Cisco UCS Central 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 Users with admin or aaa privilege are not required to provide the old password while using this method.

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: Changing Your Password

Request

```
<aaaChangeSelfPassword
    cookie="<real_cookie>"
    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>
```

Example: Verifying a User Token

Request

```
<aaaCheckComputeAuthToken
    cookie="<real_cookie>"
    inToken="045418753093022996872l1"
    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>
```

Example: Validating User Access for a Specified Server

Request

```
<aaaCheckComputeExtAccess
    cookie="<real_cookie>"
    inDn="sys/Chassis-1/blade-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>
```

Example: Requesting Token Information

Request

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

Response

```
<aaaGetNComputeAuthTokenByDn
cookie="<real_cookie>"
response="yes"
outUser="__computeToken__"
outTokens="3550599419521612726721 1,935955519085270602324 51,11769973096057301593991,527
29538672765491844031,73106643969990280919791">
</aaaGetNComputeAuthTokenByDn>
```

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>
```

Example: Keeping a Session Active

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 Central.

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="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: Logging In

Request

```
<aaaLogin
    inName="admin"
    inPassword="RU123B45"/>
```

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)">
</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>
```

Example: Logging Out

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

Central 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.

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:complexType>
```

Example: Refreshing the Session

Request

```
<aaaRefresh
    cookie="<real_cookie>"
    inName="admin"
    inPassword="RU123B45"
    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">
</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>
```

Example: Logging in with a Token

Request

```
<aaaTokenLogin
    inName="admin"
    inToken="8027850296441080579135 1" />
```

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.

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>
```

Example: Refreshing the Session with a Token

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>
```

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>
```

Example: Configuring Specific Objects

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>
```

Example: Tagging Domains

When you schedule infrastructure firmware updates, you can schedule them for specific domains, or domains assigned to a domain group, using maintenance groups and tags.

You can apply a maintenance group tag to a domain with configConfMo. See the UCS Central Administration Guide for more information on Infrastructure Firmware updates.

Request

```
<configConfMo
    dn="holder/tag-ep"
    cookie="<real_cookie>"
    inHierarchical="false">
    <inConfig>
       <tagInstance
          defName="Maintenance Group"       // Type of tag
          taggedObjectDn="compute/sys-1009" // Domain name or ID
          value="tagTest"                   // Tag name
          status="created"
          />
    </inConfig>
</configConfMo>
```

Response

```
<configConfMo
  dn="holder/tag-ep"
  cookie="<real_cookie>"
  commCookie="18/16/0/a4a"
  userContext="no"
  srcExtSys="10.193.219.120"
  destExtSys="10.193.219.120"
  srcSvc="sam_extXMLApi"
  destSvc="central-mgr_dme"
  response="yes">
<outConfig>
<tagInstance
  defDn="holder/tag-def-ep/type-Maintenance Group"
  defName="Maintenance Group"
  dn="holder/tag-ep/type-Maintenance Group-inst-[tagTest]-obj-[compute/sys-1009]"
  objectName="StorMagicFI-A"
  objectType="computeSystem"
  srcDme="resource-mgr"
  status="created"
  taggedObjectDn="compute/sys-1009"
  value="tagTest"/>Config>
</configConfMo>
```

Creating a Hardware Compatibility List

Creating the Hardware Compatibility list involves multiple steps:

1. Creating an OS tag on the target server.
2. Creating an Adapter Driver tag on the target server.
3. Deleting the tags, when finished.

Example: Creating an OS tag on the Target Server

The Hardware Compatibility report queries the OS tagSoftwareInst tag.

To find the OS vendor and version, use configResolveClassIdx. For more information, see Example: Verifying the OS Vendor and Version, on page 33.

Request

```
<configConfMo
dn="holder/tag-ep"
cookie="<real_cookie>"
inHierarchical="false">
  <inConfig>
    <tagSoftwareInst
    defName="Operating System for HCR"      //required field
    taggedObjectDn="compute/sys1/ch1/bl3"   //required field; DN for the server
    value=""
    vendor="CentOS"                //required field; UCS Central OS
    version="CentOS 6.6"           //required field; UCS Central OS version
    status="created"
    />
    </inConfig>
</configConfMo>
```

Response

