审阅稿 - 思科机密

思科身份服务引擎 API 参考指南，版本 1.4

2015 年 12 月 15 日

思科系统公司
www.cisco.com

思科在全球设有 200 多个办事处。
有关地址、电话号码和传真号码信息，
可查阅思科网站：
www.cisco.com/go/offices。

文本部件号：OL-26134-01

思科身份服务引擎 API 参考指南，版本 1.4

目 录

前言

修订日期：12/15/15, OL-26134-01

本前言介绍有关*思科身份服务引擎 API 参考指南，版本 1.4* 的目的、预期受众和组织。前言还介绍在以下部分提供说明和提供其他信息类型的约定：

思科身份服务引擎概述

思科身份服务引擎 (ISE) 是下一代身份和访问控制策略平台，可帮助企业执行策略规定、加强基础设施安全以及简化服务操作。凭借 Cisco ISE 的独特架构，企业可以通过把身份绑定到各种网络元素（包括访问交换机、无线局域网控制器 (WLC)、虚拟专用网络 (VPN) 网关和数据中心交换机），从网络、用户和设备收集实时背景信息，从而做出前瞻性的管理决策。

Cisco ISE 是思科安全组访问解决方案的关键组件。Cisco ISE 是基于统一策略的访问控制解决方案，具有以下特点：

- 将身份验证、授权、记帐 (AAA)、状态、分析器和访客管理服务合并到一个设备
- 通过检查访问网络的所有终端（包括 802.1X 环境）的设备状态执行终端策略规定
- 提供发现、分析、基于策略的布局和监控网络上的终端设备支持
- 在集中式和分布式部署中启用一致的策略，以实现根据实际需要交付服务
- 通过使用安全组标记 (SGT) 和安全组 (SG) 访问控制列表 (ACL) 使用高级实施功能，包括安全组访问 (SGA)
- 支持将多种部署方案从小型办公室扩展到大型企业环境的可扩展性

Cisco ISE 体系结构支持独立部署和分布式部署，使您可以通过集中门户配置和管理自己的网络。有关 Cisco ISE 的功能的详细信息，请参阅*思科身份服务引擎管理员指南，版本 1.3*。

目的

此应用编程接口 (API) 参考指南仅提供支持的 API 所提供功能的简短高级概述。此 API 参考指南的目的是为开发人员、系统或网络管理员或系统集成者提供在 Cisco ISE 部署中使用重要 API 的基本指南。

REST API 调用使用查询确定以下类型的数据：

- 活动会话的数量
- 活动会话的类型
- 活动会话的身份验证状态
- 使用的 MAC 地址
- 使用的 NAS IP 地址
- 节点版本和类型
- 节点会话失败的原因

外部 RESTful 服务 API 和相关 API 调用可用于对 Cisco ISE 资源执行 CRUD（创建、读取、更新、删除）操作。外部 RESTful 服务是基于 HTTP 协议和 REST 方法。

注意 有关 Cisco ISE 网络、其节点和角色、操作概念或用法、以及 Cisco ISE 用户界面使用方法的详细信息，请参阅*思科身份服务引擎管理员指南，版本 1.3*。

受众

此 API 参考指南面向管理网络环境中的 Cisco ISE 设备的系统管理员、希望利用 API 的系统集成者，或负责管理 Cisco ISE 部署或排除故障的第三方合作伙伴。使用此 API 参考指南的必备条件是您应该具备有关故障排除、诊断实践以及如何执行和解析 API 调用的基本知识。

文档结构

本指南的结构如下：

- 第 1 部分 - Cisco ISE 监控 REST API
  - 第 1 章，“监控 REST API 简介”
  - 第 2 章，“会话管理查询 API”
  - 第 3 章，“用于故障排除的查询 API”
  - 第 4 章，“授权更改 REST API”
- 第 2 部分 - Cisco ISE 外部 RESTful 服务 API
  - 第 5 章，“ERS API 简介”
  - 第 7 章，“外部 RESTful 服务 API 操作”
- 第 3 部分 - Cisco pxGrid API
  - 附录 A，“Cisco ISE 故障原因报告”

文档约定

本节概述本文档中使用的约定。

小心 表示*读者应当小心*。您的某些操作可能会导致设备损坏或数据丢失。

注意 表示*读者需要注意的地方*。注释中包含有用的建议或包含对本手册中所没有的材料的引用。

此 API 参考指南使用以下约定表示指令和信息。

项目	约定
命令、关键字、特殊术语和应在操作过程中选择的选项	**粗体**
用于提供值的变量和新的或重要的术语	*斜体*
显示的会话和系统信息、路径和文件名	屏幕字体
您输入的信息	**屏幕粗体**
您输入的变量	*屏幕斜体*
菜单项和按钮名称	**粗体**
表示要选择的菜单项，按照选择的顺序显示。	**Option > Network Preferences**

文档更新

表 1 列出本文档自创建之日起所进行的更新，最近的更新显示在列表前面。

表 1　　思科身份服务引擎 API 参考指南，版本 1.4 的更新

日期	描述
2014-8-25	思科身份服务引擎 (ISE) 1.3 版

产品文档

注意 思科在原始发布之后有时会更新打印文档和电子文档。因此，您还应该从 http://www.cisco.com 查看所有更新信息。

表 2 列出 www.cisco.com 上提供的关于 Cisco ISE 1.3 版本的相关产品文档。要在 www.cisco.com 上查找所有产品的最终用户文档，请访问：

http://www.cisco.com/go/techdocs

相关文档

本节提供版本特定文档以及平台特定文档的有关信息。

版本特定文档

表 2 列出 Cisco ISE 版本的产品文档。Cisco ISE 的一般产品信息位于 http://www.cisco.com/go/ise。最终用户文档位于 Cisco.com 上的 http://www.cisco.com/en/US/products/ps11640/tsd_products_support_series_home.html。

表 2　　思科身份服务引擎的产品文档

文档标题	位置
思科身份服务引擎版本说明，版本 1.3	http://www.cisco.com/en/US/products/ps11640/prod_release_notes_list.html
思科身份服务引擎网络组件兼容性，版本 1.3	http://www.cisco.com/en/US/products/ps11640/products_device_support_tables_list.html
思科身份服务引擎管理员指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/products_user_guide_list.html
思科身份服务引擎硬件安装指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/prod_installation_guides_list.html
适用于 Cisco Secure ACS 5.5 的思科身份服务引擎迁移指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/prod_installation_guides_list.html
思科身份服务引擎发起人门户用户指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/products_user_guide_list.html
思科身份服务引擎 CLI 参考指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/prod_command_reference_list.html
思科身份服务引擎 API 参考指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/prod_command_reference_list.html
思科身份服务引擎故障排除指南，版本 1.3	http://www.cisco.com/en/US/products/ps11640/prod_troubleshooting_guides_list.html
思科身份服务引擎、Cisco 1121 安全访问控制系统、Cisco NAC Appliance、Cisco NAC Guest Server 与 Cisco NAC Profiler 的合规性和安全信息	http://www.cisco.com/en/US/products/ps11640/prod_installation_guides_list.html
思科身份服务引擎设备内文档和中国 RoHS 指针卡	http://www.cisco.com/en/US/products/ps11640/products_documentation_roadmaps_list.html

平台特定文档

策略管理业务部门文档的链接位于 http://www.cisco.com 上的以下位置：

- Cisco ISE
  http://www.cisco.com/en/US/products/ps11640/prod_installation_guides_list.html
- Cisco Secure ACS
  http://www.cisco.com/en/US/products/ps9911/tsd_products_support_series_home.html
- Cisco NAC 设备
  http://www.cisco.com/en/US/products/ps6128/tsd_products_support_series_home.html
- Cisco NAC 分析器
  http://www.cisco.com/en/US/products/ps8464/tsd_products_support_series_home.html
- Cisco NAC 访客服务器
  http://www.cisco.com/en/US/products/ps10160/tsd_products_support_series_home.html

获取文档和提交服务请求

关于如何获取文档、提交服务请求和收集其他信息的信息，请参阅每月的*思科产品文档更新*，其中还含有所有最新及修订的思科技术文档，此文档位于：

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

通过 Really Simple Syndication (RSS) 源的方式订阅*思科产品文档更新*，相关内容将通过阅读器应用直接发送到您的桌面。RSS 源是一项免费服务，思科目前支持 RSS 2.0 版本。

第 1 部分

Cisco ISE 监控 REST API

监控 REST API 简介

通过监控 REST API，您可以使用网络中的监控节点来收集特定于会话和节点的信息。会话的定义是访问所需节点与完成收集信息所需的操作之间的持续时间。

Cisco ISE 版本 1.4 支持以下监控 REST API 类别：

- 会话管理
- 故障排除
- 授权更改 (CoA)

注意 请仅使用支持的类别来收集监控角色所监控终端的有关信息。监控角色是 Cisco ISE 版本 1.4 部署中，节点类型可以执行的三个受支持角色之一。本指南其余内容将使用“监控节点”描述 Cisco ISE 节点的监控角色。

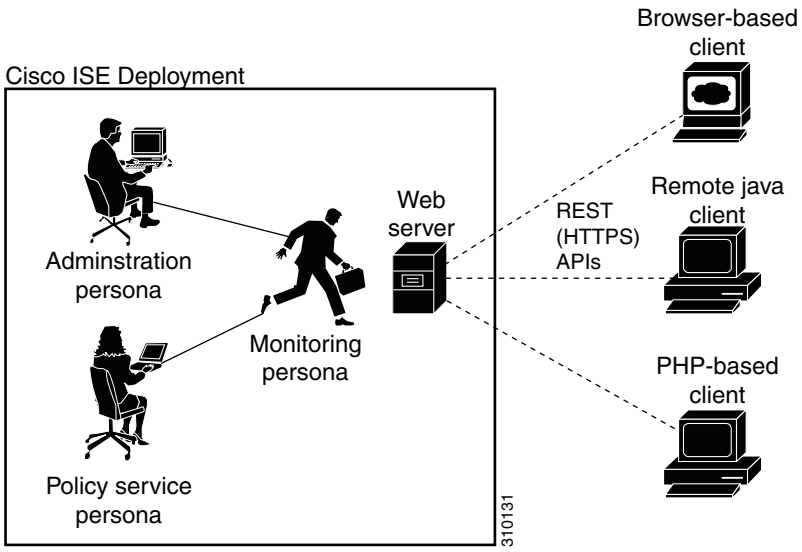
任何使用这些类别收集 Cisco ISE 设备的策略服务角色相关信息的尝试都会产生错误。有关 Cisco ISE 节点和角色的详细信息，请参阅*思科身份服务引擎管理员指南，版本 1.4*。

通过监控 REST API 调用，您可以查找、监控和汇总存储在网络中单个终端内基于会话的重要实时信息。您可以通过监控节点访问此信息。

所收集的基于会话的实时信息可帮助您了解 Cisco ISE 运行情况，有助于诊断各种情况或问题。这些信息还可用于对可能影响监控操作的出错情况、活动或行为进行故障排除。如图 1-1 所示，监控 REST API 调用用于访问监控节点以及检索存储在 Cisco ISE 部署终端内基于会话的重要信息。

