Cisco IMC XML API Method Descriptions

This chapter includes the following sections:

- aaaGetComputeAuthTokens, page 1
- aaaKeepAlive, page 2
- aaaLogin, page 3
- aaaLogout, page 4
- aaaRefresh, page 5
- configConfMo, page 7
- configConfMos, page 8
- configResolveChildren, page 10
- configResolveClass, page 11
- configResolveDn, page 12
- configResolveParent, page 13
- eventSubscribe, page 14
- eventUnsubscribe, page 15

aaaGetComputeAuthTokens

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

Note

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

Request Syntax

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

Response Syntax

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

Examples

Request:

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

Response:

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

aaaKeepAlive

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

Request Syntax

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

Response Syntax

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

Examples

Request

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

Response

```
<aaaKeepAlive cookie="<real_cookie>" 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 IMC.

Request Syntax

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogin" mixed="true">
            <xs:attribute name="inName" use="required">
                <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" use="required">
                <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="stringMin0Max47"/>
            <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="(read-only|admin|user){0,1}"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outSessionId">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="32"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outVersion" type="xs:string"/>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

```
<aaaLogin inName='admin' inPassword='password'/>
```

Response

```
<aaaLogin cookie="" response="yes" outCookie="<real_cookie>" outRefreshPeriod="600"
outPriv="admin" outSessionId="17" outVersion="3.0(0.149)"> </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="stringMin0Max47" use="required"/>
            <xs:attribute name="cookie" type="stringMin0Max47"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

Response Syntax

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

Examples

Request

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

Response

```
<aaaLogout cookie="<real_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 IMC 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" use="required">
                <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" use="required">
                <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="stringMin0Max47" use="required"/>
            <xs:attribute name="cookie" type="stringMin0Max47"/>
            <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="(read-only|admin|user){0,1}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="fullssl"/>
                        <xs:enumeration value="noencssl"/>
                        <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="fullssl"/>
                        <xs:enumeration value="noencssl"/>
                        <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

Examples

Request

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

<aaaRefresh cookie="<real_cookie>" inCookie="<real_cookie>" inName="admin"
```

```
inPassword="password">
</aaaRefresh>
```

Response

```
<aaaRefresh
    cookie="<real_cookie>"
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin">
</aaaRefresh>
```

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="1"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject" use="required"/>
        </xs:complexType>
```

Response Syntax

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

Examples

Request

```
<configConfMo
 cookie="<real_cookie>"
 dn='sys/rack-unit-1/locator-led'>
 <inConfig>
 <equipmentLocatorLed
 adminState='on' dn='sys/rack-unit-1/locator-led'>
 </equipmentLocatorLed>
```

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

Response

```
<configConfMo dn="sys/rack-unit-1/locator-led"
cookie="1461754266/2f609b81-3176-1176-8007-4cc92474a254" response="yes">
<outConfig>
<equipmentLocatorLed dn="sys/rack-unit-1/locator-led" adminState="inactive" color="unknown"
 id="1" name="" operState="off" status="modified"/></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>
```

Note

- Rolling back to the previously existing configuration when the multiple configuration fails, is not supported.
- **operStatus** in the output indicates whether or not the configuration has succeeded. This could have the following statuses:
- Success—indicates that the configuration of multiple MOs was successful.
- Partial success—indicates that a few configurations failed, displays the failed configurations and the reason for failure.
- Failure—indicates that all the configurations failed.

Examples

Request

```
<configConfMos
    cookie="<real_cookie>">
    <inConfigs>
    <pair key="sys/user-ext/user-13">
      <aaaUser priv="admin" id="13" name="TESTUSER13" pwd="testTEST15"
       accountStatus="active" dn="sys/user-ext/user-13"/>
    </pair>
    <pair key="sys/user-ext/user-14">
      <aaaUser priv="admin" id="14" name="TESTUSER14" pwd="testTEST15"
       accountStatus="active" dn="sys/user-ext/user-14"/>
    </pair>
    <pair key="sys/svc-ext/snmp-svc/snmpv3-user-5">
      <commSnmpUser id="5" name="testuser_5" securityLevel="authpriv"
       auth="MD5" authPwd="testtest" privacy="DES" privacyPwd="testTest5"/>
    </pair>
    <pair key="sys/svc-ext/snmp-svc/snmpv3-user-1">
      <commSnmpUser id="1" name="testuser_1" securityLevel="authpriv"
      auth="MD5" authPwd="testtest" privacy="DES" privacyPwd="testTest5"/>
    </pair>
    <pair key="sys/rack-unit-1/boot-precision">
      <lsbootDevPrecision dn="sys/rack-unit-1/boot-precision"
        rebootOnUpdate="no" status="modified">
        <lsbootHdd name="testhdd_2" order="2"/>
        <lsbootHdd name="testhdd_1" order="1"/>
      </lsbootDevPrecision>
    </pair>
  </inConfigs>
</configConfMos>
```

Response

