```
                                <xs:enumeration value="no"/>
                                <xs:enumeration value="yes"/>
                           </xs:restriction>
                      </xs:simpleType>
                 </xs:union>
            </xs:simpleType>
       </xs:attribute>
       <xs:attribute name="inIsDiscardMode">
            <xs:simpleType>
                 <xs:union memberTypes="xs:boolean">
                      <xs:simpleType>
                           <xs:restriction base="xs:string">
                                <xs:enumeration value="no"/>
                                <xs:enumeration value="yes"/>
                           </xs:restriction>
                      </xs:simpleType>
                 </xs:union>
            </xs:simpleType>
       </xs:attribute>
       <xs:attribute name="cookie" type="xs:string"/>
       <xs:attribute name="response" type="YesOrNo"/>
       <xs:attribute name="dn" type="referenceObject"/>
  </xs:complexType>
```

Response Syntax

```
<xs:element name="configRefreshIdentity" type="configRefreshIdentity"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configRefreshIdentity" mixed="true">
          <xs:all>
               <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
               <xs:element name="outAckables" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_11">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
               </xs:element>
               <xs:element name="outOldAckables" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_12">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
               </xs:element>
               <xs:element name="outAffected" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_13">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
               </xs:element>
               <xs:element name="outOldAffected" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_14">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
               </xs:element>
          </xs:all>
          <xs:attribute name="outRetry" type="xs:unsignedShort"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

Response

configReleaseResolveContext

The `configReleaseResolveContext` method resolves the next results for the current context.

Request Syntax

```
<xs:element name="configReleaseResolveContext" type="configReleaseResolveContext"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configReleaseResolveContext" mixed="true">
          <xs:attribute name="inContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configReleaseResolveContext" type="configReleaseResolveContext"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configReleaseResolveContext" mixed="true">
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

```
<configReleaseResolveContext
inContext=1
cookie="<real cookie>">
</configReleaseResolveContext>
```

Response

```
<configReleaseResolveContext
cookie="<real cookie>"
response="yes">
</configReleaseResolveContext>
```

configRenewResolveContext

The `configRenewResolveContext` method renews an active resolve context. This method returns a new context

Request Syntax

```
<xs:element name="configRenewResolveContext" type="configRenewResolveContext"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configRenewResolveContext" mixed="true">
          <xs:attribute name="inContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configRenewResolveContext" type="configRenewResolveContext"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configRenewResolveContext" mixed="true">
          <xs:attribute name="outContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

```
<configRenewResolveContext
inContext=1
cookie="<real cookie>">
</configRenewResolveContext>
```

Response

```
<configRenewResolveContext
cookie="<real cookie>"
response="yes" outContext="0">
</configRenewResolveContext>
```

configResolveChildren

The `configResolveChildren` method retrieves children of managed objects under a specific DN in the managed information tree. A filter can be used to reduce the number of children being returned.

Request Syntax

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveChildren" mixed="true">
            <xs:all>
```

```
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inDn" type="referenceObject"/>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveChildren" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveChildren
    cookie="<real_cookie>"
    classId="aaaUser"
    inDn="sys/user-ext"
    inHierarchical="false">
    <inFilter>
    </inFilter>
</configResolveChildren>
```

Response

```
<configResolveChildren
    cookie="<real_cookie>"
    commCookie="11/15/0/2a59"
    srcExtSys="10.193.33.120"
    destExtSys="10.193.33.120"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes"
    classId="aaaUser">
    <outConfig>
       <aaaUser descr=""  dn="sys/user-ext/user-chambers"
          email="" expiration="never" expires="no" firstName="John" intId="12716"
          lastName="Chambers" name="chambers" phone="" priv="admin,read-only"
          pwdSet="yes"/>
      <aaaUser descr="" dn="sys/user-ext/user-jackson" email="" expiration="never"
          expires="no" firstName="Andrew" intId="12734" lastName="Jackson"
          name="jackson" phone=""
          priv="fault,operations,policy,read-only,res-config,tenant" pwdSet="yes"/>
      <aaaUser descr="" dn="sys/user-ext/user-admin" email="" expiration="never"
          expires="no" firstName="" intId="10052" lastName="" name="admin" phone=""
          priv="admin,read-only" pwdSet="yes"/>
      <aaaUser descr="" dn="sys/user-ext/user-bama" email="" expiration="never"
          expires="no" firstName="Rak" intId="12711" lastName="Bama" name="bama"
          phone="" priv="fault,operations,policy,read-only,res-config,tenant"
          pwdSet="yes"/>
      <aaaUser descr="" dn="sys/user-ext/user-fuld" email="" expiration="never"
          expires="no" firstName="Richard" intId="12708" lastName="Fuld" name="fuld"
          phone="" priv="read-only" pwdSet="yes"/>
      <aaaUser descr="testuser" dn="sys/user-ext/user-aaa" email=""
          expiration="never" expires="no" firstName="a" intId="10620" lastName="aa"
          name="aaa" phone="" priv="aaa,read-only" pwdSet="no"/>
    </outConfig>
</configResolveChildren>
```

configResolveChildrenSorted

The `configResolveChildrenSorted` method finds the MO in the sorted order using the specified filter.

Request Syntax

```
<xs:element name="configResolveChildrenSorted" type="configResolveChildrenSorted"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveChildrenSorted" mixed="true">
          <xs:all>
               <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inDn" type="referenceObject"/>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inSize" type="xs:unsignedShort"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="classId" type="namingClassId"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveChildrenSorted" type="configResolveChildrenSorted"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveChildrenSorted" mixed="true">
          <xs:all/>
          <xs:attribute name="outTotalSize" type="xs:unsignedInt"/>
          <xs:attribute name="outContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="classId" type="namingClassId"/>
     </xs:complexType>
```

Examples

Request

```
<configResolveChildrenSorted
cookie="<real cookie>">
</configResolveChildrenSorted>
```

Response

```
<configResolveChildrenSorted
cookie="<real cookie>" response="yes" outTotalSize="32"
outContext="0">
 <outConfigs>
 <syntheticFileSystem dn="FS-"/>
 <aaaLog dn="aaa-log" maxSize="10000" purgeWindow="100" size="6394"/>
 <apeManager dn="ape" statsUpdateId="1"/>
 <callhomeEp adminState="off" alertThrottlingAdminState="on" configState="ok" descr=""
dn="call-home" fsmDescr="" fsmPrev="configCallhomeSuccess" fsmProgr="100"
fsmRmtInvErrCode="none" fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr=""
fsmStamp="2016-08-26T00:03:23.370" fsmStatus="nop" fsmTry="0" intId="10004" name=""
policyLevel="0" policyOwner="local"/>
 <capabilityCatalogue dn="capabilities" fileParseFailures="0" filesParsed="47"
fsmDescr="" fsmPrev="nop" fsmProgr="100" fsmRmtInvErrCode="none" fsmRmtInvErrDescr=""
fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="never" fsmStatus="nop" fsmTry="0"
loadErrors="0" loadWarnings="0" providerLoadFailures="0" providersLoaded="1412"
version="1.0(0.1)T"/>
 <computeDefaults dn="compute-defaults"/>
 <dhcpInst dn="dhcp"/>
 <eventHolder dn="event" name=""/>
 <eventLog dn="event-log" maxSize="10000" purgeWindow="100" size="9969"/>
 <extpolEp dn="extpol" fsmDescr="" fsmPrev="RegisterFsmSuccess" fsmProgr="100"
fsmRmtInvErrCode="none" fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr=""
fsmStamp="2016-08-03T02:22:43.737" fsmStatus="nop" fsmTry="0"/>
 <fabricEp dn="fabric"/>
 <faultHolder dn="fault" isPinningCleared="yes" lastPinTime="2016-08-25T23:55:23.301"
name="" totalFaults="564169724198922"/>
 <gmetaEp dn="gmeta"/>
 <ippoolUniverse dn="ip"/>
 <iqnpoolUniverse dn="iqn"/>
 <macpoolUniverse dn="mac"/>
 <topMetaInf dn="meta" ecode="" name=""/>
 <identMetaVerse dn="metaverse"/>
 <nfsEp dn="nfs-ep"/>
 <observeObservedCont dn="observe" idCount="0"/>
 <orgOrg descr="" dn="org-root" fltAggr="1185410973700" level="root" name="root"
permAccess="yes"/>
 <policyPolicyEp dn="policy-ep"/>
 <procManager dn="proc-info"/>
 <statsHolder dn="stats" name=""/>
 <storageDomainEp dn="storage-ep"/>
 <topSystem address="10.xx.xxx.xx" currentTime="2016-08-29T19:43:45.181" descr=""
dn="sys" ipv6Addr="::" mode="cluster" name="bgl-col02" owner="" site=""
systemUpTime="03:19:59:58"/>
 <topSysDefaults dn="sys-defaults"/>
 <clitestTypeTest achar="0" adate="1970-01-01T05:30:00.000"
adatetime="1970-01-01T05:30:00.000" afloat="0.000000" amac="00:00:00:00:00:00"
anenum="up" anipv4="0.0.0.0" anipv6="::" ansbyte="0" ansint16="0" ansint32="0"
ansint64="0" apassword="" arange="0" arcstring="" arxstring="redyellow" astring=""
atime="00:00:00:00.000" aubyte="0" auint16="0" auint32="0" auint64="0"
awwn="00:00:00:00:00:00:00:00" dn="tt-"/>
 <clitestTypeTest2 aPartialEnum="untagged" abitmask="up" acharbuf="" dn="tt2-"
fileDir="" fileHost="" fileName="" filePasswd="" filePath="" filePort="0"
fileProto="none" fileUser=""/>
 <uuidpoolUniverse dn="uuid"/>
 <vmEp dn="vmm"/>
 <fcpoolUniverse dn="wwn"/>
 </outConfigs>
 </configResolveChildrenSorted>
```

configResolveClass

The `configResolveClass` method returns requested managed object in a given class. If inHierarchical=true, the results contain children.