注意 Cisco ISE 版本 1.2 的监控 REST API 已弃用，所有服务调用的 URL 路径也已更改。请使用 Cisco ISE 版本 1.4 的监控 REST API。

图 1-1　　分布式部署中的监控 REST API 调用



验证监控节点

准备工作

您需要确认要监控的节点是有效的，才能在该节点上成功调用 API 调用。

注意　为使用公共监控 REST API，您必须先使用有效凭证向 Cisco ISE 进行身份验证。

步骤 1　在 Cisco ISE 登录窗口中输入有效的登录凭证（用户名和密码），然后点击 **Login**。

系统将显示 Cisco ISE 控制面板和用户界面。

步骤 2　选择 **Authorization > System > Deployment**。

系统将显示 Deployment Nodes 页面，其中列出所部署的所有已配置的节点。

步骤 3　在 Deployment Nodes 页面的 Roles 列，验证您要监控的目标节点的角色是否作为监控节点列出。

支持的 API 调用

下表介绍不同的 API 调用类型并提供 API 调用格式的示例：

- 表 1-1（第 1-3 页） - 定义用于会话管理的 API 调用。
- 表 1-2（第 1-6 页） - 定义用于故障排除的 API 调用。
- 表 1-3（第 1-7 页） - 定义 CoA API 调用。

如果要使用一般编程接口对 Cisco ISE 支持的监控 REST API 进行身份验证，您需要先创建基于 REST 的客户端，用于桥接 Cisco ISE 与所使用的特定工具。然后，使用此 REST 客户端对 Cisco ISE 监控 REST API 进行身份验证，向监控节点封送和提交 API 请求，然后取消封送 API 响应并将其传递到指定的工具。

表 1-1　Cisco ISE 会话管理 API 调用

API 调用类别	说明和示例
会话计数器	
ActiveCount	列出活动会话的数量。 https://<ISEhost>/admin/API/mnt/Session/ActiveCount
PostureCount	列出状态评估终端的数量。 https://<ISEhost>/admin/API/mnt/Session/PostureCount **备注**　状态评估是一项用于帮助检查连接到 Cisco ISE 网络的所有终端的状态（或状况）的服务。Cisco ISE 利用 NAC 代理检查设备的状态合规性。
ProfilerCount	列出活动分析器服务会话的数量。 https://<ISEhost>/admin/API/mnt/Session/ProfilerCount **备注**　分析器是一项用于帮助识别、查找和确定 Cisco ISE 网络上所有已连接终端的功能的服务。

表 1-1　Cisco ISE 会话管理 API 调用 （续）

API 调用类别	说明和示例
会话列表 **备注**　会话列表包括与会话关联的 MAC 地址、网络访问设备 (NAD) IP 地址、用户名和会话 ID 信息。	
ActiveList	列出所有活动会话。 https://<ISEhost>/admin/API/mnt/Session/ActiveList **备注**　在此 Cisco ISE 版本中，可以显示且经过身份验证的最大活动终端会话数为 250000。
AuthList	列出当前所有经过身份验证的活动会话。 https://<ISEhost>/admin/API/mnt/Session/AuthList/<parameteroptions> 您可以指定以下参数选项（分别会返回不同的值）： • null/null - 列出所有经过身份验证的活动会话。 • null/endtime - 列出所有在指定结束时间之后经过身份验证的活动会话。 • starttime/null - 列出所有在指定开始时间之前经过身份验证的活动会话。 • starttime/endtime - 列出所有在指定开始时间与结束时间之间经过身份验证的活动会话。 使用以下格式为开始时间和结束时间输入日期和时间： YYYY-MM-DD hh:mm:ss.s 其中： • YYYY - 四位数年份 • MM - 两位数月份（01= 一月，依次类推） • DD - 两位数日期（01 至 31） • HH - 两位数小时（00 至 23）（不允许使用 a.m. 和 p.m.） • mm - 两位数分钟（00 至 59） • ss - 两位数秒（00 至 59） • s - 表示秒的小数位的一个或多个数字 **备注**　每个 Cisco ISE 节点都配置了时区。建议使用 UTC 时区。 有关显示全部四个参数选项的示例，请参阅 使用 null/null 选项的 AuthList API 调用返回的数据示例，第 2-8 页。

表 1-1　Cisco ISE 会话管理 API 调用 （续）

API 调用类别	说明和示例
会话属性 **备注**　这是基于时间戳的搜索，用于搜索包含指定搜索属性的最新会话。	
MAC 地址	在数据库中搜索包含指定 MAC 地址的最新会话。 https://<ISEhost>/admin/API/mnt/Session/MACAddress/<macaddress> **备注**　XX:XX:XX:XX:XX:XX 是 MAC 地址格式，不区分大小写（例如，0a:0B:0c:0D:0e:0F）。 **备注**　MAC 地址作为用于查找要监控的正确会话的唯一键。使用 ActiveList API 调用列出所有活动会话及其 MAC 地址，您可以据此执行 MAC 地址搜索。
UserName	在数据库中搜索包含指定用户名的最新会话。 https://<ISEhost>/admin/API/mnt/Session/UserName/<username> **备注**　用户名必须符合用于网络用户名的 Cisco ISE 密码策略。监控 REST API 的唯一无效字符为反斜线 (\) 字符。有关详细信息，请参阅*思科身份服务引擎用户指南，版本 1.1* 中的“用户密码策略”。
IPAddress（IP 地址）	在数据库中搜索包含指定 NAS IP 地址的最新会话。 https://<ISEhost>/admin/API/mnt/Session/IPAddress/<nasipaddress> **备注**　xxx.xxx.xxx.xxx 是 NAS IP 地址格式（例如，10.10.10.10）。
Audit Session ID	在数据库中搜索包含指定审核会话 ID 的最新会话。 https://<ISEhost>/admin/API/mnt/Session/Active/SessionID/<audit-session-id>/0 **备注**　使用 ActiveList API 调用列出所有活动会话及其审核会话 ID，您可以据此执行审核会话 ID 搜索。或者，您可以从管理员门户的 Live Sessions 页面获取审核会话 ID。

有关用于会话管理的 Cisco ISE API 调用的特定详细信息，请参阅第 2 章，“会话管理查询 API”。

表 1-2　Cisco ISE 故障排除 API 调用 - 故障排除

API 调用	说明和示例
版本	列出节点版本和类型。 https://<ISEhost>/admin/API/mnt/Version 节点类型可以是以下任意值 (0-3)：0 - STAND_ALONE_MNT_NODE 1 - ACTIVE_MNT_NODE 2 - STAND_BY_MNT_NODE 3 - NOT_AN_MNT_NODE **备注**　STAND_ALONE_MNT_NODE 表示此节点是未在任何分布式部署中运行的监控节点。 ACTIVE_MNT_NODE 表示此节点是分布式部署内主 - 次关系中的主节点。 STAND_BY_MNT_NODE 表示此节点是分布式部署内主 - 次对中的辅助节点。 NOT_AN_MNT_NODE 表示此节点不是监控节点。有关支持的 ISE 节点和角色的详细信息，请参阅*思科身份服务引擎 API 参考指南，版本 1.1*。
FailureReasons	列出故障原因。 https://<ISEhost>/admin/API/mnt/FailureReasons 每个故障原因会显示一个错误代码 (failureReason id)、简短说明 (code)、故障原因 (cause) 和可能的响应 (resolution)，如下例所示： <failureReason id="100009"> <code> 100009 WEBAUTH_FAIL <cause> This may or may not be indicating a violation. <resolution> Please review and resolve this issue according to your organization's policy. **备注**　FailureReasons API 调用仅调用一次，用于从监控节点收集信息。您应将返回的所有故障原因的内容存储在自己的文件系统或数据库中。这些 API 调用返回的内容供参考使用。如果您在身份验证过程中遇到任何问题，应将身份验证响应中提供的故障原因代码与自己文件系统或数据库中存储的故障原因列表进行对比。 有关 Cisco ISE 故障原因的完整列表，请参阅附录 A，“Cisco ISE 故障原因报告”。

表 1-2　Cisco ISE 故障排除 API 调用 - 故障排除 （续）

API 调用	说明和示例
AuthStatus	列出所有会话的身份验证状态。 https://<ISEhost>/admin/API/mnt/AuthStatus/MACAddress/<macaddress>/<numberofseconds>/<numberofrecordspermacaddress>/All **备注**　用户可以配置秒参数 <numberofseconds>，其范围是 0 至 432000 秒（5 天）。
获取会话记帐状态	
AcctStatus	列出指定时间段内所有会话的记帐状态。 https://<ISEhost>/admin/API/mnt/AcctStatusTT/MACAddress/<macaddress>/<numberof seconds> **备注**　用户可以配置秒参数 <numberofseconds>，其范围是 0 至 432000 秒（5 天）。

有关用于故障排除的 Cisco ISE API 调用的特定详细信息，请参阅第 2 章，“会话管理查询 API”。

表 1-3　Cisco ISE 授权更改 API 调用

API 调用	说明和示例
Reauth	发送会话重新身份验证命令和类型。 https://<ISEhost>/admin/API/mnt/CoA/Reauth/<serverhostname>/<macaddress>/<reauthtype>/<nasipaddress>/<destinationipaddress> 其中，<*ISEhost*> 表示 ISE 主机的 IP 地址，<*serverhostname*> 表示 ISE 服务器的名称，<*nasipaddress*> 表示 NAS 的识别 IP 地址，<*destinationipaddress*> 表示目标的 IP 地址。 Reauth 类型可以是以下任意值 (0-2)： 0 - REAUTH_TYPE_DEFAULT 1 - REAUTH_TYPE_LAST 2 - REAUTH_TYPE_RERUN **备注**　如果您不知道 NAS IP 地址，您可以输入所需值（最大不超过该点），API 将在其搜索查询中使用这些值。但是，您必须知道 MAC 地址才能执行此 API 调用，您可以将从 NAS IP 地址开始的其他参数保留为 null。如果提供 NAS IP 地址，还必须提供目标 IP 地址。 此 API 调用只能在监控 ISE 节点上执行，此节点会提交请求以远程执行 CoA。执行这些 CoA API 调用不涉及也不需要使用管理 ISE 节点。

表 1-3　Cisco ISE 授权更改 API 调用 （续）

API 调用	说明和示例
会话断开连接	
Disconnect	发送会话断开连接命令和端口选项类型。 https://<ISEhost>/admin/API/mnt/CoA/Disconnect/<serverhostname>/<macaddress>/<disconnecttype>/<nasipaddress>/<destinationipaddress> 端口选项类型可以是以下任意值 (0-2)： 0 - DYNAMIC_AUTHZ_PORT_DEFAULT 1 - DYNAMIC_AUTHZ_PORT_BOUNCE 2 - DYNAMIC_AUTHZ_PORT_SHUTDOWN **备注**　如果您不知道 NAS IP 地址，您可以输入所需值（最大不超过该点），API 将在其搜索查询中使用这些值。但是，您必须知道 MAC 地址才能执行此 API 调用，您可以将其他参数保留为 null。