```
<configConfMo
dn="holder/tag-ep"
cookie="<real_cookie>"
commCookie="18/16/0/47b"
userContext="no"
srcExtSys="10.193.219.120"
destExtSys="10.193.219.120"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
    <outConfig>
        <tagSoftwareInst
  defDn="holder/tag-def-ep/type-Operating System for HCR"
  defName="Operating System for HCR"
  dn="holder/tag-ep/type-Operating System for HCR-inst-[CentOS
6.6]-obj-[compute/sys1/ch1/bl3]"
  objectName=""
  objectType="computeBlade"
  srcDme="resource-mgr"
  status="created"
  taggedObjectDn="compute/sys1/ch1/bl3"
  value="CentOS 6.6"
  vendor="CentOS"
  version="CentOS 6.6"/>
    </outConfig>
</configConfMo>
```

Example: Creating an Adapter tag on the Target Server

The Hardware Compatibility report queries the Adapter Driver tagDriver tag.

Note If you need to verify different driver types, create multiple tagDrivers.

To find the adapter vendor and version, use configResolveClassIdx. For more information, see Example: Verifying the Adapter Vendor and Version, on page 34.

Request

```
<configConfMo
dn="holder/tag-ep"
cookie="<real_cookie>"
inHierarchical="false">
  <inConfig>
    <tagDriver
       defName='Adapter Driver for HCR' //required field
       taggedObjectDn='compute/sys1/ch1/bl4/ad1'  //required field
       protocol='Ethernet'
       vendor='Cisco' //required field; adapter vendor
       version='2.3.0.20' //required field; adapter driver version
       value=''
       status='created'
      />
    </inConfig>
</configConfMo>
```

Response

```
<configConfMo
dn="holder/tag-ep"
cookie="<real_cookie>"
```

```
commCookie="18/16/0/480"
userContext="no"
srcExtSys="10.193.219.120"
destExtSys="10.193.219.120"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
    <outConfig>
        <tagDriver
  defDn="holder/tag-def-ep/type-Adapter Driver for HCR"
  defName="Adapter Driver for HCR"
  dn="holder/tag-ep/type-Adapter Driver for HCR-inst-[Cisco Ethernet
2.3.0.20]-obj-[compute/sys1/ch1/bl4/ad1]"
  objectName=""
  objectType="adaptorUnit"
  protocol="Ethernet"
  srcDme="resource-mgr"
  status="created"
  taggedObjectDn="compute/sys1/ch1/bl4/ad1"
  value="Cisco Ethernet 2.3.0.20"
  vendor="Cisco"
  version="2.3.0.20"/>
    </outConfig>
</configConfMo>
```

Example: Deleting an OS Tag

Request

```
<configConfMo
cookie="<real_cookie>"
 <inConfig>
<tagSoftwareInst dn="compute/sys1/chassis-1/blade-3" status='deleted' > //<dn of OsTag>
</tagSoftwareInst>
</inConfig>
</configConfMo>
```

Response

```
<configConfMo
dn=""
cookie="<real_cookie>"
commCookie="18/16/0/46b"
userContext="no"
srcExtSys="10.193.219.120"
destExtSys="10.193.219.120"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
<outConfig>
</outConfig>
</configConfMo>
```

Example: Deleting an Adapter Tag

Request

```
<configConfMo
cookie="<real_cookie>"
 <inConfig>
<tagDriver dn='compute/sys1/chassis-1/blade-4/adaptor-1'> // <dn of DriverTag>
 status='deleted'
</ tagDriver >
```

```
</inConfig>
</configConfMo>
```

Response

```
<configConfMo
dn=""
cookie="<real_cookie>"
commCookie="18/16/0/46b7"
userContext="no"
srcExtSys="10.193.219.120"
destExtSys="10.193.219.120"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
<outConfig>
</outConfig>
</configConfMo>
```

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>
```