```
<configConfMos
    cookie="<real_cookie>"
   <outConfigs>
     <pair key="sys/user-ext/user-13">
      <aaaUser id="13" accountStatus="active" name="TESTUSER13"
       priv="admin" pwd="" adminAction="no-op"
       dn="sys/user-ext/user-13" status="modified"/>
    </pair>
    <pair key="sys/user-ext/user-14">
      <aaaUser id="14" accountStatus="inactive" name="" priv="" pwd=""
      adminAction="no-op" dn="sys/user-ext/user-14" status="modified"/>
    </pair>
    <pair key="sys/svc-ext/snmp-svc/snmpv3-user-5">
      <commSnmpUser id="5" name="testuser_5" securityLevel="authpriv"
```

```
      auth="MD5" authPwd="" privacy="DES" privacyPwd=""
      adminAction="no-op" dn="sys/svc-ext/snmp-svc/snmpv3-user-5"
      status="modified"/>
    </pair>
    <pair key="sys/svc-ext/snmp-svc/snmpv3-user-1">
      <commSnmpUser id="1" name="" securityLevel="" auth="" authPwd=""
      privacy="" privacyPwd="" adminAction="no-op"
      dn="sys/svc-ext/snmp-svc/snmpv3-user-1" status="modified"/>
    </pair>
    <pair key="sys/rack-unit-1/boot-precision">
      <lsbootDevPrecision dn="sys/rack-unit-1/boot-precision"
       name="boot-precision" purpose="operational" rebootOnUpdate="no"
       reapply="no" configuredBootMode="Legacy"
       lastConfiguredBootOrderSource="CIMC" status="modified">
        <lsbootVMedia name="CDDVD" type="VMEDIA" subtype="kvm-mapped-dvd"
        access="read-only-local" order="3" state="Disabled" rn="vm-CDDVD"
        status="modified"/>
        <lsbootHdd name="testhdd_1" type="LOCALHDD" order="1"
         state="Disabled" rn="hdd-testhdd_1" status="modified"/>
        <lsbootHdd name="testhdd_2" type="LOCALHDD" order="2"
         state="Disabled" rn="hdd-testhdd_2" status="modified"/>
        <lsbootHdd name="HDD1" type="LOCALHDD" slot="HBA" order="4"
         state="Disabled" rn="hdd-HDD1" status="modified"/>
      </lsbootDevPrecision>
    </pair>
    <operStatus>all success</operStatus>
  </outConfigs>
</configConfMos>
```

configResolveChildren

The configResolveChildren method retrieves children of managed objects under a specific DN in the managed information tree.

Request Syntax

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveChildren" mixed="true">
            <xs:attribute name="inDn" type="referenceObject" use="required"/>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

Response Syntax

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

```
        </xs:complexType>
```

Examples

Request

```
<configResolveChildren cookie="<real_cookie>" inHierarchical="false" inDn="sys"/>
```

Response

```
<configResolveChildren cookie="1461754266/2f609b81-3176-1176-8007-4cc92474a254"
response="yes"> <outConfigs> <computeRackUnit dn="sys/rack-unit-1" adminPower="policy"
availableMemory="16384" model="UCSC-server-name" memorySpeed="1333" name="UCS server_name"
 numOfAdaptors="1" numOfCores="8" numOfCoresEnabled="8" numOfCpus="2" numOfEthHostIfs="2"
numOfFcHostIfs="2" numOfThreads="8" operPower="on"
originalUuid="3FDC58B1-26CF-4CFA-BFA9-B028047280B1" presence="equipped" serverId="1"
serial="FCH1917V0P1" totalMemory="16384" usrLbl="" uuid="3FDC58B1-26CF-4CFA-BFA9-B028047280B1"
 vendor="Cisco Systems Inc" cimcResetReason="graceful-reboot " assetTag="in demo"
></computeRackUnit><aaaUserEp dn="sys/user-ext" ></aaaUserEp><aaaLdap dn="sys/ldap-ext"
adminState="disabled" basedn="" domain="" filter="sAMAccountName" attribute="CiscoAvPair"
timeout="60" encryption="enabled" locateDirectoryUsingDNS="no"
dnsDomainSource="extracted-domain" dnsSearchDomain="" dnsSearchForest="" ldapServer1=""
ldapServerPort1="389" ldapServer2="" ldapServerPort2="389" ldapServer3=""
ldapServerPort3="389" ldapServer4="" ldapServerPort4="3268" ldapServer5=""
ldapServerPort5="3268" ldapServer6="" ldapServerPort6="3268" bindMethod="login-credentials"
 bindDn="" password="" groupAuth="disabled" groupAttribute="memberOf" groupNestedSearch="128"
 ></aaaLdap><commSvcEp dn="sys/svc-ext" ></commSvcEp><certificateManagement dn="sys/cert-mgmt"
 description="Certificate Management" ></certificateManagement><mgmtImporter
dn="sys/import-config" adminState="disabled" fsmStageDescr="" fsmRmtInvErrCode=""
fsmRmtInvErrDescr="" fsmDescr="import-config" proto="none" hostname="" remoteFile="" user=""
 pwd="" passphrase="" ></mgmtImporter><mgmtBackup dn="sys/export-config" adminState="disabled"
 fsmStageDescr="" fsmRmtInvErrCode="" fsmRmtInvErrDescr="" fsmDescr="export-config"
proto="none" hostname="" remoteFile="" user="" pwd="" passphrase=""
></mgmtBackup><mgmtInventory dn="sys/inventory" adminState="triggered" proto="none"
hostname="" remoteFile="" user="" pwd="" fsmStatus="COMPLETED" progress="100%"
></mgmtInventory><huuController dn="sys/huu" description="Host Upgrade Utility (HUU)"
></huuController><iodController dn="sys/iod" description="Non-Interactive Offline Diagnostics
 (IOD)" ></iodController></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:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId" use="required"/>
        </xs:complexType>
```

Response Syntax

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

Examples

Request

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

Response