有关 Cisco ISE 授权更改 API 调用的详细信息，请参阅第 4 章，“授权更改 REST API”。

HTTP PUT API 调用

与表 1-2 中的 AuthStatus API 类似，API 调用的 HTTP PUT 版本可让客户端检索帐户状态。监控 REST API 支持 HTTP PUT 和 HTTP GET 调用，本指南中提供介绍 HTTP GET 调用的示例。HTTP PUT 可解决要求参数输入的调用的需求。以下架构文件示例是一个帐户状态请求：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="acctRequest" type="mnTRESTAcctRequest"/>

  <xs:complexType name="mnTRESTAcctRequest">
    <xs:complexContent>
      <xs:extension base="mnTRESTRequest">
        <xs:sequence>
          <xs:element name="duration" type="xs:string" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="mnTRESTRequest" abstract="true">
    <xs:sequence>
      <xs:element name="valueList">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="searchCriteria" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

会话管理查询 API

本章介绍会话管理 API 调用，这些调用提供从 Cisco ISE 部署中的思科监控 ISE 节点内检索会话相关重要信息的途径。

会话计数器 API 调用

通过下列会话计数器 API 调用，您可以快速地在 Cisco ISE 部署中的目标思科监控 ISE 节点上收集会话相关信息的当前计数。

- 活动会话 (ActiveCount) - 活动会话是指已经在网络上经过身份验证的会话。
- 状态评估会话 (PostureCount) - 获得状态评估结论（合规 / 不合规）时会声明“状态评估”状态。状态评估为可选项，例如，IP 电话 / 打印机不会进入“状态评估”状态。“状态评估”是一个短暂的临时状态，因为在状态评估后，当完成记帐开始设置时，状态会变为“已开始”。
- 已分析会话 (ProfilerCount)。

如果终端滞留在某个阶段，这些不同状态可帮助进行故障排除。

活动会话计数器

您可以使用 ActiveCount API 调用来检索所有当前活动会话的计数。本节提供架构文件输出示例、通过发起 ActiveCount API 调用来计算所有活动会话计数的程序，以及发起此 API 调用后返回的活动会话数据的示例。

ActiveCount API 输出架构

此示例架构文件是用于检索 ISE 节点目标监控角色上活动会话计数的 ActiveCount API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionCount" type="activeCount"/>
  <xs:complexType name="activeCount">
    <xs:sequence>
      <xs:element name="count" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 ActiveCount API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>)，在目标节点的 URL 地址字段输入 ActiveCount API 调用：

```
https://acme123/admin/API/mnt/Session/ActiveCount
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。API 调用约定中使用“mnt”表示目标思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

ActiveCount API 调用返回的数据示例

以下示例展示的是您在目标思科监控 ISE 节点上发起 ActiveCount API 调用时返回的数据（活动会话数）：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.
-
<sessionCount>
<count>5</count>
</sessionCount>
```

状态评估会话计数器

您可以使用 PostureCount API 调用来检索所有当前活动状态评估会话的计数。

PostureCount API 输出架构

此示例架构文件是用于检索目标思科监控 ISE 节点上当前活动的分析器会话计数的 ProfilerCount API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionCount" type="postureCount"/>

  <xs:complexType name="postureCount">
    <xs:sequence>
```

```
      <xs:element name="count" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 PostureCount API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/Session/<specific-api-call>)，在目标节点的 URL 地址字段输入 PostureCount API 调用：

```
https://acme123/admin/API/mnt/Session/PostureCount
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。API 调用约定中使用“mnt”表示目标思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

PostureCount API 调用返回的数据示例

以下示例展示的是您在目标思科监控 ISE 节点上发起 PostureCount API 调用时返回的数据（当前活动的状态评估会话数）：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<sessionCount>
<count>3</count>
</sessionCount>
```

分析器会话计数器

您可以使用 ProfilerCount API 调用来检索所有当前活动分析器会话的计数。

ProfilerCount API 输出架构

此示例架构文件是用于检索目标思科监控 ISE 节点上当前活动的分析器会话计数的 ProfilerCount API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionCount" type="profilerCount"/>

  <xs:complexType name="profilerCount">
    <xs:sequence>
      <xs:element name="count" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 ProfilerCount API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址 >/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/Session/<specific-api-call>)，在目标节点的 URL 地址字段输入 ProfilerCount API 调用：

```
https://acme123/admin/API/mnt/Session/ProfilerCount
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

ProfilerCount API 调用返回的数据示例

以下示例展示的是您在目标思科监控 ISE 节点上发起 ProfilerCount API 调用时返回的数据（活动分析器会话数）：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<sessionCount>
<count>1</count>
</sessionCount>
```

简单会话列表 API 调用

通过以下简单会话列表 API 调用，您可以快速地在 Cisco ISE 部署中的目标思科监控 ISE 节点上收集会话相关信息，如与当前活动会话关联的 MAC 地址、网络访问设备 (NAD) IP 地址、用户名和会话 ID。

- 活动会话列表 (ActiveList)
- 经过身份验证的会话列表 (AuthList)

活动会话列表

您可以使用 ActiveList API 调用来列出所有当前活动的会话。

注意　可以显示且经过身份验证的最大活动终端会话数为 100000。

ActiveList API 输出架构

此示例架构文件是用于检索目标思科监控 ISE 节点上当前活动会话列表（及会话相关信息）的 ActiveList API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="activeSessionList" type="simpleActiveSessionList"/>

<xs:complexType name="simpleActiveSessionList">
    <xs:sequence>
      <xs:element name="activeSession" type="simpleActiveSession" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="noOfActiveSession" type="xs:int" use="required"/>
  </xs:complexType>

  <xs:complexType name="simpleActiveSession">
    <xs:sequence>
      <xs:element name="user_name" type="xs:string" minOccurs="0"/>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="acct_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="server" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 ActiveList API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址 >/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/Session/<specific-api-call>)，在目标节点的 URL 地址字段输入 ActiveList API 调用：

```
https://acme123/admin/API/mnt/Session/ActiveList
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

ActiveList API 调用返回的数据示例

以下示例展示的是您在目标思科监控 ISE 节点上发起 ActiveList API 调用时从活动会话列表返回的会话相关数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<activeSessionList noOfActiveSession="5">
-
<activeSession>
<calling_station_id>00:0C:29:FA:EF:0A</calling_station_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<calling_station_id>70:5A:B6:68:F7:CC</calling_station_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>tom_wolfe</user_name>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000032</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>graham_hancock</user_name>
```

```
<calling_station_id>00:50:56:8E:28:BD</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>0000002C</acct_session_id>
<audit_session_id>0ACB6BA10000002A165FD0C8</audit_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>ipepvpnuser</user_name>
<calling_station_id>172.23.130.89</calling_station_id>
<nas_ip_address>10.203.107.45</nas_ip_address>
<acct_session_id>A2000070</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
</activeSessionList>
```

经过身份验证的会话列表

您可以使用 AuthList API 调用来检索所有经过身份验证的当前活动会话的列表。

注意　可以显示且经过身份验证的最大活动终端会话数为 100000。

AuthList API 输出架构

此示例架构文件是 AuthList API 调用的输出，此调用用于检索指定时间段内（或者，使用“null/null”参数不指定时间）目标思科监控 ISE 节点上所有经过身份验证的当前活动会话的列表：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="activeSessionList" type="simpleActiveSessionList"/>

<xs:complexType name="simpleActiveSessionList">
    <xs:sequence>
      <xs:element name="activeSession" type="simpleActiveSession" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="noOfActiveSession" type="xs:int" use="required"/>
  </xs:complexType>

  <xs:complexType name="simpleActiveSession">
    <xs:sequence>
      <xs:element name="user_name" type="xs:string" minOccurs="0"/>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="acct_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="server" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 AuthList API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/Session/<specific-api-call>)，在目标节点的 URL 地址字段输入 AuthList API 调用：

注意　在下面两个示例中，第一个示例使用定义的 starttime 和 null 参数，显示指定的开始时间后经过身份验证的当前活动会话的列表。第二个示例使用 null/null 参数，显示所有经过身份验证的当前活动会话的列表。请参阅使用 null/null 选项的 AuthList API 调用返回的数据示例，第 2-8 页，其中显示此 API 调用的四个参数类型的示例。

```
https://acme123/admin/API/mnt/Session/AuthList/2010-12-14 15:33:15/null

https://acme123/admin/API/mnt/Session/AuthList/null/null
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

使用 null/null 选项的 AuthList API 调用返回的数据示例

以下示例展示的是您使用 null/null 选项发起 AuthList API 调用时所返回的经过身份验证的当前活动会话的列表：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<activeSessionList noOfActiveSession="3">
-
<activeSession>
<user_name>ipepwlcuser</user_name>
<calling_station_id>00:26:82:7B:D2:51</calling_station_id>
<nas_ip_address>10.203.107.10</nas_ip_address>
<audit_session_id>0acb6b0c000000174D07F487</audit_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>tom_wolfe</user_name>
<calling_station_id>00:50:56:8E:28:BD</calling_station_id>
```

```
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000035</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>graham_hancock</user_name>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000033</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
</activeSessionList>
```

使用 endtime/null 选项的 AuthList API 调用返回的数据示例

以下示例展示的是您使用 endtime/null 选项发起 AuthList API 调用时所返回的经过身份验证的当前活动会话的列表：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<activeSessionList noOfActiveSession="3">
-
<activeSession>
<user_name>ipepwlcuser</user_name>
<calling_station_id>00:26:82:7B:D2:51</calling_station_id>
<nas_ip_address>10.203.107.10</nas_ip_address>
<audit_session_id>0acb6b0c0000001F4D08085A</audit_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>hunter_thompson</user_name>
<calling_station_id>00:50:56:8E:28:BD</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000035</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>bob_ludlum</user_name>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000033</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
</activeSessionList>
```

使用 null/starttime 选项的 AuthList API 调用返回的数据示例

以下示例展示的是您使用 null/starttime 选项发起 AuthList API 调用时所返回的经过身份验证的当前活动会话的列表：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<activeSessionList noOfActiveSession="3">
-
```

```
<activeSession>
<user_name>ipepwlcuser</user_name>
<calling_station_id>00:26:82:7B:D2:51</calling_station_id>
<nas_ip_address>10.203.107.10</nas_ip_address>
<audit_session_id>0acb6b0c0000001F4D08085A</audit_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>bob_ludlum</user_name>
<calling_station_id>00:50:56:8E:28:BD</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000035</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>tom_wolfe</user_name>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000033</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
</activeSessionList>
```