Example: Configuring Managed Objects in Multiple Subtrees

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>
```

configUCEstimateImpact

The configUCEstimateImpact 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.

The user can estimate the impact of a change set by passing the set to the method and checking the corresponding ImpactAnalyzer object. The output parameter is the DN of a corresponding ImpactAnalyzer object for the change that the user passed down. The user use the configMoChangeEvent of the ImpactAnalyzer object. Once the state of this ImpactAnalyzer object is 'complete', the user can resolve the ImpactAnalyzer object to estimate impact results.

Estimate results contain the following information:

- Whether the changes are disruptive (for example, if the action requires a reboot of the server associated to the service profile).
- Whether the changes result in configuration issues; and the type of configuration issues.
- Number of UCS domains analyzed.
- Number of UCS domains affected.
- Number of suspended UCS domains.
- Number of UCS domains that lose visibility.
- Number of UCS domains that time out.
- Number of servers affected.
- Summary of changes.
- Time when the changes are applied (For example, immediately, after user acknowledgment, or during the scheduled occurrence of a maintenance window).

The parameters are defined as:

- configs—Set of changes to be evaluated (add, delete, or modify managed objects).
- ImpactAnalyzerID (optional)
- outImpactAnalyzerDN—DN of the corresponding ImpactAnalyzer object.

Request Syntax

```
<xs:element name="configUCEstimateImpact" type="configUCEstimateImpact"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configUCEstimateImpact" 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="configUCEstimateImpact" type="configUCEstimateImpact"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configUCEstimateImpact" 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=" ImpactAnalyzerDn " type=" referenceObject "/>
        </xs:complexType>
```

Example: Estimating the Impact of a Modification

Request

```
<configUCEstimateImpact
cookie="<real_cookie>"
inImpactAnalyzerId="0">
   <inConfigs>
<pair key="org-root/org-orgs/ls-gsp">
   <lsServer
   biosProfileName="global-SRIOV"
   dn="org-root/org-orgs/ls-gsp"
   status="modified"/>
</pair>
   </inConfigs>
</configUCEstimateImpact>
```

Response

```
<configUCEstimateImpact
cookie="<real_cookie>"
response="yes"
errorCode="0"
errorDescr=""
outImpactAnalyzerDn="impactanalyzer-ep/impact-analyzer-1401723100">
</configUCEstimateImpact>
```

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>
```

Example: Finding Device Policy Details

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>
```

configMoChangeEvent

The configMoChangeEvent method provides event details from Cisco UCS Central 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 Central 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>
```

Example: Providing Event Details to Subscribers

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.

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>
```

Example: Computing the Scrub Policy Effect

Request

```
<configResolveChildren
cookie="<real_cookie>""

classId="computeScrubPolicy"
inDn="org-root"
inHierarchical="false">
    <inFilter>
    </inFilter>
</configResolveChildren>
```

Response

```
<configResolveChildren
cookie="<real_cookie>""
commCookie="7716/0/25c"
srcExtSys="10.193.219.18"
destExtSys="10.193.219.18"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes"
classId="computeScrubPolicy">
  <outConfigs>

<computeScrubPolicy
biosSettingsScrub="no"
descr=""
diskScrub="no"
dn="org-root/scrub-global-default"
flexFlashScrub="no"
intId="10238"
name="global-default"
policyLevel="0"
policyOwner="local"/>
  </outConfigs>
</configResolveChildren>
```

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>
```

Example: Searching for Organization Information

Request

```
<configResolveClass
cookie="<real_cookie>"

classId="orgOrg"
inHierarchical="false">
    <inFilter>
<eq class="orgOrg" property="level" value="root"/>
    </inFilter>
```

```
</configResolveClass>
```

Response

```
<configResolveClass
cookie="<real_cookie>"
commCookie="7/16/0/259"
srcExtSys="10.193.219.18"
destExtSys="10.193.219.18"
srcSvc="sam_extXMLApi"
destSvc="policy-mgr_dme"
response="yes"
classId="orgOrg">
  <outConfigs>

<orgOrg
descr=""
dn="org-root"
fltAggr="0"
level="root"
name="root"/>
  </outConfigs>
</configResolveClass>
```

configResolveClassIdx

The configResolveClass method returns requested managed object in a given class.