```
<configResolveClass cookie="<real_cookie>" response="yes" classId="topSystem">
 <outConfigs> <topSystem dn="sys" address="10.10.10.10" currentTime="Wed Apr 27 10:51:08
2016 "
  localTime="Wed Apr 27 13:51:08 2016 EAT +0300" timeZone="Africa/Addis Ababa"
mode="stand-alone"
  name="Cxxx-FCH1917V0P1" >
 </topSystem>
</outConfigs>
 </configResolveClass>
```

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 YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject" use="required"/>
        </xs:complexType>
```

Response Syntax

```
<xs:element name="configResolveDn" type="configResolveDn"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveDn" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
```

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

Examples

Request

```
<configResolveDn cookie="1461754266/2f609b81-3176-1176-8007-4cc92474a254"
inHierarchical="false" dn="sys/rack-unit-1"/>
```

Response

```
<configResolveDn cookie="<real_cookie>" response="yes" dn="sys/rack-unit-1">
<outConfig> <computeRackUnit dn="sys/rack-unit-1" adminPower="policy" availableMemory="16384"

 model="UCSC-server-name" memorySpeed="1333" name="UCS server_name" numOfAdaptors="1"
numOfCores="8"
 numOfCoresEnabled="8" numOfCpus="2" numOfEthHostIfs="2" numOfFcHostIfs="2" numOfThreads="8"

 operPower="on" originalUuid="3FDC58B1-26CF-4CFA-BFA9-B028047280B1" presence="equipped"
 serverId="1" serial="FCH1917V0P1" totalMemory="16384" usrLbl=""
 uuid="3FDC58B1-26CF-4CFA-BFA9-B028047280B1" vendor="Cisco Systems Inc"
 cimcResetReason="graceful-reboot " assetTag="in demo" >
 </computeRackUnit>
</outConfig>
</configResolveDn>
```

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 YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject" use="required"/>
        </xs:complexType>
```

Response Syntax

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

```
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

Examples

Request

```
<configResolveParent cookie="<real_cookie>" inHierarchical="false" dn="sys/rack-unit-1"/>
```

Response

```
<configResolveParent cookie="<real_cookie>" response="yes" dn="sys/rack-unit-1">
 <outConfig>
<topSystem dn="sys" address="10.197.125.42" currentTime="Wed Apr 27 10:53:26 2016 "
  localTime="Wed Apr 27 13:53:26 2016 EAT +0300" timeZone="Africa/Addis Ababa"
mode="stand-alone"
  name="server-FCH1917V0P1" >
 </topSystem>
</outConfig>
</configResolveParent>
```

eventSubscribe

The eventSubscribe method allows a client to subscribe to asynchronous System Event Log (SEL) events generated by Cisco IMC.

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

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

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

Request Syntax

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

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

Response Syntax

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

Examples

Request

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

Response

```
NO RESPONSE OR ACKNOWLEDGMENT.
```

eventUnsubscribe

The eventUnsubscribe method allows a client to unsubscribe from asynchronous System Event Log (SEL) events generated by Cisco IMC, reversing event subscriptions that resulted from eventUnsubscribe.

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

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

Request Syntax

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

Response Syntax

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

Examples

Request

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

Response

```
NO RESPONSE OR ACKNOWLEDGMENT.
```

eventUnsubscribe