使用 statttime/endtime 选项的 AuthList API 调用返回的数据示例

以下示例展示的是您使用 starttime/endtime 选项发起 AuthList API 调用时所返回的经过身份验证的当前活动会话的列表：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.
-
<activeSessionList noOfActiveSession="3">
-
<activeSession>
<user_name>ipepwlcuser</user_name>
<calling_station_id>00:26:82:7B:D2:51</calling_station_id>
<nas_ip_address>10.203.107.10</nas_ip_address>
<audit_session_id>0acb6b0c0000001F4D08085A</audit_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>graham_hancock</user_name>
<calling_station_id>00:50:56:8E:28:BD</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000035</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
-
<activeSession>
<user_name>hunter_thompson</user_name>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_ip_address>10.203.107.161</nas_ip_address>
<acct_session_id>00000033</acct_session_id>
<server>HAREESH-R6-1-PDP2</server>
</activeSession>
</activeSessionList>
```

详细会话属性 API 调用

通过以下详细会话属性 API 调用，您可以快速搜索最新会话以了解关键信息，如下列信息：

- MAC 地址会话搜索 (MACAddress)
- 用户名会话搜索 (UserName)
- NAS IP 地址会话搜索（与目标监控 ISE 节点关联的 IPAddress）
- 审核会话 ID 搜索 (Audit Session ID)

MAC 地址会话搜索

您可以使用 MACAddress API 调用从当前活动会话检索指定的 MAC 地址。此 API 调用列出从节点数据库表格中提取的各种会话相关信息。

MACAddress API 输出架构

此示例架构文件是用于从当前活动会话检索指定 MAC 地址的 MACAddress API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionParameters" type="restsdStatus"/>

  <xs:complexType name="restsdStatus">
    <xs:sequence>
      <xs:element name="passed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="failed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="user_name" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="failure_reason" type="xs:string" minOccurs="0"/>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_group" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_name" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_server" type="xs:string" minOccurs="0"/>
      <xs:element name="authen_protocol" type="xs:string" minOccurs="0"/>
      <xs:element name="framed_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_groups" type="xs:string" minOccurs="0"/>
      <xs:element name="access_service" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="authentication_method" type="xs:string" minOccurs="0"/>
      <xs:element name="execution_steps" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_response" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_identifier" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_policy_compliance" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_id" type="xs:long" minOccurs="0"/>
      <xs:element name="auth_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="message_code" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="service_selection_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="authorization_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="response" type="xs:string" minOccurs="0"/>
      <xs:element name="service_type" type="xs:string" minOccurs="0"/>
      <xs:element name="cts_security_group" type="xs:string" minOccurs="0"/>
```

```
<xs:element name="use_case" type="xs:string" minOccurs="0"/>
<xs:element name="cisco_av_pair" type="xs:string" minOccurs="0"/>
<xs:element name="ad_domain" type="xs:string" minOccurs="0"/>
<xs:element name="acs_username" type="xs:string" minOccurs="0"/>
<xs:element name="radius_username" type="xs:string" minOccurs="0"/>
<xs:element name="nac_role" type="xs:string" minOccurs="0"/>
<xs:element name="nac_username" type="xs:string" minOccurs="0"/>
<xs:element name="nac_posture_token" type="xs:string" minOccurs="0"/>
<xs:element name="nac_radius_is_user_auth" type="xs:string" minOccurs="0"/>
<xs:element name="selected_posture_server" type="xs:string" minOccurs="0"/>
<xs:element name="selected_identity_store" type="xs:string" minOccurs="0"/>
<xs:element name="authentication_identity_store" type="xs:string" minOccurs="0"/>
<xs:element name="azn_exp_pol_matched_rule" type="xs:string" minOccurs="0"/>
<xs:element name="ext_pol_server_matched_rule" type="xs:string" minOccurs="0"/>
<xs:element name="grp_mapping_pol_matched_rule" type="xs:string" minOccurs="0"/>
<xs:element name="identity_policy_matched_rule" type="xs:string" minOccurs="0"/>
<xs:element name="nas_port_type" type="xs:string" minOccurs="0"/>
<xs:element name="query_identity_stores" type="xs:string" minOccurs="0"/>
<xs:element name="selected_azn_profiles" type="xs:string" minOccurs="0"/>
<xs:element name="sel_exp_azn_profiles" type="xs:string" minOccurs="0"/>
<xs:element name="selected_query_identity_stores" type="xs:string" minOccurs="0"/>
<xs:element name="eap_tunnel" type="xs:string" minOccurs="0"/>
<xs:element name="tunnel_details" type="xs:string" minOccurs="0"/>
<xs:element name="cisco_h323_attributes" type="xs:string" minOccurs="0"/>
<xs:element name="cisco_ssg_attributes" type="xs:string" minOccurs="0"/>
<xs:element name="other_attributes" type="xs:string" minOccurs="0"/>
<xs:element name="response_time" type="xs:long" minOccurs="0"/>
<xs:element name="nad_failure" type="xs:anyType" minOccurs="0"/>
<xs:element name="destination_ip_address" type="xs:string" minOccurs="0"/>
<xs:element name="acct_id" type="xs:long" minOccurs="0"/>
<xs:element name="acct_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
<xs:element name="acct_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
<xs:element name="acct_session_id" type="xs:string" minOccurs="0"/>
<xs:element name="acct_status_type" type="xs:string" minOccurs="0"/>
<xs:element name="acct_session_time" type="xs:long" minOccurs="0"/>
<xs:element name="acct_input_octets" type="xs:string" minOccurs="0"/>
<xs:element name="acct_output_octets" type="xs:string" minOccurs="0"/>
<xs:element name="acct_input_packets" type="xs:long" minOccurs="0"/>
<xs:element name="acct_output_packets" type="xs:long" minOccurs="0"/>
<xs:element name="acct_class" type="xs:string" minOccurs="0"/>
<xs:element name="acct_terminate_cause" type="xs:string" minOccurs="0"/>
<xs:element name="acct_multi_session_id" type="xs:string" minOccurs="0"/>
<xs:element name="acct_authentic" type="xs:string" minOccurs="0"/>
<xs:element name="termination_action" type="xs:string" minOccurs="0"/>
<xs:element name="session_timeout" type="xs:string" minOccurs="0"/>
<xs:element name="idle_timeout" type="xs:string" minOccurs="0"/>
<xs:element name="acct_interim_interval" type="xs:string" minOccurs="0"/>
<xs:element name="acct_delay_time" type="xs:string" minOccurs="0"/>
<xs:element name="event_timestamp" type="xs:string" minOccurs="0"/>
<xs:element name="acct_tunnel_connection" type="xs:string" minOccurs="0"/>
<xs:element name="acct_tunnel_packet_lost" type="xs:string" minOccurs="0"/>
<xs:element name="security_group" type="xs:string" minOccurs="0"/>
<xs:element name="cisco_h323_setup_time" type="xs:dateTime" minOccurs="0"/>
<xs:element name="cisco_h323_connect_time" type="xs:dateTime" minOccurs="0"/>
<xs:element name="cisco_h323_disconnect_time" type="xs:dateTime" minOccurs="0"/>
<xs:element name="framed_protocol" type="xs:string" minOccurs="0"/>
<xs:element name="started" type="xs:anyType" minOccurs="0"/>
<xs:element name="stopped" type="xs:anyType" minOccurs="0"/>
<xs:element name="ckpt_id" type="xs:long" minOccurs="0"/>
<xs:element name="type" type="xs:long" minOccurs="0"/>
<xs:element name="nad_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
<xs:element name="vlan" type="xs:string" minOccurs="0"/>
<xs:element name="dacl" type="xs:string" minOccurs="0"/>
<xs:element name="authentication_type" type="xs:string" minOccurs="0"/>
```

```
      <xs:element name="interface_name" type="xs:string" minOccurs="0"/>
      <xs:element name="reason" type="xs:string" minOccurs="0"/>
      <xs:element name="endpoint_policy" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 MACAddress API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>/<macaddress>)，在目标节点的 URL 地址字段输入 MACAddress API 调用：

```
https://acme123/admin/API/mnt/Session/MACAddress/0A:0B:0C:0D:0E:0F
```

注意　确保使用 XX:XX:XX:XX:XX:XX 格式指定 MAC 地址。

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

MACAddress API 调用返回的数据示例

以下示例展示的是您发起 ActiveList API 调用时从活动会话列表返回的会话相关数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<sessionParameters>
<passed xsi:type="xs:boolean">true</passed>
<failed xsi:type="xs:boolean">false</failed>
<user_name>hunter_thompson</user_name>
<nas_ip_address>10.203.107.161</nas_ip_address>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_port>50115</nas_port>
<identity_group>Profiled</identity_group>
<network_device_name>Core-Switch</network_device_name>
<acs_server>HAREESH-R6-1-PDP2</acs_server>
```

```
<authen_protocol>Lookup</authen_protocol>
-
<network_device_groups>
Device Type#All Device Types,Location#All Locations
</network_device_groups>
<access_service>RADIUS</access_service>
<auth_acs_timestamp>2010-12-15T02:11:12.359Z</auth_acs_timestamp>
<authentication_method>mab</authentication_method>
-
<execution_steps>
11001,11017,11027,15008,15048,15004,15041,15004,15013,24209,24211,22037,15036,15048,15048,
15004,15016,11022,11002
</execution_steps>
<audit_session_id>0ACB6BA1000000351BBFBF8B</audit_session_id>
<nas_port_id>GigabitEthernet1/0/15</nas_port_id>
<nac_policy_compliance>Pending</nac_policy_compliance>
<auth_id>1291240762077361</auth_id>
<auth_acsview_timestamp>2010-12-15T02:11:12.360Z</auth_acsview_timestamp>
<message_code>5200</message_code>
<acs_session_id>HAREESH-R6-1-PDP2/81148292/681</acs_session_id>
<service_selection_policy>MAB</service_selection_policy>
<identity_store>Internal Hosts</identity_store>
-
<response>
{UserName=00-14-BF-5A-0C-03; User-Name=00-14-BF-5A-0C-03;
State=ReauthSession:0ACB6BA1000000351BBFBF8B;
Class=CACS:0ACB6BA1000000351BBFBF8B:HAREESH-R6-1-PDP2/81148292/681;
Termination-Action=RADIUS-Request; cisco-av-pair=url-redirect-acl=ACL-WEBAUTH-REDIRECT;
cisco-av-pair=url-redirect=https://HAREESH-R6-1-PDP2.cisco.com:8443/guestportal/gateway?se
ssionId=0ACB6BA1000000351BBFBF8B&action=cwa;
cisco-av-pair=ACS:CiscoSecure-Defined-ACL=#ACSACL#-IP-ACL-DENY-4ced8390; }
</response>
<service_type>Call Check</service_type>
<use_case>Host Lookup</use_case>
<cisco_av_pair>audit-session-id=0ACB6BA1000000351BBFBF8B</cisco_av_pair>
<acs_username>00:14:BF:5A:0C:03</acs_username>
<radius_username>00:14:BF:5A:0C:03</radius_username>
<selected_identity_store>Internal Hosts</selected_identity_store>
<authentication_identity_store>Internal Hosts</authentication_identity_store>
<identity_policy_matched_rule>Default</identity_policy_matched_rule>
<nas_port_type>Ethernet</nas_port_type>
<selected_azn_profiles>CWA</selected_azn_profiles>
-
<other_attributes>
ConfigVersionId=44,DestinationIPAddress=10.203.107.162,DestinationPort=1812,Protocol=Radiu
s,Framed-MTU=1500,EAP-Key-Name=,CPMSessionID=0ACB6BA1000000351BBFBF8B,CPMSessionID=0ACB6BA
1000000351BBFBF8B,EndPointMACAddress=00-14-BF-5A-0C-03,HostIdentityGroup=Endpoint Identity
Groups:Profiled,Device Type=Device Type#All Device Types,Location=Location#All
Locations,Model Name=Unknown,Software Version=Unknown,Device IP
Address=10.203.107.161,Called-Station-ID=04:FE:7F:7F:C0:8F
</other_attributes>
<response_time>77</response_time>
<acct_id>1291240762077386</acct_id>
<acct_acs_timestamp>2010-12-15T02:12:30.779Z</acct_acs_timestamp>
<acct_acsview_timestamp>2010-12-15T02:12:30.780Z</acct_acsview_timestamp>
<acct_session_id>00000038</acct_session_id>
<acct_status_type>Interim-Update</acct_status_type>
<acct_session_time>78</acct_session_time>
<acct_input_octets>13742</acct_input_octets>
<acct_output_octets>6277</acct_output_octets>
<acct_input_packets>108</acct_input_packets>
<acct_output_packets>66</acct_output_packets>
-
<acct_class>
```

```
CACS:0ACB6BA1000000351BBFBF8B:HAREESH-R6-1-PDP2/81148292/681
</acct_class>
<acct_delay_time>0</acct_delay_time>
<started xsi:type="xs:boolean">false</started>
<stopped xsi:type="xs:boolean">false</stopped>
</sessionParameters>
```