Request Syntax

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

        <xs:complexType name="configResolveClassIdx" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>

            <xs:attribute name="inQuery">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>

            <xs:attribute name="inClass">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>

            <xs:attribute name="inParentDn">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
```

```
            <xs:attribute name="inSortStr">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>

            <xs:attribute name="inIncludeProp">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </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="inOffset" type="xs:unsignedShort"/>

            <xs:attribute name="inLimit" 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

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

<xs:element name="outConfigs" type="configSet" minOccurs="0"/>
</xs:all>

<xs:attribute name="outTotalCount" 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:attribute name="classId" type="namingClassId"/>
</xs:complexType>
```

Example: Verifying the OS Vendor and Version

Request

```
<configResolveClassIdx
cookie='<real_cookie>'
inQuery='*:*'
inClass='hcOsInfoItem'
inSortStr='version+asc'
inOffset='0'
inLimit='100000'
/>
```

Response

```
<configResolveClassIdx outTotalCount="141">
    <outConfigs>
        <hcOsInfoItem vendor="CentOS" id="807" version="CentOS 6.4"/>
        <hcOsInfoItem vendor="CentOS" id="802" version="CentOS 6.5"/>
        <hcOsInfoItem vendor="CentOS" id="803" version="CentOS 6.6"/>
        <hcOsInfoItem vendor="CentOS" id="804" version="CentOS 6.7"/>
        <hcOsInfoItem vendor="CentOS" id="806" version="CentOS 7.0"/>
        <hcOsInfoItem vendor="CentOS" id="801" version="CentOS 7.1"/>
        <hcOsInfoItem vendor="CentOS" id="805" version="CentOS 7.2"/>
        <hcOsInfoItem vendor="Oracle" id="202" version="OL 7.0 UEK R3 U3"/>
        <hcOsInfoItem vendor="Oracle" id="2024" version="OL 7.1 UEK R3 U6"/>
        <hcOsInfoItem vendor="Oracle" id="2030" version="OL 7.1 UEK R3 U7"/>
        <hcOsInfoItem vendor="Oracle" id="2022" version="OL 7.1 UEK R4"/>
        <hcOsInfoItem vendor="Oracle" id="2023" version="OL 7.2 UEK R4"/>
        <hcOsInfoItem vendor="Oracle" id="2032" version="OL 7.2 UEK R4"/>
        <hcOsInfoItem vendor="Oracle" id="207" version="OVM 3.3.2"/>
        <hcOsInfoItem vendor="Oracle" id="2011" version="OVM 3.3.3"/>
        <hcOsInfoItem vendor="Oracle" id="2037" version="OVM 3.3.4"/>
        <hcOsInfoItem vendor="Oracle" id="2025" version="OVM 3.4"/>
        <hcOsInfoItem vendor="Red Hat" id="3018" version="Red Hat Enterprise Linux 6.0"/>
        <hcOsInfoItem vendor="Red Hat" id="3017" version="Red Hat Enterprise Linux 6.1"/>
        <hcOsInfoItem vendor="Red Hat" id="3011" version="Red Hat Enterprise Linux 6.2"/>
        <hcOsInfoItem vendor="Red Hat" id="308" version="Red Hat Enterprise Linux 6.3"/>
        <hcOsInfoItem vendor="Red Hat" id="302" version="Red Hat Enterprise Linux 6.4"/>
        <hcOsInfoItem vendor="Red Hat" id="301" version="Red Hat Enterprise Linux 6.5"/>