Request Syntax

```
<xs:element name="configResolveClass" type="configResolveClass"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveClass" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveClass" type="configResolveClass"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveClass" mixed="true">
```

```
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveClass
    cookie="<real_cookie>"
    classId="pkiEp"
    inHierarchical="false">
    <inFilter>
    </inFilter>
</configResolveClass>
```

Response

```
<configResolveClass
    cookie="<real_cookie>"
    commCookie="11/15/0/2a5b"
    srcExtSys="10.193.33.120"
    destExtSys="10.193.33.120"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes"
    classId="pkiEp">
    <outConfig>
       <pkiEp descr=""
          dn="sys/pki-ext"
          intId="10037"
          name=""/>
    </outConfig>
</configResolveClass>
```

configResolveClasses

The `configResolveClasses` method returns requested managed objects in several classes. If inHierarchical=true, the results contain children.

Request Syntax

```
<xs:element name="configResolveClasses" type="configResolveClasses"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveClasses" mixed="true">
            <xs:all>
                <xs:element name="inIds" type="classIdSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveClasses" type="configResolveClasses"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveClasses" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveClasses
    cookie="<real_cookie>"
    inHierarchical="false">
    <inIds>
       <Id value="computeItem"/>
       <Id value="equipmentChassis"/>
    </inIds>
</configResolveClasses>
```

Response

(This is an abbreviated response.)

```
<configResolveClasses
    cookie="<real_cookie>"
    response="yes">
    <outConfigs>
       <computeItem
          adminPower="policy"
          adminState="in-service"
          dn="sys/chassis-1/blade-1"
          .
```

```
            ./>
         <computeItem
            adminPower="policy"
            adminState="in-service"
            dn="sys/chassis-1/blade-3"
            .
            ./>
         <computeItem
            adminPower="policy"
            adminState="in-service"
            dn="sys/chassis-1/blade-5"
            .
            ./>
         <computeItem
            adminState="acknowledged"
            configState="ok"
            .
            ./>
      </outConfigs>
   </configResolveClasses>
```

configResolveClassesSorted

The `configResolveClassesSorted` method returns MO instances of the specified class IDs in sorted order, satisfying the filter constraints. This method returns MOs of the specified class as well as subclasses. If true, this method returns the requested Managed Object and all of the contained children, recursively. If false, this method returns the requested Managed Object, without any contained children.

Request Syntax

```
<xs:element name="configResolveClassesSorted" type="configResolveClassesSorted"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveClassesSorted" mixed="true">
          <xs:all>
               <xs:element name="inIds" type="classIdSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inSize" type="xs:unsignedShort"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveClassesSorted" type="configResolveClassesSorted"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveClassesSorted" mixed="true">
          <xs:all/>
          <xs:attribute name="outTotalSize" type="xs:unsignedInt"/>
          <xs:attribute name="outContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

```
<configResolveClassesSorted
classId="computeItem"
cookie="<real cookie>">
</configResolveClassesSorted>
```

Response

```
<configResolveClassesSorted
cookie="<real cookie>"
response="yes" outTotalSize="0" outContext="0">
<outConfigs>
</outConfigs>
</configResolveClassesSorted>
```

configResolveClassSorted

The `configResolveClassSorted` method returns MO instances of the specified class ID in sorted order, satisfying the filter constraints. This method returns MOs of the specified class and subclasses. If true, this method returns the requested Managed Object and all of contained children, recursively. If false, this method returns the requested Managed Object, without any contained children.

Request Syntax

```
<xs:element name="configResolveClassSorted" type="configResolveClassSorted"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveClassSorted" mixed="true">
          <xs:all>
               <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inSize" type="xs:unsignedShort"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="classId" type="namingClassId"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveClassSorted" type="configResolveClassSorted"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveClassSorted" mixed="true">
          <xs:all/>
          <xs:attribute name="outTotalSize" type="xs:unsignedInt"/>
          <xs:attribute name="outContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="classId" type="namingClassId"/>
     </xs:complexType>
```

Examples

Request

```
<configResolveClassSorted
inSize=1
cookie="<real cookie>">
</configResolveClassSorted>
```

Response

```
<configResolveClassSorted
cookie="<real cookie>"
response="yes" outTotalSize="33" outContext="1472480453517">
<outConfigs>
<syntheticFileSystem dn="FS-"/>
</outConfigs>
</configResolveClassSorted>
```

configResolveContext

The `configResolveContext` method specifies how many objects should be returned. If 0, return all objects

Request Syntax

```
<xs:element name="configResolveContext" type="configResolveContext"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveContext" mixed="true">
          <xs:attribute name="inContext" type="xs:unsignedLong"/>
          <xs:attribute name="inSize" type="xs:unsignedShort"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveContext" type="configResolveContext"
substitutionGroup="externalMethod"/>
     <xs:complexType name="configResolveContext" mixed="true">
          <xs:all/>
          <xs:attribute name="outContext" type="xs:unsignedLong"/>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

```
<configResolveContext
cookie="<real cookie>"
response="Yes"
inSize="1">
</configResolveContext>
```

Response

```
<configResolveContext
cookie="<real cookie>"
response="yes" errorCode="150" invocationResult="unidentified-fail"
errorDescr="Invalid context: 0">
</configResolveContext>
```

configResolveDn

The `configResolveDn` method retrieves a single managed object for a specified DN.

Request Syntax

```
<xs:element name="configResolveDn" type="configResolveDn"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveDn" mixed="true">
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveDn" type="configResolveDn"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveDn" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveDn
    cookie="<real_cookie>"
    dn="vmmEp/vm-mgr-vcenter1" />
```

Response

```
<configResolveDn dn="vmmEp/vm-mgr-vcenter1"
    cookie="<real_cookie>"
    commCookie="9/15/0/1c0d"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="vm-mgr_dme"
    response="yes">
    <outConfig>
       <vmManager
          adminState="enable"
          descr=""
          dn="vmmEp/vm-mgr-vcenter1"
          fltAggr="0"
          fsmDescr="AG registration with
              vCenter(FSM:sam:dme:VmManagerRegisterWithVCenter)"
          fsmPrev="RegisterWithVCenterRegistering"
          fsmProgr="13"
          fsmRmtInvErrCode="none"
          fsmRmtInvErrDescr=""
          fsmRmtInvRslt=""
          fsmStageDescr="AG registration with
             vCenter(FSM-STAGE:sam:dme:VmManagerRegisterWithVCenter:Registering)"
          fsmStamp="2010-11-11T21:37:15.696"
          fsmStatus="RegisterWithVCenterRegistering"
          fsmTry="1"
          hostName="savbu-vpod-dev-31.cisco.com"
          intId="21959"
          name="vcenter1"
          operState="unknown"
          stateQual=""
          type="vmware"
          version=""/>
    </outConfig>
</configResolveDn>
```

configResolveDns

The `configResolveDns` method retrieves the managed objects for a list of DNs.