用户名会话搜索

您可以使用 UserName API 调用从当前活动会话检索指定用户名。此 API 将列出从节点数据库表格中提取的各种会话相关信息。

UserName API 输出架构

此示例架构文件是用于从当前活动会话检索指定用户名的 UserName API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionParameters" type="restsdStatus"/>

  <xs:complexType name="restsdStatus">
    <xs:sequence>
      <xs:element name="passed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="failed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="user_name" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="failure_reason" type="xs:string" minOccurs="0"/>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_group" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_name" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_server" type="xs:string" minOccurs="0"/>
      <xs:element name="authen_protocol" type="xs:string" minOccurs="0"/>
      <xs:element name="framed_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_groups" type="xs:string" minOccurs="0"/>
      <xs:element name="access_service" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="authentication_method" type="xs:string" minOccurs="0"/>
      <xs:element name="execution_steps" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_response" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_identifier" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_policy_compliance" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_id" type="xs:long" minOccurs="0"/>
      <xs:element name="auth_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="message_code" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="service_selection_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="authorization_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="response" type="xs:string" minOccurs="0"/>
      <xs:element name="service_type" type="xs:string" minOccurs="0"/>
      <xs:element name="cts_security_group" type="xs:string" minOccurs="0"/>
      <xs:element name="use_case" type="xs:string" minOccurs="0"/>
      <xs:element name="cisco_av_pair" type="xs:string" minOccurs="0"/>
      <xs:element name="ad_domain" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_username" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_username" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_role" type="xs:string" minOccurs="0"/>
```

```
        <xs:element name="nac_username" type="xs:string" minOccurs="0"/>
        <xs:element name="nac_posture_token" type="xs:string" minOccurs="0"/>
        <xs:element name="nac_radius_is_user_auth" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_posture_server" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_identity_store" type="xs:string" minOccurs="0"/>
        <xs:element name="authentication_identity_store" type="xs:string" minOccurs="0"/>
        <xs:element name="azn_exp_pol_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="ext_pol_server_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="grp_mapping_pol_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="identity_policy_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_port_type" type="xs:string" minOccurs="0"/>
        <xs:element name="query_identity_stores" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="sel_exp_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_query_identity_stores" type="xs:string" minOccurs="0"/>
        <xs:element name="eap_tunnel" type="xs:string" minOccurs="0"/>
        <xs:element name="tunnel_details" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_h323_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_ssg_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="other_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="response_time" type="xs:long" minOccurs="0"/>
        <xs:element name="nad_failure" type="xs:anyType" minOccurs="0"/>
        <xs:element name="destination_ip_address" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_id" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="acct_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="acct_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_status_type" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_session_time" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_input_octets" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_output_octets" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_input_packets" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_output_packets" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_class" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_terminate_cause" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_multi_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_authentic" type="xs:string" minOccurs="0"/>
        <xs:element name="termination_action" type="xs:string" minOccurs="0"/>
        <xs:element name="session_timeout" type="xs:string" minOccurs="0"/>
        <xs:element name="idle_timeout" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_interim_interval" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_delay_time" type="xs:string" minOccurs="0"/>
        <xs:element name="event_timestamp" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_tunnel_connection" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_tunnel_packet_lost" type="xs:string" minOccurs="0"/>
        <xs:element name="security_group" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_h323_setup_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="cisco_h323_connect_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="cisco_h323_disconnect_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="framed_protocol" type="xs:string" minOccurs="0"/>
        <xs:element name="started" type="xs:anyType" minOccurs="0"/>
        <xs:element name="stopped" type="xs:anyType" minOccurs="0"/>
        <xs:element name="ckpt_id" type="xs:long" minOccurs="0"/>
        <xs:element name="type" type="xs:long" minOccurs="0"/>
        <xs:element name="nad_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="vlan" type="xs:string" minOccurs="0"/>
        <xs:element name="dacl" type="xs:string" minOccurs="0"/>
        <xs:element name="authentication_type" type="xs:string" minOccurs="0"/>
        <xs:element name="interface_name" type="xs:string" minOccurs="0"/>
        <xs:element name="reason" type="xs:string" minOccurs="0"/>
        <xs:element name="endpoint_policy" type="xs:string" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
```

发起 UserName API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>/<username>)，在目标节点的 URL 地址字段输入 UserName API 调用：

```
https://acme123/admin/API/mnt/Session/UserName/graham_hancock
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

UserName API 调用返回的数据示例

以下示例展示的是您发起 UserName API 调用时从活动会话列表返回的会话相关数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<sessionParameters>
<passed xsi:type="xs:boolean">true</passed>
<failed xsi:type="xs:boolean">false</failed>
<user_name>graham_hancock</user_name>
<nas_ip_address>10.203.107.161</nas_ip_address>
<calling_station_id>00:14:BF:5A:0C:03</calling_station_id>
<nas_port>50115</nas_port>
<identity_group>Profiled</identity_group>
<network_device_name>Core-Switch</network_device_name>
<acs_server>HAREESH-R6-1-PDP2</acs_server>
<authen_protocol>Lookup</authen_protocol>
-
<network_device_groups>
Device Type#All Device Types,Location#All Locations
</network_device_groups>
<access_service>RADIUS</access_service>
<auth_acs_timestamp>2010-12-15T02:11:12.359Z</auth_acs_timestamp>
<authentication_method>mab</authentication_method>
-
<execution_steps>
11001,11017,11027,15008,15048,15004,15041,15004,15013,24209,24211,22037,15036,15048,15048,
15004,15016,11022,11002
</execution_steps>
```