        <hcOsInfoItem vendor="Red Hat" id="305" version="Red Hat Enterprise Linux 6.6"/>
        <hcOsInfoItem vendor="Red Hat" id="3010" version="Red Hat Enterprise Linux 6.7"/>
        <hcOsInfoItem vendor="Red Hat" id="306" version="Red Hat Enterprise Linux 7.0"/>
        <hcOsInfoItem vendor="Red Hat" id="309" version="Red Hat Enterprise Linux 7.1"/>
        <hcOsInfoItem vendor="Red Hat" id="3013" version="Red Hat Enterprise Linux 7.2"/>
        <hcOsInfoItem vendor="SuSE" id="501" version="SUSE Linux Enterprise Server 11.3"/>
        <hcOsInfoItem vendor="SuSE" id="504" version="SUSE Linux Enterprise Server 11.4"/>
        <hcOsInfoItem vendor="SuSE" id="502" version="SUSE Linux Enterprise Server 12"/>
        <hcOsInfoItem vendor="SuSE" id="505" version="SUSE Linux Enterprise Server 12.1"/>
        <hcOsInfoItem vendor="Oracle" id="2043" version="Solaris 10 (U8)"/>
        <hcOsInfoItem vendor="Oracle" id="2026" version="Solaris 10 (U9)"/>
        <hcOsInfoItem vendor="Oracle" id="2029" version="Solaris 11"/>
        <hcOsInfoItem vendor="Oracle" id="2016" version="Solaris 11.1"/>
        <hcOsInfoItem vendor="Ubuntu" id="702" version="Ubuntu 14.04.2"/>
        <hcOsInfoItem vendor="Ubuntu" id="704" version="Ubuntu 14.04.3"/>
        <hcOsInfoItem vendor="Ubuntu" id="707" version="Ubuntu 14.04.4"/>
        <hcOsInfoItem vendor="Microsoft" id="406" version="Windows Server 2008"/>
        <hcOsInfoItem vendor="Microsoft" id="405" version="Windows Server 2008 R2"/>
        <hcOsInfoItem vendor="Microsoft" id="401" version="Windows Server 2008 R2 SP1"/>
        <hcOsInfoItem vendor="Microsoft" id="409" version="Windows Server 2008 SP1"/>
        <hcOsInfoItem vendor="Microsoft" id="404" version="Windows Server 2008 SP2"/>
        <hcOsInfoItem vendor="Microsoft" id="402" version="Windows Server 2012"/>
        <hcOsInfoItem vendor="Microsoft" id="403" version="Windows Server 2012 R2"/>
        <hcOsInfoItem vendor="Citrix" id="603" version="XenServer 6.1"/>
        <hcOsInfoItem vendor="Citrix" id="601" version="XenServer 6.2"/>
        <hcOsInfoItem vendor="Citrix" id="602" version="XenServer 6.5"/>
```

```
        <hcOsInfoItem vendor="Citrix" id="608" version="XenServer 6.5 SP1"/>
        <hcOsInfoItem vendor="VMware" id="1015" version="vSphere 5.1"/>
        <hcOsInfoItem vendor="VMware" id="106" version="vSphere 5.1 U1"/>
        <hcOsInfoItem vendor="VMware" id="102" version="vSphere 5.1 U2"/>
        <hcOsInfoItem vendor="VMware" id="104" version="vSphere 5.1 U3"/>
        <hcOsInfoItem vendor="VMware" id="101" version="vSphere 5.5"/>
        <hcOsInfoItem vendor="VMware" id="107" version="vSphere 5.5 U1"/>
        <hcOsInfoItem vendor="VMware" id="105" version="vSphere 5.5 U2"/>
        <hcOsInfoItem vendor="VMware" id="1011" version="vSphere 5.5 U3"/>
        <hcOsInfoItem vendor="VMware" id="1010" version="vSphere 6.0"/>
        <hcOsInfoItem vendor="VMware" id="1012" version="vSphere 6.0 U1"/>
        <hcOsInfoItem vendor="VMware" id="1017" version="vSphere 6.0 U2"/>
    </outConfigs>
</configResolveClassIdx>
```

Example: Verifying the Adapter Vendor and Version

Request