Request Syntax

```
<xs:element name="configResolveDns" type="configResolveDns"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveDns" mixed="true">
            <xs:all>
                <xs:element name="inDns" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveDns" type="configResolveDns"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveDns" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
                <xs:element name="outUnresolved" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveDns
    cookie="<real_cookie>"
    inHierarchical="false">
    <inDns>
       <dn value="sys/chassis-1" />
       <dn value="sys/chassis-1/blade-1/board/cpu-1" />
       <dn value="sys/chassis-1/blade-1/board/t-stats" />
    </inDns>
</configResolveDns>
```

Response

```
<configResolveDns
    cookie="<real_cookie>"
    response="yes">
    <outConfigs>
       <processorUnit
          arch="Xeon"
          cores="4"
          dn="sys/chassis-1/blade-1/board/cpu-1"
          id="1"
          model="Intel(R) Xeon(R) CPU E5520 @ 2.27GHz"
          operState="operable"
          operability="operable"
          perf="not-supported"
          power="not-supported"
          presence="equipped"
          revision="0"
          serial=""
          socketDesignation="CPU1"
          speed="2.266000"
          stepping="5"
          thermal="ok"
          threads="8"
          vendor="Intel(R) Corporation"
          voltage="ok"/>
       <equipmentChassis
       .
       .>
    </outConfigs>
    <outUnresolved>
       <dn value="sys/chassis-1/blade-1/board/t-stats"/>
    </outUnresolved>
</configResolveDns>
```

configResolveParent

For a specified DN, the `configResolveParent` method retrieves the parent of the managed object.

Request Syntax

```
<xs:element name="configResolveParent" type="configResolveParent"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveParent" mixed="true">
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveParent" type="configResolveParent"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveParent" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveParent
    cookie="<real_cookie>"
    inHierarchical="false"
    dn="sys/chassis-1/blade-1/adaptor-1">
</configResolveParent>
```

Response

```
<configResolveParent
    dn="sys/chassis-1/blade-1/adaptor-1"
    cookie="<real_cookie>"
    response="yes">
    <outConfig>
       <computeItem
          adminPower="policy"
          adminState="in-service"
          assignedToDn=""
          association="none"
          availability="available"
          availableMemory="10240"
          chassisId="1"
          checkPoint="discovered"
          connPath="A,B"
          connStatus="A,B"
          descr=""
          diagnostics="complete"
          discovery="complete"
          dn="sys/chassis-1/blade-1"
          fltAggr="0"
          fsmDescr=""
          fsmFlags=""
          fsmPrev="DiscoverSuccess"
          fsmProgr="100"
          fsmRmtInvErrCode="none"
          fsmRmtInvErrDescr=""
          fsmRmtInvRslt=""
          fsmStageDescr=""
          fsmStamp="2009-09-23T23:44:30"
          fsmStatus="nop"
          fsmTry="0"
          intId="768052"
          lc="discovered"
          lcTs="1969-12-31T16:00:00"
          managingInst="B"
          model="N20-B6620-1"
          name=""
          numOfAdaptors="1"
          numOfCores="8"
          numOfCpus="2"
          numOfEthHostIfs="2"
          numOfFcHostIfs="0"
          numOfThreads="16"
          operPower="off"
          operQualifier=""
          operState="unassociated"
          operability="operable"
          originalUuid="e3516842-d0a4-11dd-baad-000bab01bfd6"
          presence="equipped"
          revision="0"
          serial="QCI12520024"
          slotId="1"
          totalMemory="10240"
          uuid="e3516842-d0a4-11dd-baad-000bab01bfd6"
          vendor="Cisco Systems Inc"/>
    </outConfig>
</configResolveParent>
```

configScope

The `configScope` method returns managed objects and details about their configuration.

Request Syntax

```
<xs:element name="configScope" type="configScope" substitutionGroup="externalMethod"/>
        <xs:complexType name="configScope" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inClass" type="namingClassId"/>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inRecursive">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configScope" type="configScope" substitutionGroup="externalMethod"/>
        <xs:complexType name="configScope" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<configScope
    dn="org-root"
    cookie="<real_cookie>"
    inClass="orgOrgRes"
    inHierarchical="false"
    inRecursive="false">
    <inFilter>
    </inFilter>
</configScope>
```

Response

```
<configScope dn="org-root"
    cookie="<real_cookie>"
    commCookie="2/15/0/2a53"
    srcExtSys="10.193.33.120"
    destExtSys="10.193.33.120"
    srcSvc="sam_extXMLApi"
    destSvc="service-reg_dme"
    response="yes">
    <outConfigs>
       <orgOrgCaps dn="org-root/org-caps" org="512" tenant="64"/>
       <orgOrgCounts dn="org-root/org-counter" org="36" tenant="7"/>
    </outConfigs>
</configScope>
```

equipmentClone

The `equipmentClone` method is used to clone a chassis profile template.

Request Syntax

```
<xs:element name="equipmentClone" type="equipmentClone"
substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentClone" mixed="true">
          <xs:attribute name="inTargetOrg" type="referenceObject"/>
          <xs:attribute name="inChassisProfileName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="equipmentClone" type="equipmentClone"
substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentClone" mixed="true">
          <xs:all>
               <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

```
<equipmentClone
dn="org-root/cp-testTemplate"
cookie="<real cookie>"
inTargetOrg="org-root"
inChassisProfileName="TestCP"
inHierarchical="no">
</equipmentClone>
```

Response

```
<equipmentClone dn="org-root/cp-testTemplate"
cookie="<real cookie>" response="yes">
<outConfig>
<equipmentChassisProfile assignState="unassigned" assocState="unassociated"
chassisDn="" chassisFwPolicyName="" configQualifier="" configState="not-applied"
descr="" diskZoningPolicyName="default" dn="org-root/cp-TestCP" fltAggr="0"
fsmDescr="" fsmFlags="" fsmPrev="nop" fsmProgr="100" fsmRmtInvErrCode="none"
fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="never" fsmStatus="nop"
fsmTry="0" intId="6090680" maintPolicyName="" name="TestCP" operChassisFwPolicyName=""
operDiskZoningPolicyName="" operMaintPolicyName="" operSrcTemplName=""
operState="unassociated" owner="management" policyLevel="0" policyOwner="local"
propAcl="0" resolveRemote="yes" srcTemplName="" status="created" type="initial-template"
usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
</outConfig>
</equipmentClone>
```

equipmentInstantiateNNamedTemplate

The `equipmentInstantiateNNamedTemplate` method takes the specified service profile template and instantiates the desired number of service profiles.

Request Syntax

```
<xs:element name="equipmentInstantiateNNamedTemplate"
type="equipmentInstantiateNNamedTemplate" substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentInstantiateNNamedTemplate" mixed="true">
          <xs:all>
               <xs:element name="inNameSet" type="dnSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inTargetOrg" type="referenceObject"/>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inErrorOnExisting">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="equipmentInstantiateNNamedTemplate"
type="equipmentInstantiateNNamedTemplate" substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentInstantiateNNamedTemplate" mixed="true">
          <xs:all>
               <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

```
<equipmentInstantiateNNamedTemplate
dn="org-root/cp-testTemplate"
cookie="<real cookie>"
inTargetOrg="org-root"
inHierarchical="no">
<inNameSet>
<dn value="mycp"/>
</inNameSet>
</equipmentInstantiateNNamedTemplate>
```

Response

```
<equipmentInstantiateNNamedTemplate
dn="org-root/cp-testTemplate"
cookie="<real cookie>" response="yes">
<outConfigs>
<equipmentChassisProfile assignState="assigned" assocState="associated"
chassisDn="sys/chassis-1" chassisFwPolicyName="" configQualifier=""
configState="applied" descr="" diskZoningPolicyName="default" dn="org-root/cp-mycp"
fltAggr="0" fsmDescr="" fsmFlags="" fsmPrev="ConfigureSuccess" fsmProgr="100"
fsmRmtInvErrCode="none" fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr=""
fsmStamp="2016-08-25T23:52:22.480" fsmStatus="nop" fsmTry="0" intId="5437252"
maintPolicyName="" name="mycp" operChassisFwPolicyName="org-root/fw-chassis-pack-default"
operDiskZoningPolicyName="org-root/disk-zoning-policy-default"
operMaintPolicyName="org-root/chassis-profile-maint-default" operSrcTemplName=""
operState="ok" owner="management" policyLevel="0" policyOwner="local" propAcl="0"
resolveRemote="yes" srcTemplName="testTemplate" status="modified" type="instance"
usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
```

```
</outConfigs>
</equipmentInstantiateNNamedTemplate>
```

equipmentInstantiateNTemplate

The equipmentInstantiateNTemplate method creates a number (N) of service profiles from a template.