```
<audit_session_id>0ACB6BA1000000351BBFBF8B</audit_session_id>
<nas_port_id>GigabitEthernet1/0/15</nas_port_id>
<nac_policy_compliance>Pending</nac_policy_compliance>
<auth_id>1291240762077361</auth_id>
<auth_acsview_timestamp>2010-12-15T02:11:12.360Z</auth_acsview_timestamp>
<message_code>5200</message_code>
<acs_session_id>HAREESH-R6-1-PDP2/81148292/681</acs_session_id>
<service_selection_policy>MAB</service_selection_policy>
<identity_store>Internal Hosts</identity_store>
-
<response>
{UserName=graham_hancock; User-Name=graham_hancock;
State=ReauthSession:0ACB6BA1000000351BBFBF8B;
Class=CACS:0ACB6BA1000000351BBFBF8B:HAREESH-R6-1-PDP2/81148292/681;
Termination-Action=RADIUS-Request; cisco-av-pair=url-redirect-acl=ACL-WEBAUTH-REDIRECT;
cisco-av-pair=url-redirect=https://HAREESH-R6-1-PDP2.cisco.com:8443/guestportal/gateway?se
ssionId=0ACB6BA1000000351BBFBF8B&action=cwa;
cisco-av-pair=ACS:CiscoSecure-Defined-ACL=#ACSACL#-IP-ACL-DENY-4ced8390; }
</response>
<service_type>Call Check</service_type>
<use_case>Host Lookup</use_case>
<cisco_av_pair>audit-session-id=0ACB6BA1000000351BBFBF8B</cisco_av_pair>
<acs_username>graham_hancock</acs_username>
<radius_username>00:14:BF:5A:0C:03</radius_username>
<selected_identity_store>Internal Hosts</selected_identity_store>
<authentication_identity_store>Internal Hosts</authentication_identity_store>
<identity_policy_matched_rule>Default</identity_policy_matched_rule>
<nas_port_type>Ethernet</nas_port_type>
<selected_azn_profiles>CWA</selected_azn_profiles>
-
<other_attributes>
ConfigVersionId=44,DestinationIPAddress=10.203.107.162,DestinationPort=1812,Protocol=Radiu
s,Framed-MTU=1500,EAP-Key-Name=,CPMSessionID=0ACB6BA1000000351BBFBF8B,CPMSessionID=0ACB6BA
1000000351BBFBF8B,EndPointMACAddress=00-14-BF-5A-0C-03,HostIdentityGroup=Endpoint Identity
Groups:Profiled,Device Type=Device Type#All Device Types,Location=Location#All
Locations,Model Name=Unknown,Software Version=Unknown,Device IP
Address=10.203.107.161,Called-Station-ID=04:FE:7F:7F:C0:8F
</other_attributes>
<response_time>77</response_time>
<acct_id>1291240762077386</acct_id>
<acct_acs_timestamp>2010-12-15T02:12:30.779Z</acct_acs_timestamp>
<acct_acsview_timestamp>2010-12-15T02:12:30.780Z</acct_acsview_timestamp>
<acct_session_id>00000038</acct_session_id>
<acct_status_type>Interim-Update</acct_status_type>
<acct_session_time>78</acct_session_time>
<acct_input_octets>13742</acct_input_octets>
<acct_output_octets>6277</acct_output_octets>
<acct_input_packets>108</acct_input_packets>
<acct_output_packets>66</acct_output_packets>
-
<acct_class>
CACS:0ACB6BA1000000351BBFBF8B:HAREESH-R6-1-PDP2/81148292/681
</acct_class>
<acct_delay_time>0</acct_delay_time>
<started xsi:type="xs:boolean">false</started>
<stopped xsi:type="xs:boolean">false</stopped>
</sessionParameters>
```

NAS IP 地址会话搜索

您可以使用 IPAddress API 调用从当前会话检索指定 NAS IP 地址的数据。此 API 将列出从节点数据库表格中提取的各种会话相关信息。

IPAddress API 输出架构

此示例架构文件是用于从当前活动会话检索指定 NAS IP 地址的 IPAddress API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionParameters" type="restsdStatus"/>

  <xs:complexType name="restsdStatus">
    <xs:sequence>
      <xs:element name="passed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="failed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="user_name" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="failure_reason" type="xs:string" minOccurs="0"/>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_group" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_name" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_server" type="xs:string" minOccurs="0"/>
      <xs:element name="authen_protocol" type="xs:string" minOccurs="0"/>
      <xs:element name="framed_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_groups" type="xs:string" minOccurs="0"/>
      <xs:element name="access_service" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="authentication_method" type="xs:string" minOccurs="0"/>
      <xs:element name="execution_steps" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_response" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_identifier" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_policy_compliance" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_id" type="xs:long" minOccurs="0"/>
      <xs:element name="auth_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="message_code" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="service_selection_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="authorization_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="response" type="xs:string" minOccurs="0"/>
      <xs:element name="service_type" type="xs:string" minOccurs="0"/>
      <xs:element name="cts_security_group" type="xs:string" minOccurs="0"/>
      <xs:element name="use_case" type="xs:string" minOccurs="0"/>
      <xs:element name="cisco_av_pair" type="xs:string" minOccurs="0"/>
      <xs:element name="ad_domain" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_username" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_username" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_role" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_username" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_posture_token" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_radius_is_user_auth" type="xs:string" minOccurs="0"/>
      <xs:element name="selected_posture_server" type="xs:string" minOccurs="0"/>
      <xs:element name="selected_identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="authentication_identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="azn_exp_pol_matched_rule" type="xs:string" minOccurs="0"/>
```

```
        <xs:element name="ext_pol_server_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="grp_mapping_pol_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="identity_policy_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_port_type" type="xs:string" minOccurs="0"/>
        <xs:element name="query_identity_stores" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="sel_exp_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_query_identity_stores" type="xs:string" minOccurs="0"/>
        <xs:element name="eap_tunnel" type="xs:string" minOccurs="0"/>
        <xs:element name="tunnel_details" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_h323_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_ssg_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="other_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="response_time" type="xs:long" minOccurs="0"/>
        <xs:element name="nad_failure" type="xs:anyType" minOccurs="0"/>
        <xs:element name="destination_ip_address" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_id" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="acct_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="acct_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_status_type" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_session_time" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_input_octets" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_output_octets" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_input_packets" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_output_packets" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_class" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_terminate_cause" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_multi_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_authentic" type="xs:string" minOccurs="0"/>
        <xs:element name="termination_action" type="xs:string" minOccurs="0"/>
        <xs:element name="session_timeout" type="xs:string" minOccurs="0"/>
        <xs:element name="idle_timeout" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_interim_interval" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_delay_time" type="xs:string" minOccurs="0"/>
        <xs:element name="event_timestamp" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_tunnel_connection" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_tunnel_packet_lost" type="xs:string" minOccurs="0"/>
        <xs:element name="security_group" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_h323_setup_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="cisco_h323_connect_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="cisco_h323_disconnect_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="framed_protocol" type="xs:string" minOccurs="0"/>
        <xs:element name="started" type="xs:anyType" minOccurs="0"/>
        <xs:element name="stopped" type="xs:anyType" minOccurs="0"/>
        <xs:element name="ckpt_id" type="xs:long" minOccurs="0"/>
        <xs:element name="type" type="xs:long" minOccurs="0"/>
        <xs:element name="nad_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="vlan" type="xs:string" minOccurs="0"/>
        <xs:element name="dacl" type="xs:string" minOccurs="0"/>
        <xs:element name="authentication_type" type="xs:string" minOccurs="0"/>
        <xs:element name="interface_name" type="xs:string" minOccurs="0"/>
        <xs:element name="reason" type="xs:string" minOccurs="0"/>
        <xs:element name="endpoint_policy" type="xs:string" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
```

发起 NAS IPAddress API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>/<nasipaddress>)，在目标节点的 URL 地址字段输入 IPAddress API 调用：

```
https://acme123/admin/API/mnt/Session/IPAddress/10.10.10.10
```

注意　确保使用 xxx.xxx.xxx.xxx 格式指定 NAS IP 地址。

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

IPAddress API 调用返回的数据示例

以下示例展示的是您发起 IPAddress API 调用时从活动会话列表返回的会话相关数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.
-
<sessionParameters>
<passed xsi:type="xs:boolean">true</passed>
<failed xsi:type="xs:boolean">false</failed>
<user_name>ipepvpnuser</user_name>
<nas_ip_address>10.10.10.10</nas_ip_address>
<calling_station_id>172.23.130.90</calling_station_id>
<nas_port>1015</nas_port>
<identity_group>iPEP-VPN-Group</identity_group>
<network_device_name>iPEP-HA-Routed</network_device_name>
<acs_server>HAREESH-R6-1-PDP2</acs_server>
<authen_protocol>PAP_ASCII</authen_protocol>
-
<network_device_groups>
Device Type#All Device Types,Location#All Locations
</network_device_groups>
<access_service>RADIUS</access_service>
<auth_acs_timestamp>2010-12-15T19:57:29.885Z</auth_acs_timestamp>
<authentication_method>PAP_ASCII</authentication_method>
-
```

```
<execution_steps>
11001,11017,15008,15048,15048,15004,15041,15004,15013,24210,24212,22037,15036,15048,15048,
15004,15016,11002
</execution_steps>
<audit_session_id>0acb6be4000000044D091DA9</audit_session_id>
<nac_policy_compliance>NotApplicable</nac_policy_compliance>
<auth_id>1291240762083580</auth_id>
<auth_acsview_timestamp>2010-12-15T19:57:29.887Z</auth_acsview_timestamp>
<message_code>5200</message_code>
<acs_session_id>HAREESH-R6-1-PDP2/81148292/693</acs_session_id>
<service_selection_policy>iPEP-VPN</service_selection_policy>
<identity_store>Internal Users</identity_store>
-
<response>
{User-Name=ipepvpnuser; State=ReauthSession:0acb6be4000000044D091DA9;
Class=CACS:0acb6be4000000044D091DA9:HAREESH-R6-1-PDP2/81148292/693;
Termination-Action=RADIUS-Request; }
</response>
<service_type>Framed</service_type>
-
<cisco_av_pair>
audit-session-id=0acb6be4000000044D091DA9,ipep-proxy=true
</cisco_av_pair>
<acs_username>ipepvpnuser</acs_username>
<radius_username>ipepvpnuser</radius_username>
<selected_identity_store>Internal Users</selected_identity_store>
<authentication_identity_store>Internal Users</authentication_identity_store>
<identity_policy_matched_rule>Default</identity_policy_matched_rule>
<nas_port_type>Virtual</nas_port_type>
<selected_azn_profiles>iPEP-Unknown-Auth-Profile</selected_azn_profiles>
<tunnel_details>Tunnel-Client-Endpoint=(tag=0) 172.23.130.90</tunnel_details>
-
<other_attributes>
ConfigVersionId=44,DestinationIPAddress=10.203.107.162,DestinationPort=1812,Protocol=Radiu
s,Framed-Protocol=PPP,Proxy-State=Cisco Secure
ACS9e733142-070a-11e0-c000-000000000000-2906094480-3222,CPMSessionID=0acb6be4000000044D091
DA9,CPMSessionID=0acb6be4000000044D091DA9,Device Type=Device Type#All Device
Types,Location=Location#All Locations,Model Name=Unknown,Software Version=Unknown,Device
IP Address=10.203.107.228,Called-Station-ID=172.23.130.94
</other_attributes>
<response_time>20</response_time>
<acct_id>1291240762083582</acct_id>
<acct_acs_timestamp>2010-12-15T19:57:30.281Z</acct_acs_timestamp>
<acct_acsview_timestamp>2010-12-15T19:57:30.283Z</acct_acsview_timestamp>
<acct_session_id>F1800007</acct_session_id>
<acct_status_type>Start</acct_status_type>
-
<acct_class>
CACS:0acb6be4000000044D091DA9:HAREESH-R6-1-PDP2/81148292/693
</acct_class>
<acct_delay_time>0</acct_delay_time>
<framed_protocol>PPP</framed_protocol>
<started xsi:type="xs:boolean">true</started>
<stopped xsi:type="xs:boolean">false</stopped>
</sessionParameters>
```

审核会话 ID 搜索

您可以使用 Audit Session ID API 调用从当前活动会话检索指定的审核会话。此 API 调用列出从节点数据库表格中提取的各种会话相关信息。

Audit Session ID API 输出架构

此示例架构文件是用于从当前活动会话检索指定审核会话 ID 的 Audit Session ID API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="sessionParameters" type="restsdStatus"/>

  <xs:complexType name="restsdStatus">
    <xs:sequence>
      <xs:element name="passed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="failed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="user_name" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="failure_reason" type="xs:string" minOccurs="0"/>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_group" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_name" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_server" type="xs:string" minOccurs="0"/>
      <xs:element name="authen_protocol" type="xs:string" minOccurs="0"/>
      <xs:element name="framed_ip_address" type="xs:string" minOccurs="0"/>
      <xs:element name="network_device_groups" type="xs:string" minOccurs="0"/>
      <xs:element name="access_service" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="authentication_method" type="xs:string" minOccurs="0"/>
      <xs:element name="execution_steps" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_response" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_identifier" type="xs:string" minOccurs="0"/>
      <xs:element name="nas_port_id" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_policy_compliance" type="xs:string" minOccurs="0"/>
      <xs:element name="auth_id" type="xs:long" minOccurs="0"/>
      <xs:element name="auth_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="message_code" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="service_selection_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="authorization_policy" type="xs:string" minOccurs="0"/>
      <xs:element name="identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="response" type="xs:string" minOccurs="0"/>
      <xs:element name="service_type" type="xs:string" minOccurs="0"/>
      <xs:element name="cts_security_group" type="xs:string" minOccurs="0"/>
      <xs:element name="use_case" type="xs:string" minOccurs="0"/>
      <xs:element name="cisco_av_pair" type="xs:string" minOccurs="0"/>
      <xs:element name="ad_domain" type="xs:string" minOccurs="0"/>
      <xs:element name="acs_username" type="xs:string" minOccurs="0"/>
      <xs:element name="radius_username" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_role" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_username" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_posture_token" type="xs:string" minOccurs="0"/>
      <xs:element name="nac_radius_is_user_auth" type="xs:string" minOccurs="0"/>
      <xs:element name="selected_posture_server" type="xs:string" minOccurs="0"/>
      <xs:element name="selected_identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="authentication_identity_store" type="xs:string" minOccurs="0"/>
      <xs:element name="azn_exp_pol_matched_rule" type="xs:string" minOccurs="0"/>
```

```
        <xs:element name="ext_pol_server_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="grp_mapping_pol_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="identity_policy_matched_rule" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_port_type" type="xs:string" minOccurs="0"/>
        <xs:element name="query_identity_stores" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="sel_exp_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_query_identity_stores" type="xs:string" minOccurs="0"/>
        <xs:element name="eap_tunnel" type="xs:string" minOccurs="0"/>
        <xs:element name="tunnel_details" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_h323_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_ssg_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="other_attributes" type="xs:string" minOccurs="0"/>
        <xs:element name="response_time" type="xs:long" minOccurs="0"/>
        <xs:element name="nad_failure" type="xs:anyType" minOccurs="0"/>
        <xs:element name="destination_ip_address" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_id" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_acs_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="acct_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="acct_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_status_type" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_session_time" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_input_octets" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_output_octets" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_input_packets" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_output_packets" type="xs:long" minOccurs="0"/>
        <xs:element name="acct_class" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_terminate_cause" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_multi_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_authentic" type="xs:string" minOccurs="0"/>
        <xs:element name="termination_action" type="xs:string" minOccurs="0"/>
        <xs:element name="session_timeout" type="xs:string" minOccurs="0"/>
        <xs:element name="idle_timeout" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_interim_interval" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_delay_time" type="xs:string" minOccurs="0"/>
        <xs:element name="event_timestamp" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_tunnel_connection" type="xs:string" minOccurs="0"/>
        <xs:element name="acct_tunnel_packet_lost" type="xs:string" minOccurs="0"/>
        <xs:element name="security_group" type="xs:string" minOccurs="0"/>
        <xs:element name="cisco_h323_setup_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="cisco_h323_connect_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="cisco_h323_disconnect_time" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="framed_protocol" type="xs:string" minOccurs="0"/>
        <xs:element name="started" type="xs:anyType" minOccurs="0"/>
        <xs:element name="stopped" type="xs:anyType" minOccurs="0"/>
        <xs:element name="ckpt_id" type="xs:long" minOccurs="0"/>
        <xs:element name="type" type="xs:long" minOccurs="0"/>
        <xs:element name="nad_acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="vlan" type="xs:string" minOccurs="0"/>
        <xs:element name="dacl" type="xs:string" minOccurs="0"/>
        <xs:element name="authentication_type" type="xs:string" minOccurs="0"/>
        <xs:element name="interface_name" type="xs:string" minOccurs="0"/>
        <xs:element name="reason" type="xs:string" minOccurs="0"/>
        <xs:element name="endpoint_policy" type="xs:string" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
```