```
<configResolveClassIdx
cookie='<real_cookie>'
inQuery='*:*'
inClass='hcDriverInfoItem'
inSortStr='version+asc'
inOffset='0'
inLimit='100000'
/>
```

Response

```
<configResolveClassIdx outTotalCount="3940">
    <outConfigs>
        <hcDriverInfoItem adapterPid="N20-AC162-M4" vendor="Cisco" id="86000100203"
protocol="SNIC"
                          version="0.0.1.19"/>
        <hcDriverInfoItem adapterPid="N20-AC162-M4" vendor="Cisco" id="86000100202"
protocol="SNIC"
                          version="0.0.1.22"/>
        <hcDriverInfoItem adapterPid="N20-AC162-M4" vendor="Cisco" id="86000100201"
protocol="SNIC"
                          version="0.0.1.26"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE1600GB" vendor="Cisco" id="83000100203"
                          protocol="NVMe PCIe SSD"
                          version="0.8"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE800GB" vendor="Cisco" id="84000100203"
                          protocol="NVMe PCIe SSD"
                          version="0.8"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE1600GB" vendor="Cisco" id="83000100102"
protocol=" "
                          version="0.8 ()"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE800GB" vendor="Cisco" id="84000100102"
protocol=" "
                          version="0.8 ()"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE1600GB" vendor="Cisco" id="83000100202"
                          protocol="NVMe PCIe SSD"
                          version="0.9"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE800GB" vendor="Cisco" id="84000100202"
                          protocol="NVMe PCIe SSD"
                          version="0.9"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE1600GB" vendor="Cisco" id="83000100103"
protocol=" "
                          version="0.9 ()"/>
        <hcDriverInfoItem adapterPid="UCS-SDHPCIE800GB" vendor="Cisco" id="84000100103"
protocol=" "
                          version="0.9 ()"/>
        <hcDriverInfoItem adapterPid="R2XX-PL002" vendor="LSI Logic" id="850001001013"
```

```
protocol="RAID"
                        version="00.00.04.32.1vmw"/>
        <hcDriverInfoItem adapterPid="R2X0-ML002" vendor="LSI Logic" id="560001001013"
protocol="RAID"
                        version="00.00.05.30"/>
       <hcDriverInfoItem adapterPid="UCSC-RAID-SFFC200" vendor="LSI Logic" id="730001001023"

                        protocol="RAID"
                        version="00.00.05.30"/>
        <hcDriverInfoItem adapterPid="R2XX-PL002" vendor="LSI Logic" id="85000100101"
protocol="RAID"
                        version="00.00.05.30"/>
        <hcDriverInfoItem adapterPid="RC460-PL002" vendor="LSI Logic" id="260001001014"
protocol="RAID"
                        version="00.00.05.33"/>
        <hcDriverInfoItem adapterPid="UCS-RAID-9266" vendor="LSI Logic" id="100001001034"
protocol="RAID"
                        version="00.00.05.34"/>
       <hcDriverInfoItem adapterPid="UCS-RAID9286CV-8E" vendor="LSI Logic" id="180001001053"

                        protocol="RAID"
                        version="00.00.05.34"/>
    </outConfigs>
</configResolveClassIdx>
```

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>
```

Example: Resolving Hardware Information

Request

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

Response

```
<configResolveClasses
cookie="<real_cookie>"
commCookie="5/16/0/7f2"
srcExtSys="10.193.219.12"
destExtSys="10.193.219.12"
srcSvc="sam_extXMLApi"
destSvc="resource-mgr_dme"
response="yes">
  <outConfigs>

<equipmentChassis
adminState="acknowledged"
configState="ok"
connPath="A,B"
connStatus="A,B"
dn="compute/sys-1008/chassis-1"
.
./>

<equipmentChassis
adminState="acknowledged"
configState="ok"
connPath="A,B"
connStatus="A,B"
dn="compute/sys-1009/chassis-1"
.
./>

<computeRackUnit
adminPower="policy"
adminState="in-service"
assignedToDn=""
association="none"
availability="available"
availableMemory="16384"
checkPoint="discovered"
connPath="A,B"
connStatus="A,B"
```

```
descr=""
discovery="complete"
dn="compute/sys-1009/rack-unit-3"
.
./>

<computeRackUnit
adminPower="policy"
adminState="in-service"
.
./>
  </outConfigs>
</configResolveClasses>
```