Request Syntax

```
<xs:element name="equipmentInstantiateNTemplate"
type="equipmentInstantiateNTemplate" substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentInstantiateNTemplate" mixed="true">
          <xs:attribute name="inTargetOrg" type="referenceObject"/>
          <xs:attribute name="inChassisProfileNamePrefixOrEmpty">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inNumberOf" type="xs:unsignedByte"/>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="equipmentInstantiateNTemplate"
type="equipmentInstantiateNTemplate" substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentInstantiateNTemplate" mixed="true">
          <xs:all>
               <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

```
<equipmentInstantiateNTemplate
dn="org-root/cp-testTemplate"
cookie="1472654892/884439e6-4824-4697-9bb5-f00d96bafe9b"
inTargetOrg="org-root"
inChassisProfileNamePrefixOrEmpty="TestCP"
inNumberOf="2"
inHierarchical="no">
</equipmentInstantiateNTemplate>
```

Response

```
<equipmentInstantiateNTemplate
dn="org-root/cp-testTemplate"
cookie="1472654892/884439e6-4824-4697-9bb5-f00d96bafe9b" response="yes">
<outConfigs>
<equipmentChassisProfile assignState="unassigned" assocState="unassociated"
chassisDn="" chassisFwPolicyName="" configQualifier="" configState="not-applied"
descr="" diskZoningPolicyName="default" dn="org-root/cp-TestCP1" fltAggr="0" fsmDescr=""
fsmFlags="" fsmPrev="nop" fsmProgr="100" fsmRmtInvErrCode="none" fsmRmtInvErrDescr=""
fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="never" fsmStatus="nop" fsmTry="0"
intId="6090568" maintPolicyName="" name="TestCP1" operChassisFwPolicyName=""
operDiskZoningPolicyName="" operMaintPolicyName="" operSrcTemplName=""
operState="unassociated" owner="management" policyLevel="0" policyOwner="local"
propAcl="0" resolveRemote="yes" srcTemplName="testTemplate" status="created"
type="instance" usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
<equipmentChassisProfile assignState="unassigned" assocState="unassociated"
chassisDn="" chassisFwPolicyName="" configQualifier="" configState="not-applied"
descr="" diskZoningPolicyName="default" dn="org-root/cp-TestCP2" fltAggr="0"
fsmDescr="" fsmFlags="" fsmPrev="nop" fsmProgr="100" fsmRmtInvErrCode="none"
fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="never" fsmStatus="nop"
fsmTry="0" intId="6090571" maintPolicyName="" name="TestCP2" operChassisFwPolicyName=""
operDiskZoningPolicyName="" operMaintPolicyName="" operSrcTemplName=""
operState="unassociated" owner="management" policyLevel="0" policyOwner="local"
propAcl="0" resolveRemote="yes" srcTemplName="testTemplate" status="created"
type="instance" usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
</outConfigs>
</equipmentInstantiateNTemplate>
```

equipmentInstantiateTemplate

The `equipmentInstantiateTemplate` method creates one chassis profile from a specified template.

Request Syntax

```
<xs:element name="equipmentInstantiateTemplate"
type="equipmentInstantiateTemplate" substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentInstantiateTemplate" mixed="true">
          <xs:attribute name="inTargetOrg" type="referenceObject"/>
          <xs:attribute name="inChassisProfileName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
```

```
                                <xs:enumeration value="yes"/>
                           </xs:restriction>
                      </xs:simpleType>
                 </xs:union>
            </xs:simpleType>
       </xs:attribute>
       <xs:attribute name="inErrorOnExisting">
            <xs:simpleType>
                 <xs:union memberTypes="xs:boolean">
                      <xs:simpleType>
                           <xs:restriction base="xs:string">
                                <xs:enumeration value="no"/>
                                <xs:enumeration value="yes"/>
                           </xs:restriction>
                      </xs:simpleType>
                 </xs:union>
            </xs:simpleType>
       </xs:attribute>
       <xs:attribute name="cookie" type="xs:string"/>
       <xs:attribute name="response" type="YesOrNo"/>
       <xs:attribute name="dn" type="referenceObject"/>
  </xs:complexType>
```

Response Syntax

```
<xs:element name="equipmentInstantiateTemplate"
type="equipmentInstantiateTemplate" substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentInstantiateTemplate" mixed="true">
          <xs:all>
               <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

```
<equipmentInstantiateTemplate
dn="org-root/cp-testTemplate"
cookie="<real_cookie>"
inTargetOrg="org-root"
inChassisProfileName="CP1"
inHierarchical="no">
</equipmentInstantiateTemplate>
```

Response

```
<equipmentInstantiateTemplate
dn="org-root/cp-testTemplate"
cookie="<real_cookie>" response="yes">
<outConfig>
<equipmentChassisProfile assignState="unassigned" assocState="unassociated"
chassisDn="" chassisFwPolicyName="" configQualifier="" configState="not-applied"
descr="" diskZoningPolicyName="default" dn="org-root/cp-CP1" fltAggr="0" fsmDescr=""
fsmFlags="" fsmPrev="nop" fsmProgr="100" fsmRmtInvErrCode="none" fsmRmtInvErrDescr=""
fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="never" fsmStatus="nop" fsmTry="0"
intId="6090685" maintPolicyName="" name="CP1" operChassisFwPolicyName=""
```

```
operDiskZoningPolicyName="" operMaintPolicyName="" operSrcTemplName=""
operState="unassociated" owner="management" policyLevel="0" policyOwner="local"
propAcl="0" resolveRemote="yes" srcTemplName="testTemplate" status="created"
type="instance" usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
</outConfig>
</equipmentInstantiateTemplate>
```

equipmentResolveTemplates

The `equipmentResolveTemplates` method retrieves the chassis profile templates from the specified organization, which is matched hierarchically.

Request Syntax

```
<xs:element name="equipmentResolveTemplates" type="equipmentResolveTemplates"
substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentResolveTemplates" mixed="true">
          <xs:all>
               <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="inExcludeIfBound">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inType">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="initial-template"/>
                         <xs:enumeration value="updating-template"/>
                         <xs:enumeration value="all"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="equipmentResolveTemplates" type="equipmentResolveTemplates"
substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentResolveTemplates" mixed="true">
          <xs:all>
               <xs:element name="outConfigs" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_15">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
               </xs:element>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

```
<equipmentResolveTemplates
dn="org-root"
cookie="<real_cookie>"
inExcludeIfBound="false"
inType="initial-template"
inHierarchical="no">
</equipmentResolveTemplates>
```

Response

```
<equipmentResolveTemplates dn="org-root"
cookie="<real_cookie>" response="yes">
<outConfigs>
<pair key="cp-testTemplate">
<equipmentChassisProfile assignState="unassigned" assocState="unassociated"
chassisDn="" chassisFwPolicyName="" configQualifier="" configState="not-applied"
descr="" diskZoningPolicyName="default" dn="org-root/cp-testTemplate" fltAggr="0"
fsmDescr="" fsmFlags="" fsmPrev="nop" fsmProgr="100" fsmRmtInvErrCode="none"
fsmRmtInvErrDescr="" fsmRmtInvRslt="" fsmStageDescr="" fsmStamp="never" fsmStatus="nop"
fsmTry="0" intId="6090461" maintPolicyName="" name="testTemplate"
operChassisFwPolicyName="org-root/fw-chassis-pack-default"
operDiskZoningPolicyName="org-root/disk-zoning-policy-default"
operMaintPolicyName="org-root/chassis-profile-maint-default"
operSrcTemplName="" operState="unassociated" owner="management" policyLevel="0"
policyOwner="local" propAcl="0" resolveRemote="yes" srcTemplName=""
type="initial-template" usrLbl="" uuid="00000000-0000-0000-0000-000000000000"/>
</pair>
</outConfigs>
</equipmentResolveTemplates>
```

equipmentTemplatise

The `equipmentTemplatise` method creates a template from a specified chassis profile.

Request Syntax

```
<xs:element name="equipmentTemplatise" type="equipmentTemplatise"
substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentTemplatise" mixed="true">
          <xs:attribute name="inTargetOrg" type="referenceObject"/>
          <xs:attribute name="inTemplateName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inTemplateType">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="instance"/>
                         <xs:enumeration value="initial-template"/>
                         <xs:enumeration value="updating-template"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inHierarchical">
               <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                              <xs:restriction base="xs:string">
                                   <xs:enumeration value="no"/>
                                   <xs:enumeration value="yes"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="equipmentTemplatise" type="equipmentTemplatise"
substitutionGroup="externalMethod"/>
     <xs:complexType name="equipmentTemplatise" mixed="true">
          <xs:all>
               <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
          <xs:attribute name="dn" type="referenceObject"/>
     </xs:complexType>
```

Examples

Request

Response

eventSendHeartbeat

The `eventSendHeartbeat` method allows clients to retrieve any missed event. Each event has a unique event ID. These event IDs operate as counters and are included in all method responses.

Each time an event is generated, the event ID counter increases and the new event is assigned a new event ID. This enables the subscriber to track the events. If an event is missed by the client, the client can use the `eventSendEvent` method to retrieve the missed event.

Request Syntax

```
<xs:element name="eventSendHeartbeat" type="eventSendHeartbeat"
substitutionGroup="externalMethod"/>
        <xs:complexType name="eventSendHeartbeat" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="eventSendHeartbeat" type="eventSendHeartbeat"
substitutionGroup="externalMethod"/>
        <xs:complexType name="eventSendHeartbeat" mixed="true">
            <xs:attribute name="outSystemTime" type="dateTime"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

When the client application subscribes to an event or events by using `eventSubscribe`, Cisco UCS sends `eventSendHeartbeat` periodically (default 120 seconds).