发起 Audit Session ID API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/Session/Active/SessionID/*<audit-session-id>*/0)，在目标节点的 URL 地址字段输入 Audit Session ID API 调用：

```
https://acme123/admin/API/mnt/Session/Active/SessionID/0A000A770000006B609A13A9/0
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

Audit Session ID API 调用返回的数据示例

以下示例展示的是您发起 Audit Session ID API 调用时从活动会话列表返回的会话相关数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.
-<activeSessionList noOfActiveSession="1">
    -<activeSession>
        <calling_station_id>00:50:56:10:13:02</calling_station_id>
        <session_state_bit>0</session_state_bit>
        <session_source>0</session_source>
        <acct_session_time>0</acct_session_time>
        <nas_ip_address>10.0.10.119</nas_ip_address>
        <nas_port_id>GigabitEthernet1/0/15</nas_port_id>
        <auth_method>dot1x</auth_method>
        <auth_protocol>PEAP (EAP-MSCHAPv2)</auth_protocol>
        <posture_status>Compliant</posture_status>
        <endpoint_policy>Undetermined</endpoint_policy>
        <server>acme123</server>
        <paks_in>0</paks_in>
        <paks_out>0</paks_out>
        <bytes_in>0</bytes_in>
        <bytes_out>0</bytes_out>
    </activeSession>
</activeSessionList>
```

过时会话

有些设备（如无线局域网控制器 [WLC]）可能允许存留过时会话 在这种情况下，您可以使用 HTTP **DELETE** API 调用来手动删除非活动会话。要执行此操作，请使用 **cURL**，它是一种用于使用 URL（HTTP、HTTPS）语法传输数据的免费第三方命令行工具。

ISE 不再跟踪这些会话。这是为了缓解以下情况：ISE 过长时间失去网络连接，并缺失大量来自 WLC/NAD 的记帐停止请求。您可以使用此 API 从 ISE 中清除此类过时信息。

注意　使用 HTTP 和 HTTPS 检索文件的免费实用程序 GNU Wget 不支持 HTTP **DELETE** API 调用。

删除过时会话

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址 >/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

注意　API 调用区分大小写，必须小心输入。变量 <mntnode> 代表思科监控 ISE 节点。

步骤 4　要手动删除 MAC 地址的过时会话，请在命令行发出以下 API 调用：

```
curl -X DELETE https://<mntnode>/admin/API/mnt/Session/Delete/MACAddress/<madaddress>
```

步骤 5　要手动删除会话 ID 的过时会话，请在命令行发出以下 API 调用：

```
curl -X DELETE https://<mntnode>/admin/API/mnt/Session/Delete/SessionID/<sid#>
```

步骤 6　要手动删除监控节点上的所有会话，请在命令行发出以下 API 调用：

```
curl -X DELETE https://<mntnode>/admin/API/mnt/Session/Delete/All
```

相关主题

- 验证监控节点，第 1-2 页

用于故障排除的查询 API

本章提供有关使用各个 Cisco Prime 网络控制系统 (NCS) REST API 调用的示例和说明。

Cisco Prime NCS API 调用

Cisco Prime NCS API 调用提供检索有关目标思科监控 ISE 节点会话的关键故障排除信息的机制，这些信息包括节点版本和类型、故障原因、身份验证状态和记帐状态。

使用查询 API 调用对 Cisco ISE 进行故障排除

Cisco Prime NCS 故障排除 API 调用向 Cisco ISE 部署中的目标思科监控 ISE 节点发送状态请求，并检索下列与诊断相关的信息：

- 节点版本和类型（使用 Version API 调用）
- 故障原因（使用 FailureReasons API 调用）
- 身份验证状态（使用 AuthStatus API 调用）
- 记帐状态（使用 AcctStatus API 调用）

节点版本和类型 API 调用

您可以使用 Version API 调用测试 REST 编程接口 (PI) 服务和每个节点的凭证。本节提供架构文件输出示例、通过发起此 API 调用来请求 Cisco ISE 软件版本和节点类型信息的程序，以及发起此 API 调用后返回的节点版本和类型示例。

节点类型可以是以下任意值：

- STANDALONE_MNT_NODE = 0
- ACTIVE_MNT_NODE = 1
- BACKUP_MNT_NODE = 2
- NOT_AN_MNT_NODE = 3

Version API 输出架构

此示例架构文件是将 Version API 调用发送到目标思科监控 ISE 节点后，此调用所返回的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="product" type="product"/>

  <xs:complexType name="product">
    <xs:sequence>
      <xs:element name="version" type="xs:string" minOccurs="0"/>
  <xs:element name="type_of_node" type="xs:int"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string"/>
  </xs:complexType>
</xs:schema>
```

发起 Version API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

如果未能成功登录，请点击 Login 页面上的 **Problem logging in?** 链接，然后按照步骤 2 中的说明操作。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>)，在目标节点的 URL 地址字段输入 Version API 调用：

```
https://acme123/admin/API/mnt/Version
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

Version API 调用返回的数据示例

以下示例展示了在目标思科监控 ISE 节点上发起 Version API 调用时返回的数据： 此 API 调用返回目标节点的以下两个值：

- 节点版本（本示例显示 1.0.3.032）。
- 思科监控 ISE 节点的类型（本示例显示“1”，表示活动的思科监控 ISE 节点）。

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<product name="Cisco Identity Services Engine">
<version>1.0.3.032</version>
<type_of_node>1</type_of_node>
</product>
```

故障原因 API 调用

您可以使用 FailureReasons API 调用返回在目标节点上完成的身份验证状态检查所返回的故障原因列表。本节提供了架构文件输出示例、通过发起此 API 调用请求思科监控 ISE 节点记录的所有故障原因列表的程序，以及发起此 API 调用后返回的故障原因示例。返回的每个故障原因包括表 3-1 中所显示的以下元素。

注意　有关使用 Cisco ISE 故障原因编辑器访问故障原因完整列表的详细信息，请参阅 Cisco ISE 故障原因报告，第 A-1 页。

表 3-1　　思科身份服务引擎的产品文档

故障原因元素	示例
故障原因 ID	<failureReason id="11011">
代码	<11011 RADIUS listener failed>
原因	<Could not open one or more of the ports used to receive RADIUS requests>
解决方法	<Ensure that the ports 1812, 1813, 1645 and 1646 are not being used by another process on the system>

注意　您也可以使用 Cisco ISE 用户界面（依次点击 **Monitor > Reports > Catalog > Failure Reasons**，系统将显示故障原因报告）检查故障原因报告。

FailureReasons API 输出架构

此示例架构文件是将请求发送到目标思科监控 ISE 节点后，FailureReasons API 调用的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="failureReasonList" type="failureReasonList"/>

  <xs:complexType name="failureReasonList">
    <xs:sequence>
      <xs:element name="failureReason" type="failureReason" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="failureReason">
    <xs:sequence>
```

```
      <xs:element name="code" type="xs:string" minOccurs="0"/>
      <xs:element name="cause" type="xs:string" minOccurs="0"/>
      <xs:element name="resolution" type="xs:string" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:string"/>
  </xs:complexType>
</xs:schema>
```

发起 FailureReasons API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址 >/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