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>
```

Example: Searching for Hardware

Request

```
<configResolveDn
cookie="<real_cookie>"
inHierarchical="false"
dn="compute/sys-1009/chassis-1/blade-1"
/>
```

Response

```
<configResolveDn
dn="compute/sys-1009/chassis-1/blade-1"
cookie="<real_cookie>"
commCookie="18/16/0/802"
srcExtSys="10.193.219.12"
destExtSys="10.193.219.12"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
  <outConfig>
  <computeBlade
  adminPower="policy"
  adminState="in-service"
  assignedToDn=""
  association="none"
  availability="available"
  availableMemory="65536"
  chassisId="1"
  checkPoint="discovered"
  connPath="A,B"
  connStatus="A,B"
  descr=""
  discovery="complete"
  dn="compute/sys-1009/chassis-1/blade-1"
  .
  ./>
  </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>
```

Example: Searching for Multiple Blades

Request

```
<configResolveDns
cookie="<real cookie>"
inHierarchical="false">
<inDns>
<dn value="compute/sys-1009/chassis-1/blade-1" />
<dn value="compute/sys-1009/chassis-1/blade-2" />
</inDns>
</configResolveDns>
```

Response

```
<configResolveDns
cookie="<real_cookie>"
commCookie="18/16/0/806"
srcExtSys="10.193.219.12"
destExtSys="10.193.219.12"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
  <outConfigs>

<computeBlade
adminPower="policy"
adminState="in-service"
assignedToDn=""
```

```
association="none"
availability="available"
availableMemory="65536"
chassisId="1"
checkPoint="discovered"
connPath="A,B"
connStatus="A,B"
descr=""
discovery="complete"
dn="compute/sys-1009/chassis-1/blade-1"
.
./>

<computeBlade
adminPower="policy"
adminState="in-service"
assignedToDn=""
association="none"
availability="available"
availableMemory="65536"
chassisId="1"
checkPoint="discovered"
connPath="A,B"
connStatus="A,B"
descr=""
discovery="complete"
dn="compute/sys-1009/chassis-1/blade-2"
.
./>
  </outConfigs>
  <outUnresolved>
  </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>
```

Example: Locating a Blade

Request

```
<configResolveParent
cookie="<real_cookie>"
inHierarchical="false"
dn="compute/sys-1009/chassis-1/blade-1"/>
```

Response

```
<configResolveParent
dn="compute/sys-1009/chassis-1/blade-1"
cookie="<real_cookie>"
commCookie="18/16/0/809"
srcExtSys="10.193.219.12"
destExtSys="10.193.219.12"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
  <outConfig>
  <equipmentChassis
  adminState="acknowledged"
  configState="ok"
  connPath="A,B"
  connStatus="A,B"
  dn="compute/sys-1009/chassis-1"
  .
  ./>
  </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>
```

Example: Scoping a vNIC

Request

```
<configScope
dn="org-root/org-org-1/org-org-2"
cookie="<real_cookie>"
inClass="vnicLanConnPolicy"
inHierarchical="false"
inRecursive="false">
<inFilter>
</inFilter>
</configScope>
```

Response

```
<configScope
dn="org-root/org-org-1/org-org-2"
cookie="<real_cookie>"
commCookie="18/16/0/814"
srcExtSys="10.193.219.12"
destExtSys="10.193.219.12"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
<outConfigs>