Response

```
<eventSendHeartbeat
    cookie="<real_cookie>"
    commCookie=""
    srcExtSys="0.0.0.0"
    destExtSys="0.0.0.0"
    srcSvc=""
    destSvc=""
    response="yes"
    outSystemTime="2010-11-12T20:38:19.630">
</eventSendHeartbeat>
```

eventSubscribe

The `eventSubscribe` method allows a client to subscribe to asynchronous events generated by Cisco UCS, including all object changes in the system (created, changed, or deleted).

Event subscription allows a client application to register for event notification from Cisco UCS. When an event occurs, Cisco UCS informs the client application of the event and its type. Only the actual change information is sent. The object's unaffected attributes are not included.

Use `eventSubscribe` to register for events as shown in the following example:

```
<eventSubscribe
    cookie="<real_cookie>">
</eventSubscribe>
```

Request Syntax

```
<xs:element name="eventSubscribe" type="eventSubscribe"
            substitutionGroup="externalMethod"/>
        <xs:complexType name="eventSubscribe" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="eventSubscribe" type="eventSubscribe"
substitutionGroup="externalMethod"/>
        <xs:complexType name="eventSubscribe" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<eventSubscribe
    cookie="<real_cookie>">
    <inFilter>
    </inFilter>
</eventSubscribe>
```

Response

```
NO RESPONSE OR ACKNOWLEDGMENT.
```

eventUnsubscribe

The `eventUnsubscribe` method allows a client to unsubscribe from asynchronous events generated by Cisco UCS, reversing event subscriptions that resulted from `eventSubscribe.`

Use `eventUnsubscribe` to unsubscribe from events as shown in the following example:

```
<eventUnsubscribe
    cookie="<real_cookie>">
</eventUnsubscribe>
```

Request Syntax

```
<xs:element name="eventUnsubscribe" type="eventUnsubscribe"
            substitutionGroup="externalMethod"/>
        <xs:complexType name="eventUnsubscribe" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="eventUnsubscribe" type="eventUnsubscribe"
substitutionGroup="externalMethod"/>
        <xs:complexType name="eventUnsubscribe" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<eventUnsubscribe
    cookie="<real_cookie>">
    <inFilter>
    </inFilter>
</eventUnsubscribe>
```

Response

```
NO RESPONSE OR ACKNOWLEDGMENT.
```

faultAckFault

The `faultAckFault` method acknowledges a fault. The acknowledgment response marks the fault severity as cleared. Faults categorized as auto-cleared do not require acknowledgment.

Request Syntax

```
<xs:element name="faultAckFault" type="faultAckFault" substitutionGroup="externalMethod"/>
        <xs:complexType name="faultAckFault" mixed="true">
            <xs:attribute name="inId" type="xs:unsignedLong"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="faultAckFault" type="faultAckFault" substitutionGroup="externalMethod"/>
        <xs:complexType name="faultAckFault" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<faultAckFault
    inHierarchical="false"
    cookie="<real_cookie>"
    inId="10120" />
```

Response

```
<faultAckFault
    cookie="<real_cookie>"
    commCookie="5/15/0/6c"
    srcExtSys="10.193.33.214"
    destExtSys="10.193.33.214"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
</faultAckFault>
```

faultAckFaults

The `faultAckFaults` method acknowledges multiple faults. The acknowledgment response marks the fault severity as cleared. Faults categorized as auto-cleared do not require acknowledgment.

Request Syntax

```
<xs:element name="faultAckFaults" type="faultAckFaults"
substitutionGroup="externalMethod"/>
        <xs:complexType name="faultAckFaults" mixed="true">
            <xs:all>
                <xs:element name="inIds" type="idSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="faultAckFaults" type="faultAckFaults"
substitutionGroup="externalMethod"/>
        <xs:complexType name="faultAckFaults" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<faultAckFaults
    cookie="<real_cookie>">
    <inIds>
       <id value="10656"/>
       <id value="10660"/>
    </inIds>
</faultAckFaults>
```

Response

```
<faultAckFaults
    cookie="<real_cookie>"
    commCookie="11/15/0/505"
    srcExtSys="10.193.34.70"
    destExtSys="10.193.34.70"
    srcSvc="sam_extXMLApi"
    destSvc="mgmt-controller_dme"
    response="yes">
</faultAckFaults>
```

faultResolveFault

The `faultResolveFault` method sends a response when a fault has been resolved.

Request Syntax

```
<xs:element name="faultResolveFault" type="faultResolveFault"
substitutionGroup="externalMethod"/>
        <xs:complexType name="faultResolveFault" mixed="true">
            <xs:attribute name="inId" type="xs:unsignedLong"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="faultResolveFault" type="faultResolveFault"
substitutionGroup="externalMethod"/>
        <xs:complexType name="faultResolveFault" mixed="true">
            <xs:all>
                <xs:element name="outFault" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<faultResolveFault
    inHierarchical="false"
    cookie="<real_cookie>"
    inId="10120" />
```

Response

```
<faultResolveFault
    cookie="<real_cookie>"
    commCookie="5/15/0/6a"
    srcExtSys="10.193.33.214"
    destExtSys="10.193.33.214"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outFault>
       <faultInst
          ack="yes"
          cause="empty-pool"
          changeSet=""
```

```
            code="F0135"
            created="2010-11-19T11:02:41.568"
            descr="Virtual Security Gateway pool default is empty"
            dn="org-root/fwpool-default/fault-F0135"
            highestSeverity="minor"
            id="10120"
            lastTransition="2010-11-19T11:02:41.568"
            lc=""
            occur="1"
            origSeverity="minor"
            prevSeverity="minor"
            rule="fw-pool-empty"
            severity="minor"
            tags=""
            type="equipment"/>
      </outFault>
</faultResolveFault>
```

lsClone

The `lsClone` method clones a service profile or a service profile template.

Request Syntax

```
<xs:element name="lsClone" type="lsClone" substitutionGroup="externalMethod"/>
        <xs:complexType name="lsClone" mixed="true">
            <xs:attribute name="inTargetOrg" type="referenceObject"/>
            <xs:attribute name="inServerName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="lsClone" type="lsClone" substitutionGroup="externalMethod"/>
        <xs:complexType name="lsClone" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Two examples are provided: cloning a service profile and closing a service profile template.

Request (service profile)

```
<lsClone
    dn="org-root/ls-SP1"
    cookie="<real_cookie>"
    inTargetOrg="org-root"
    inServerName="CP-1"
    inHierarchical="no">
</lsClone>
```

Response (service profile)

```
<lsClone
    dn="org-root/ls-SP1"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfig>
       <lsServer
          agentPolicyName=""
          assignState="unassigned"
          assocState="unassociated"
          biosProfileName=""
          bootPolicyName=""
          configQualifier=""
          configState="not-applied"
          descr=""
          dn="org-root/ls-CP-1"
          dynamicConPolicyName=""
          extIPState="none"
          fltAggr="0"
          hostFwPolicyName=""
          identPoolName="default"
          intId="52365"
          localDiskPolicyName="default"
          maintPolicyName=""
          mgmtAccessPolicyName=""
          mgmtFwPolicyName=""
          name="CP-1"
          operBiosProfileName=""
          operBootPolicyName=""
          operDynamicConPolicyName=""
          operHostFwPolicyName=""
          operIdentPoolName=""
          operLocalDiskPolicyName=""
          operMaintPolicyName=""
```

```
            operMgmtAccessPolicyName=""
            operMgmtFwPolicyName=""
            operPowerPolicyName=""
            operScrubPolicyName=""
            operSolPolicyName=""
            operSrcTemplName=""
            operState="unassociated"
            operStatsPolicyName=""
            operVconProfileName=""
            owner="management"
            pnDn=""
            powerPolicyName="default"
            scrubPolicyName=""
            solPolicyName=""
            srcTemplName="service-templ-001"
            statsPolicyName="default"
            status="created"
            type="instance"
            usrLbl=""
            uuid="derived"
            uuidSuffix="0000-000000000000"
            vconProfileName=""/>
    </outConfig>
</lsClone>
```

Request (service profile template)

```
<lsClone
    dn="org-root/ls-template-3"
    cookie="<real_cookie>"
    inTargetOrg="org-root"
    inServerName="CT-1"
    inHierarchical="no">
</lsClone>
```

Response (service profile template)

```
<lsClone
    dn="org-root/ls-template-3"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfig>
       <lsServer
          agentPolicyName=""
          assignState="unassigned"
          assocState="unassociated"
          biosProfileName=""
          bootPolicyName=""
          configQualifier=""
          configState="not-applied"
          descr=""
          dn="org-root/ls-CT-1"
          dynamicConPolicyName=""
          extIPState="none"
          fltAggr="0"
          hostFwPolicyName=""
          identPoolName="default"
          intId="52389"
          localDiskPolicyName=""
```

```
            maintPolicyName=""
            mgmtAccessPolicyName=""
            mgmtFwPolicyName=""
            name="CT-1"
            operBiosProfileName=""
            operBootPolicyName=""
            operDynamicConPolicyName=""
            operHostFwPolicyName=""
            operIdentPoolName=""
            operLocalDiskPolicyName=""
            operMaintPolicyName=""
            operMgmtAccessPolicyName=""
            operMgmtFwPolicyName=""
            operPowerPolicyName=""
            operScrubPolicyName=""
            operSolPolicyName=""
            operSrcTemplName=""
            operState="unassociated"
            operStatsPolicyName=""
            operVconProfileName=""
            owner="management"
            pnDn=""
            powerPolicyName="default"
            scrubPolicyName=""
            solPolicyName=""
            srcTemplName=""
            statsPolicyName="default"
            status="created"
            type="updating-template"
            usrLbl=""
            uuid="derived"
            uuidSuffix="0000-000000000000"
            vconProfileName=""/>
      </outConfig>
</lsClone>
```

lsInstantiateNNamedTemplate

The `lsInstantiateNNamedTemplate` method takes the specified service profile template and instantiates the desired number of service profiles. This method uses the following parameters:

- dn—Specifies the service template used to instantiate the new service profiles.
- nameSet—Contains the names of the service profiles to be instantiated.
- targetOrg—Specifies the organization under which these service profiles are instantiated.

Request Syntax

```
<xs:element name="lsInstantiateNNamedTemplate" type="lsInstantiateNNamedTemplate"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsInstantiateNNamedTemplate" mixed="true">
            <xs:all>
                <xs:element name="inNameSet" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inTargetOrg" type="referenceObject"/>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
```

```
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="lsInstantiateNNamedTemplate" type="lsInstantiateNNamedTemplate"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsInstantiateNNamedTemplate" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<lsInstantiateNNamedTemplate
    dn="org-root/ls-service-template-001"
    cookie="<real_cookie>"
    inTargetOrg="org-root"
    inHierarchical="no">
    <inNameSet>
       <dn value="service-profile-A"/>
       <dn value="service-profile-B"/>
       <dn value="service-profile-C"/>
    </inNameSet>
</lsInstantiateNNamedTemplate>
```

Response

```
<lsInstantiateNNamedTemplate
    dn="org-root/ls-service-template-001"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfigs>
       <lsServer
```

```
            agentPolicyName=""
            assignState="unassigned"
            assocState="unassociated"
            biosProfileName=""
            bootPolicyName=""
            configQualifier=""
            configState="not-applied"
            descr=""
            dn="org-root/ls-service-profile-A "
            dynamicConPolicyName=""
            .
            .
            status="created"
            type="instance"
            usrLbl=""
            uuid="derived"
            uuidSuffix="0000-000000000000"
            vconProfileName=""/>
         <lsServer
            .
         ./>
         <lsServer
            .
         ./>
      </outConfigs>
</lsInstantiateNNamedTemplate>
```

lsInstantiateNTemplate

The `lsInstantiateNTemplate` method creates a number (N) of service profiles from a template.

Request Syntax

```
<xs:element name="lsInstantiateNTemplate" type="lsInstantiateNTemplate"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsInstantiateNTemplate" mixed="true">
            <xs:attribute name="inTargetOrg" type="referenceObject"/>
            <xs:attribute name="inServerNamePrefixOrEmpty">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inNumberOf" type="xs:unsignedByte"/>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="lsInstantiateNTemplate" type="lsInstantiateNTemplate"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsInstantiateNTemplate" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<lsInstantiateNTemplate
    dn="org-root/ls-service-templ-001"
    cookie="<real_cookie>"
    inTargetOrg="org-root"
    inServerNamePrefixOrEmpty="SP"
    inNumberOf="2"
    inHierarchical="no">
</lsInstantiateNTemplate>
```

Response

```
<lsInstantiateNTemplate
    dn="org-root/ls-service-templ-001"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfigs>
       <lsServer
          agentPolicyName=""
          assignState="unassigned"
          assocState="unassociated"
          biosProfileName=""
          bootPolicyName=""
          configQualifier=""
          configState="not-applied"
          descr=""
          dn="org-root/ls-SP1"
          dynamicConPolicyName=""
          extIPState="none"
          fltAggr="0"
          hostFwPolicyName=""
          identPoolName="default"
          intId="52227"
```

```
                localDiskPolicyName="default"
                maintPolicyName=""
                mgmtAccessPolicyName=""
                mgmtFwPolicyName=""
                name="SP1"
                operBiosProfileName=""
                operBootPolicyName=""
                operDynamicConPolicyName=""
                operHostFwPolicyName=""
                operIdentPoolName=""
                operLocalDiskPolicyName=""
                operMaintPolicyName=""
                operMgmtAccessPolicyName=""
                operMgmtFwPolicyName=""
                operPowerPolicyName=""
                operScrubPolicyName=""
                operSolPolicyName=""
                operSrcTemplName=""
                operState="unassociated"
                operStatsPolicyName=""
                operVconProfileName=""
                owner="management"
                pnDn=""
                powerPolicyName="default"
                scrubPolicyName=""
                solPolicyName=""
                srcTemplName="service-templ-001"
                statsPolicyName="default"
                status="created"
                type="instance"
                usrLbl=""
                uuid="derived"
                uuidSuffix="0000-000000000000"
                vconProfileName=""/>
             <lsServer
                .
            ./>
        </outConfigs>
    </lsInstantiateNTemplate>
```

lsInstantiateTemplate

The `lsInstantiateTemplate` method creates one service profile from a specified template.

Request Syntax

```
<xs:element name="lsInstantiateTemplate" type="lsInstantiateTemplate"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsInstantiateTemplate" mixed="true">
            <xs:attribute name="inTargetOrg" type="referenceObject"/>

            <xs:attribute name="inServerName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inHierarchical">
```

```
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="lsInstantiateTemplate" type="lsInstantiateTemplate"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsInstantiateTemplate" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<lsInstantiateTemplate
    dn="org-root/ls-service-templ-001"
    cookie="<real_cookie>"
    inTargetOrg="org-root"
    inServerName="SP1"
    inHierarchical="no">
 </lsInstantiateTemplate>
```

Response

```
<lsInstantiateTemplate
    dn="org-root/ls-service-templ-001"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfigs>
       <lsServer
          agentPolicyName=""
          assignState="unassigned"
          assocState="unassociated"
          biosProfileName=""
          bootPolicyName=""
          configQualifier=""
          configState="not-applied"
          descr=""
          dn="org-root/ls-SP1"
          dynamicConPolicyName=""
          extIPState="none"
          fltAggr="0"
          fsmDescr=""
          fsmFlags=""
          fsmPrev="nop"
          fsmProgr="100"
          fsmRmtInvErrCode="none"
          fsmRmtInvErrDescr=""
          fsmRmtInvRslt=""
          fsmStageDescr=""
          fsmStamp="never"
          fsmStatus="nop"
          fsmTry="0"
          hostFwPolicyName=""
          identPoolName="default"
          intId="52227"
          localDiskPolicyName="default"
          maintPolicyName=""
          mgmtAccessPolicyName=""
          mgmtFwPolicyName=""
          name="SP1"
          operBiosProfileName=""
          operBootPolicyName=""
          operDynamicConPolicyName=""
          operHostFwPolicyName=""
          operIdentPoolName=""
          operLocalDiskPolicyName=""
          operMaintPolicyName=""
          operMgmtAccessPolicyName=""
          operMgmtFwPolicyName=""
          operPowerPolicyName=""
          operScrubPolicyName=""
          operSolPolicyName=""
          operSrcTemplName=""
          operState="unassociated"
          operStatsPolicyName=""
          operVconProfileName=""
          owner="management"
          pnDn=""
          powerPolicyName="default"
          scrubPolicyName=""
          solPolicyName=""
          srcTemplName="service-templ-001"
          statsPolicyName="default"
          status="created"
          type="instance"
          usrLbl=""
          uuid="derived"
          uuidSuffix="0000-000000000000"
          vconProfileName=""/>
    </outConfigs>
</lsInstantiateTemplate>
```

lsResolveTemplates

The `lsResolveTemplates` method retrieves the service profile templates from the specified organization, which is matched hierarchically. The search can be further refined by providing standard querying filters in addition to querying by template type (`initial-template` or `updating-template`) and the `exclude-if-bounded` flag.

Request Syntax