<vnicLanConnPolicy
descr=""
dn="org-root/org-org-1/org-org-2/lan-conn-pol-lcp-1"
fltAggr="0"
intId="13667"
name="lcp-1"
policyLevel="3"
policyOwner="policy"/>

<vnicLanConnPolicy
descr=""
dn="org-root/org-org-1/org-org-2/lan-conn-pol-lcp-2"
fltAggr="0"
intId="13668"
name="lcp-2"
policyLevel="3"
policyOwner="policy"/>
</outConfigs>
</configScope>
```

eventSubscribe

The eventSubscribe method allows a client to subscribe to asynchronous events generated by Cisco UCS Central, 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 Central. When an event occurs, Cisco UCS Central 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.

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>
```

Example: Registering for an Event

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 Central, reversing event subscriptions that resulted from eventSubscribe.

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>
```

Example: Unregistering for an Event

Request

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

Response

```
NO RESPONSE OR ACKNOWLEDGMENT.
```

faultAckFaultByDn

The faultAckFaultByDn method acknowledges a fault using the DN as input. The acknowledgment response marks the fault severity as cleared. Faults categorized as auto-cleared do not require acknowledgment.

Request Syntax

```
<!--
   - Method: fault:AckFaultByDn
-->
<xs:element name="faultAckFaultByDn" type="faultAckFaultByDn"
substitutionGroup="externalMethod"/>

<xs:complexType name="faultAckFaultByDn" 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

```
<!--
   - Method: fault:AckFaultByDn
-->
<xs:element name="faultAckFaultByDn" type="faultAckFaultByDn"
substitutionGroup="externalMethod"/>

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

</xs:complexType>
```

Example: Acknowledging and Clearing a Fault

Request

```
<faultAckFaultByDn
cookie="<real_cookie>"
dn="sys/switch-A/stor-part-bootflash/fault-F10000336">
</faultAckFaultByDn>
```

Response

```
<faultAckFaultByDn
dn="sys/switch-A/stor-part-bootflash/fault-F10000336"
cookie="<real_cookie>"
commCookie="18/16/0/98e"
srcExtSys="10.193.219.18"
destExtSys="10.193.219.18"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
</faultAckFaultByDn>
```

faultAckFaultsByDn

The faultAckFaultsByDn method acknowledges multiple faults using DN as the input. The acknowledgment response marks the fault severity as cleared. Faults categorized as auto-cleared do not require acknowledgment.

Request Syntax

```
<!--
   - Method: fault:AckFaultsByDn
-->
<xs:element name="faultAckFaultsByDn" type="faultAckFaultsByDn"
substitutionGroup="externalMethod"/>

<xs:complexType name="faultAckFaultsByDn" 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="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>
```

Example: Acknowledging and Clearing Multiple Faults

Request

```
<faultAckFaultsByDn
cookie="<real_cookie>"
<inDns>
<dn
value="sys/corefiles/file-1401449743_SAM_kondal-vm-aub115_svc_centralMgr__log.11944.tar.gz/fault-F10000005"/>
<dn value="org-root/wwn-pool-global-node-default/fault-F10000200"/>
</inDns>
</faultAckFaultsByDn>
```

Response

```
<faultAckFaultsByDn
cookie="<real_cookie>"
commCookie="18/16/0/99c"
srcExtSys="10.193.219.18"
destExtSys="10.193.219.18"
srcSvc="sam_extXMLApi"
destSvc="central-mgr_dme"
response="yes">
</faultAckFaultsByDn>
```

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>
```

Example: Cloning a Service Profile

Request

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

Response

```
<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>
```

Example: Cloning a Service Profile Template

Request

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

Response

```
<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 creates the desired number of service profiles. This method uses the following parameters:

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

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>
```

Creating a Service Profiles from a Specific Template

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>
```

Example: Creating Multiple Service Profiles from a Template

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>
```

Example: Creating a Service Profile from a Specific Template

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>
```

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>
```

Example: Creating a Template from a Specific Service Profile

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>
```

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>
```

Example: Searching for Parent Directories of a Pool

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>
```