```
<xs:element name="lsResolveTemplates" type="lsResolveTemplates"
substitutionGroup="externalMethod"/>
        <xs:complexType name="lsResolveTemplates" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inExcludeIfBound">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inType">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="initial-template"/>
                         <xs:enumeration value="updating-template"/>
                         <xs:enumeration value="all"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="lsResolveTemplates" type="lsResolveTemplates"
substitutionGroup="externalMethod"/>
```

```
<xs:complexType name="lsResolveTemplates" mixed="true">
    <xs:all>
        <xs:element name="outConfigs" type="configMap" minOccurs="0">
            <xs:unique name="unique_map_key_10">
                 <xs:selector xpath="pair"/>
                 <xs:field xpath="@key"/>
            </xs:unique>
        </xs:element>
    </xs:all>
    <xs:attribute name="cookie" type="xs:string"/>
    <xs:attribute name="response" type="YesOrNo"/>
    <xs:attribute name="errorCode" type="xs:unsignedInt"/>
    <xs:attribute name="errorDescr" type="xs:string"/>
    <xs:attribute name="invocationResult" type="xs:string"/>
    <xs:attribute name="dn" type="referenceObject"/>
</xs:complexType>
```

Examples

Request

```
<lsResolveTemplates
    dn="org-root/org-level1"
    cookie="<real_cookie>"
    inExcludeIfBound="false"
    inType="initial-template"
    inHierarchical="no">
</lsResolveTemplates>
```

Response

```
<lsResolveTemplates
    dn="org-root/org-level1"
    cookie="<real_cookie>"
    response="yes">
    <outConfigs>
       <pair key="ls-service-template-001">
          <lsServer agentPolicyName=""
             assignState="unassigned"
             assocState="unassociated"
             biosProfileName=""
             bootPolicyName=""
             configQualifier=""
             configState="not-applied"
             descr=""
             dn="org-root/ls-service-template-001"
             type="initial-template"
             usrLbl=""
             uuid="derived"
             uuidSuffix="0000-000000000000"
             vconProfileName=""/>
       </pair>
    </outConfigs>
</lsResolveTemplates>
```

lsTemplatise

The lsTemplatise method creates a template from a specified service profile.

Request Syntax

```
<xs:element name="lsTemplatise" type="lsTemplatise" substitutionGroup="externalMethod"/>
        <xs:complexType name="lsTemplatise" mixed="true">
            <xs:attribute name="inTargetOrg" type="referenceObject"/>

            <xs:attribute name="inTemplateName">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inTemplateType">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="instance"/>
                         <xs:enumeration value="initial-template"/>
                         <xs:enumeration value="updating-template"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="lsTemplatise" type="lsTemplatise" substitutionGroup="externalMethod"/>
        <xs:complexType name="lsTemplatise" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<lsTemplatise
    dn="org-root/ls-SP1"
    cookie="<real_cookie>"
    inTargetOrg="org-root"
    inTemplateName="tempate-2"
    inTemplateType="initial-template"
    inHierarchical="no">
</lsTemplatise>
```

Response

```
<lsTemplatise
    dn="org-root/ls-SP1"
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfig>
       <lsServer
          agentPolicyName=""
          assignState="unassigned"
          assocState="unassociated"
          biosProfileName=""
          bootPolicyName=""
          configQualifier=""
          configState="not-applied"
          descr=""
          dn="org-root/ls-tempate-2"
          dynamicConPolicyName=""
          extIPState="none"
          fltAggr="0"
          hostFwPolicyName=""
          identPoolName="default"
          intId="52339"
          localDiskPolicyName="default"
          maintPolicyName=""
          mgmtAccessPolicyName=""
          mgmtFwPolicyName=""
          name="tempate-2"
          operBiosProfileName=""
          operBootPolicyName=""
          operDynamicConPolicyName=""
          operHostFwPolicyName=""
          operIdentPoolName=""
          operLocalDiskPolicyName=""
          operMaintPolicyName=""
          operMgmtAccessPolicyName=""
          operMgmtFwPolicyName=""
          operPowerPolicyName=""
          operScrubPolicyName=""
          operSolPolicyName=""
          operSrcTemplName=""
          operState="unassociated"
          operStatsPolicyName=""
          operVconProfileName=""
          owner="management"
```

```
            pnDn=""
            powerPolicyName="default"
            scrubPolicyName=""
            solPolicyName=""
            srcTemplName="service-templ-001"
            statsPolicyName="default"
            status="created"
            type="initial-template"
            usrLbl=""
            uuid="derived"
            uuidSuffix="0000-000000000000"
            vconProfileName=""/>
      </outConfig>
</lsTemplatise>
```

lstorageCreateZoningFromInv

The `lstorageCreateZoningFromInv` method is used to create zoning policy from inventory chassis.

Request Syntax

```
<xs:element name="lstorageCreateZoningFromInv" type="lstorageCreateZoningFromInv"
 substitutionGroup="externalMethod"/>
     <xs:complexType name="lstorageCreateZoningFromInv" mixed="true">
          <xs:attribute name="inChassisDn" type="referenceObject"/>
          <xs:attribute name="inTargetOrg" type="referenceObject"/>
          <xs:attribute name="inDiskZoningPolicyName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="lstorageCreateZoningFromInv" type="lstorageCreateZoningFromInv"
 substitutionGroup="externalMethod"/>
     <xs:complexType name="lstorageCreateZoningFromInv" mixed="true">
          <xs:all>
               <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

```
<lstorageCreateZoningFromInv
cookie="<real cookie>">
</lstorageCreateZoningFromInv>
```

Response

```
<lstorageCreateZoningFromInv
cookie="<real cookie>"
response="yes" errorCode="102" invocationResult="unidentified-fail"
errorDescr="chassis profile is unresolvable">
</lstorageCreateZoningFromInv>
```

methodResolveVessel

The `methodResolveVessel` method is used to send single request for multiple retrieval methods.

Request Syntax

```
<xs:element name="methodResolveVessel" type="methodResolveVessel"
substitutionGroup="externalMethod"/>
     <xs:complexType name="methodResolveVessel" mixed="true">
          <xs:all>
               <xs:element name="inStimuli" type="MethodSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="methodResolveVessel" type="methodResolveVessel"
substitutionGroup="externalMethod"/>
     <xs:complexType name="methodResolveVessel" mixed="true">
          <xs:all>
               <xs:element name="outStimuli" type="MethodSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

Response

methodVessel

The `methodVessel` method is used to send single request for multiple retrievals and update methods.

Request Syntax

```
<xs:element name="methodVessel" type="methodVessel"
substitutionGroup="externalMethod"/>
     <xs:complexType name="methodVessel" mixed="true">
          <xs:all>
               <xs:element name="inStimuli" type="MethodSet" minOccurs="0"/>
          </xs:all>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="methodVessel" type="methodVessel"
substitutionGroup="externalMethod"/>
     <xs:complexType name="methodVessel" mixed="true">
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

Response

orgResolveElements

The `orgResolveElements` method resolves the instance of the policy class using a specified organization, policy class ID, and name.

Request Syntax

```
<xs:element name="orgResolveElements" type="orgResolveElements"
substitutionGroup="externalMethod"/>
        <xs:complexType name="orgResolveElements" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inClass" type="namingClassId"/>
            <xs:attribute name="inSingleLevel">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
```

```
                              </xs:restriction>
                          </xs:simpleType>
                     </xs:union>
                 </xs:simpleType>
             </xs:attribute>
             <xs:attribute name="inHierarchical">
                 <xs:simpleType>
                     <xs:union memberTypes="xs:boolean">
                          <xs:simpleType>
                              <xs:restriction base="xs:string">
                                  <xs:enumeration value="no"/>
                                  <xs:enumeration value="yes"/>
                              </xs:restriction>
                          </xs:simpleType>
                     </xs:union>
                 </xs:simpleType>
             </xs:attribute>
             <xs:attribute name="cookie" type="xs:string"/>
             <xs:attribute name="response" type="YesOrNo"/>
             <xs:attribute name="dn" type="referenceObject"/>
         </xs:complexType>
```

Response Syntax

```
<xs:element name="orgResolveElements" type="orgResolveElements"
substitutionGroup="externalMethod"/>
        <xs:complexType name="orgResolveElements" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_11">
                         <xs:selector xpath="pair"/>
                         <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<orgResolveElements
    dn="org-root/org-Soda"
       cookie="<real_cookie>"
    commCookie="7/15/0/19"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    inClass="policyPolicySet"
    inSingleLevel="no"
    inHierarchical="no">
    <inFilter>
    </inFilter>
</orgResolveElements>
```

Response

```
<orgResolveElements
    dn="org-root/org-Soda"
    cookie="<real_cookie>"
    commCookie="7/15/0/19"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="policy-mgr_dme"
    response="yes"
    errorCode="0"
    errorDescr="">
    <outConfigs>
       <pair key="pset-default">
          <policyPolicySet
             adminState="enabled"
             descr="The default Policy Set"
             dn="org-root/pset-default"
             intId="10082"
             name="default"/>
       </pair>
       <pair key="pset-myPolicySet3">
            .
       </pair>
       <pair key="pset-policySetSanity">
          <policyPolicySet
             adminState="enabled"
             descr=""
             dn="org-root/org-Soda/pset-policySetSanity"
             intId="24627"
             name="policySetSanity"/>
       </pair>
       <pair key="pset-pci_compliance_f">
          <policyPolicySet
             adminState="enabled"
             descr=""
             dn="org-root/pset-pci_compliance_f"
             intId="24539"
             name="pci_compliance_f"/>
       </pair>
       <pair key="pset-pci_compliance_h">
          <policyPolicySet
             adminState="enabled"
             descr=""
             dn="org-root/pset-pci_compliance_h"
             intId="24541"
             name="pci_compliance_h"/>
       </pair>
    </outConfigs>
 </orgResolveElements>
```

poolResolveInScope

The `poolResolveInScope` method, using the specified DN, looks up the pool and parent pools (optional) recursively to the root. If no pool exists, an empty map is returned. If any pool is found, this method searches all pools with the specified class and filters.

Note If inSingleLevel = false, this method searches parent pools up to the root directory.

Request Syntax

```
<xs:element name="poolResolveInScope" type="poolResolveInScope"
substitutionGroup="externalMethod"/>
        <xs:complexType name="poolResolveInScope" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inClass" type="namingClassId"/>
            <xs:attribute name="inSingleLevel">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean">
                         <xs:simpleType>
                             <xs:restriction base="xs:string">
                                 <xs:enumeration value="no"/>
                                 <xs:enumeration value="yes"/>
                             </xs:restriction>
                         </xs:simpleType>
                    </xs:union>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="poolResolveInScope" type="poolResolveInScope"
substitutionGroup="externalMethod"/>
        <xs:complexType name="poolResolveInScope" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configMap" minOccurs="0">
                    <xs:unique name="unique_map_key_13">
```

```
                        <xs:selector xpath="pair"/>
                        <xs:field xpath="@key"/>
                    </xs:unique>
                </xs:element>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<poolResolveInScope
    dn="org-root/org-Cisco"
    cookie="<real_cookie>"
    class=fwPool />
```

Response

```
<poolResolveInScope
    dn="org-root/org-Cisco"
    cookie="<real_cookie>"
    commCookie="5/15/0/5bf"
    srcExtSys="10.193.33.221"
    destExtSys="10.193.33.221"
    srcSvc="sam_extXMLApi"
    destSvc="resource-mgr_dme"
    response="yes">
    <outConfigs>
       <pair key="fwpool-default">
          <fwPool
             assigned="0"
             descr="Default Pool of Virtual Security Gateway resources"
             dn="org-root/fwpool-default"
             fltAggr="65536"
             id="1"
             intId="10065"
             name="default"
             size="0"/>
       </pair>
       <pair key="fwpool-ciscoCfwPool">
          .
       </pair>
    </outConfigs>
</poolResolveInScope>
```

statsClearInterval

The `statsClearInterval` method resets the collection interval timer for the `statsClass`. All of the statistics' implicit properties (for example, min, max, and avg calculations) are reset, and the corresponding history properties are updated. The interval updates restart from 1, and the stats collection is reset.

Request Syntax

```
<xs:element name="statsClearInterval" type="statsClearInterval"
substitutionGroup="externalMethod"/>
        <xs:complexType name="statsClearInterval" mixed="true">
            <xs:all>
                <xs:element name="inDns" type="dnSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="statsClearInterval" type="statsClearInterval"
substitutionGroup="externalMethod"/>
        <xs:complexType name="statsClearInterval" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<statsClearInterval
    cookie="<real_cookie>">
    <inDns>
       <dn value="sys/chassis-1/blade-1/board/temp-stats"/>
    </inDns>
</statsClearInterval>
```

Response

```
<statsClearInterval
    cookie="<real_cookie>"
    response="yes"
    errorCode="0"
    errorDescr="">
</statsClearInterval>
```

statsResolveThresholdPolicy

The `statsResolveThresholdPolicy` method resolves threshold policy based on the container class ID. The container class is objects with policies (for example, server domain, LAN cloud, and SAN cloud). Cisco UCS uses the hierarchy of an organization to resolve the names of policies.

Request Syntax

```
<xs:element name="statsResolveThresholdPolicy" type="statsResolveThresholdPolicy"
    substitutionGroup="externalMethod"/>
    <xs:complexType name="statsResolveThresholdPolicy" mixed="true">
       <xs:attribute name="cookie" type="xs:string"/>
       <xs:attribute name="response" type="YesOrNo"/>
       <xs:attribute name="dn" type="referenceObject"/>
 </xs:complexType>
```

Response Syntax

```
<xs:element name="statsResolveThresholdPolicy" type="statsResolveThresholdPolicy"
substitutionGroup="externalMethod"/>
 <xs:complexType name="statsResolveThresholdPolicy" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
 <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<statsResolveThresholdPolicy
    dn="sys/chassis-1/blade-5/adaptor-1/ext-eth-1/eth-port-stats-rx"
    cookie="<real_cookie>">
</statsResolveThresholdPolicy>
```

Response

```
<statsResolveThresholdPolicy
    dn="sys/chassis-1/blade-3/adaptor-1/ext-eth-1/eth-port-stats-rx"
    cookie="<real_cookie>"
    response="yes">
    <outConfig>
       <statsThresholdPolicy
          childAction="deleteNonPresent"
          descr="" dn="fabric/lan/thr-policy-default"
          intId="20243"
          name="default"  >
          <statsThresholdClass
             childAction="deleteNonPresent"
             descr=""
             intId="32274"
             name="" rn="adaptorEthPortStats"
             statsClassId="adaptorEthPortStats" >
             <statsThr64Definition
                childAction="deleteNonPresent"
                descr=""
                intId="32275"
                name=""
                normalValue="1"
                propId="adaptorEthPortStatstotalPacketsDelta"
                propType="uint64"
                rn="adaptorEthPortStatstotalPacketsDelta" />
          </statsThresholdClass>
       </statsThresholdPolicy>
    </outConfig>
</statsResolveThresholdPolicy>
```

trigPerformTokenAction

The `trigPerformTokenAction` method

Request Syntax

```
<xs:element name="trigPerformTokenAction" type="trigPerformTokenAction"
substitutionGroup="externalMethod"/>
     <xs:complexType name="trigPerformTokenAction" mixed="true">
          <xs:attribute name="inTokenAction">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="request"/>
                         <xs:enumeration value="refresh"/>
                         <xs:enumeration value="release"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inContext" type="referenceObject"/>
          <xs:attribute name="inTokenId" type="xs:unsignedLong"/>
          <xs:attribute name="inSchedName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inWindowName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inWindowType">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="absolute"/>
                         <xs:enumeration value="recurring"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="inTriggerableDn" type="referenceObject"/>
          <xs:attribute name="inOwnership">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="local"/>
                         <xs:enumeration value="pending-policy"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
     </xs:complexType>
```

Response Syntax

```
<xs:element name="trigPerformTokenAction" type="trigPerformTokenAction"
substitutionGroup="externalMethod"/>
     <xs:complexType name="trigPerformTokenAction" mixed="true">
          <xs:attribute name="outWindowName">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="outWindowType">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="absolute"/>
                         <xs:enumeration value="recurring"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="outOldTokenId" type="xs:unsignedLong"/>
          <xs:attribute name="outNewTokenId" type="xs:unsignedLong"/>
          <xs:attribute name="outOldTriggerableDn" type="referenceObject"/>
          <xs:attribute name="outLastTokenOperation">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="request"/>
                         <xs:enumeration value="refresh"/>
                         <xs:enumeration value="release"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="outOwnership">
               <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="local"/>
                         <xs:enumeration value="policy"/>
                         <xs:enumeration value="pending-policy"/>
                    </xs:restriction>
               </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="cookie" type="xs:string"/>
          <xs:attribute name="response" type="YesOrNo"/>
          <xs:attribute name="errorCode" type="xs:unsignedInt"/>
          <xs:attribute name="errorDescr" type="xs:string"/>
          <xs:attribute name="invocationResult" type="xs:string"/>
     </xs:complexType>
```

Examples

Request

Response