如果未能成功登录，请点击 Login 页面上的 **Problem logging in?** 链接，然后按照步骤 2 中的说明操作。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>)，在目标节点的 URL 地址字段输入 FailureReasons API 调用：

```
https://acme123/admin/API/mnt/FailureReasons
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

FailureReasons API 调用返回的数据示例

以下示例展示了在目标思科监控 ISE 节点上发起 FailureReasons API 调用时返回的数据： 此 API 调用可以从目标节点返回故障原因列表，且每个故障原因通过故障 ID、故障代码、原因和解决方法（如果知道）进行定义。

注意　以下 FailureReasons API 调用示例仅显示可以返回的一小段数据示例。

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<failureReasonList>
-
<failureReason id="100001">
-
<code>
```

```
100001 AUTHMGR-5-FAIL Authorization failed for client
</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
</failureReason>
-
<failureReason id="100002">
-
<code>
100002 AUTHMGR-5-SECURITY_VIOLATION Security violation on the interface
</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
</failureReason>
-
<failureReason id="100003">
-
<code>
100003 AUTHMGR-5-UNAUTHORIZED Interface unauthorized
</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
</failureReason>
-
<failureReason id="100004">
-
<code>
100004 DOT1X-5-FAIL Authentication failed for client
</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
</failureReason>
-
<failureReason id="100005">
<code>100005 MAB-5-FAIL Authentication failed for client</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
</failureReason>
-
<failureReason id="100006">
-
<code>
100006 RADIUS-4-RADIUS_DEAD RADIUS server is not responding
</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
```

```
</failureReason>
-
<failureReason id="100007">
-
<code>
100007 EPM-6-POLICY_APP_FAILURE Interface ACL not configured
</code>
<cause>This may or may not be indicating a violation</cause>
-
<resolution>
Please review and resolve according to your organization's policy
</resolution>
</failureReason>
```

相关主题

- 验证监控节点，第 1-2 页
- 附录 A，“Cisco ISE 故障原因报告”

身份验证状态 API 调用

您可以使用 AuthStatus API 调用检查目标节点上会话的身份验证状态。与此 API 调用关联的查询需要搜索至少一个 MAC 地址以查找匹配项，并且用户可以配置所返回的指定 MAC 地址的最近记录限制。

本节提供了架构文件输出示例、通过发起此 API 调用请求在目标监控节点上搜索会话身份验证状态的程序，以及发起此 API 调用后返回的数据示例。

通过 uthStatus API 调用，您可以配置以下与搜索相关的参数：

- 持续时间 - 定义尝试搜索和检索与指定 MAC 地址关联的身份验证状态记录所持续的秒数。用户可配置的有效值范围是 1 至 864000 秒（10 天）。如果输入 0 秒，则指定默认持续时间 10 天。
- 记录数 - 定义为每个 MAC 地址搜索的会话记录数。用户可配置的有效值范围是 1 至 500 条记录。如果输入 0，则指定默认设置，即 200 条记录。

  注意　如果为持续时间和记录数参数指定的值均为 0，此 API 调用只会返回与指定 MAC 地址关联的最新身份验证会话记录。

  下面是包含持续时间和记录数两个属性的 URL 的一般形式示例：

  https://10.10.10.10/admin/API/mnt/AuthStatus/MACAddress/01:23:45:67:89:98/900000/2/All

- 属性 - 定义使用 AuthStatus API 调用搜索身份验证状态所返回的身份验证状态表格中包含的属性数。有效值包括 0（默认值）、All 或 user_name+acs_timestamp（请参阅 AuthStatus 架构示例 AcctStatus API 输出架构，第 3-12 页）。
  - 如果输入“0”，系统会返回表 3-2 中定义的属性。这些属性列于输出架构的 restAuthStatus 部分。
  - 如果输入“All”，系统会返回所有属性。这些属性列于输出架构的 fullRESTAuthStatus 部分。
  - 如果输入架构中为 user_name+acs_timestamp 列出的值，系统只会返回这些属性。user_name 和 acs_timestamp 属性列于输出架构的 restAuthStatus 部分。

表 3-2　身份验证状态表属性

属性	说明
name=”passed” 或 name=”failed”	身份验证状态结果： • 已通过 • 失败
name=”user_name”	用户名
name=”nas_ip_address”	网络访问设备的 IP 地址 / 主机名
name=”failure_reason”	会话身份验证失败的原因
name=”calling_station_id”	源 IP 地址
name=”nas_port”	网络访问服务器端口
name=”identity_group”	包含相关用户和主机的逻辑组
name=”network_device_name”	网络设备的名称
name=”acs_server”	Cisco ISE 设备的名称
name=”eap_authentication”	用于身份验证请求的可扩展身份验证协议 (EAP) 方法
name=”framed_ip_address”	为特定用户配置的地址
network_device_groups”	包含相关网络设备的逻辑组
name=”access_service”	应用的访问服务
name=”acs_timestamp”	与 Cisco ISE 身份验证请求关联的时间戳
name=”authentication_method”	确定身份验证中使用的方法
name=”execution_steps”	处理请求时所记录的每个诊断消息的消息代码列表
name=”radius_response”	RADIUS 响应的类型（例如，VLAN 或 ACL）
name=”audit_session_id”	身份验证会话的 ID
name=”nas_identifier”	与特定资源关联的网络访问服务器 (NAS)
name=”nas_port_id”	使用的 NAS 端口的 ID
name=”nac_policy_compliance”	反映状态评估状态（合规或不合规）
name=”selected_azn_profiles”	识别身份验证使用的配置文件
name=”service_type”	表示帧用户
name=”eap_tunnel”	用于 EAP 身份验证的隧道或外部方法
name=”message_code”	定义已处理请求结果的审核消息标识符
name=”destination_ip_address”	标识目标 IP 地址

AuthStatus API 输出架构

此示例架构文件是将 AuthStatus API 调用发送到目标思科监控 ISE 节点上的指定会话后，此调用所返回的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="authStatusOutputList" type="fullRESTAuthStatusOutputList"/>

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

```
    <xs:sequence>
      <xs:element name="authStatusList" type="fullRESTAuthStatusList" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="fullRESTAuthStatusList">
    <xs:sequence>
      <xs:element name="authStatusElements" type="fullRESTAuthStatus" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="key" type="xs:string"/>
  </xs:complexType>

  <xs:complexType name="fullRESTAuthStatus">
    <xs:complexContent>
      <xs:extension base="restAuthStatus">
        <xs:sequence>
          <xs:element name="id" type="xs:long" minOccurs="0"/>
          <xs:element name="acsview_timestamp" type="xs:dateTime" minOccurs="0"/>
          <xs:element name="acs_session_id" type="xs:string" minOccurs="0"/>
          <xs:element name="service_selection_policy" type="xs:string" minOccurs="0"/>
          <xs:element name="authorization_policy" type="xs:string" minOccurs="0"/>
          <xs:element name="identity_store" type="xs:string" minOccurs="0"/>
          <xs:element name="response" type="xs:string" minOccurs="0"/>
          <xs:element name="cts_security_group" type="xs:string" minOccurs="0"/>
          <xs:element name="use_case" type="xs:string" minOccurs="0"/>
          <xs:element name="cisco_av_pair" type="xs:string" minOccurs="0"/>
          <xs:element name="ad_domain" type="xs:string" minOccurs="0"/>
          <xs:element name="acs_username" type="xs:string" minOccurs="0"/>
          <xs:element name="radius_username" type="xs:string" minOccurs="0"/>
          <xs:element name="nac_role" type="xs:string" minOccurs="0"/>
          <xs:element name="nac_username" type="xs:string" minOccurs="0"/>
          <xs:element name="nac_posture_token" type="xs:string" minOccurs="0"/>
          <xs:element name="nac_radius_is_user_auth" type="xs:string" minOccurs="0"/>
          <xs:element name="selected_posture_server" type="xs:string" minOccurs="0"/>
          <xs:element name="selected_identity_store" type="xs:string" minOccurs="0"/>
          <xs:element name="authentication_identity_store" type="xs:string"

minOccurs="0"/>
          <xs:element name="azn_exp_pol_matched_rule" type="xs:string" minOccurs="0"/>
          <xs:element name="ext_pol_server_matched_rule" type="xs:string" minOccurs="0"/>
          <xs:element name="grp_mapping_pol_matched_rule" type="xs:string" minOccurs="0"/>
          <xs:element name="identity_policy_matched_rule" type="xs:string" minOccurs="0"/>
          <xs:element name="nas_port_type" type="xs:string" minOccurs="0"/>
          <xs:element name="query_identity_stores" type="xs:string" minOccurs="0"/>
          <xs:element name="sel_exp_azn_profiles" type="xs:string" minOccurs="0"/>
          <xs:element name="selected_query_identity_stores" type="xs:string"
minOccurs="0"/>
          <xs:element name="tunnel_details" type="xs:string" minOccurs="0"/>
          <xs:element name="cisco_h323_attributes" type="xs:string" minOccurs="0"/>
          <xs:element name="cisco_ssg_attributes" type="xs:string" minOccurs="0"/>
          <xs:element name="other_attributes" type="xs:string" minOccurs="0"/>
          <xs:element name="response_time" type="xs:long" minOccurs="0"/>
          <xs:element name="nad_failure" type="xs:anyType" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:complexType name="restAuthStatus">
    <xs:sequence>
      <xs:element name="passed" type="xs:anyType" minOccurs="0"/>
      <xs:element name="failed" type="xs:anyType" minOccurs="0"/>
```

```
        <xs:element name="user_name" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_ip_address" type="xs:string" minOccurs="0"/>
        <xs:element name="failure_reason" type="xs:string" minOccurs="0"/>
        <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_port" type="xs:string" minOccurs="0"/>
        <xs:element name="identity_group" type="xs:string" minOccurs="0"/>
        <xs:element name="network_device_name" type="xs:string" minOccurs="0"/>
        <xs:element name="acs_server" type="xs:string" minOccurs="0"/>
        <xs:element name="eap_authentication" type="xs:string" minOccurs="0"/>
        <xs:element name="framed_ip_address" type="xs:string" minOccurs="0"/>
        <xs:element name="network_device_groups" type="xs:string" minOccurs="0"/>
        <xs:element name="access_service" type="xs:string" minOccurs="0"/>
        <xs:element name="acs_timestamp" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="authentication_method" type="xs:string" minOccurs="0"/>
        <xs:element name="execution_steps" type="xs:string" minOccurs="0"/>
        <xs:element name="radius_response" type="xs:string" minOccurs="0"/>
        <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_identifier" type="xs:string" minOccurs="0"/>
        <xs:element name="nas_port_id" type="xs:string" minOccurs="0"/>
        <xs:element name="nac_policy_compliance" type="xs:string" minOccurs="0"/>
        <xs:element name="selected_azn_profiles" type="xs:string" minOccurs="0"/>
        <xs:element name="service_type" type="xs:string" minOccurs="0"/>
        <xs:element name="eap_tunnel" type="xs:string" minOccurs="0"/>
        <xs:element name="message_code" type="xs:string" minOccurs="0"/>
        <xs:element name="destination_ip_address" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 AuthStatus API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

如果未能成功登录，请点击 Login 页面上的 **Problem logging in?** 链接，然后按照步骤 2 中的说明操作。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>/MACAddress/<macaddress>/<seconds>/<numberofrecordspermacaddress>/All)，在目标节点的 URL 地址字段输入 AuthStatus API 调用：

```
https://acme123/admin/API/mnt/AuthStatus/MACAddress/00:50:56:10:13:02/120/100/All
```

注意　REST API 调用区分大小写。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

AuthStatus API 调用返回的数据示例

以下示例展示了在目标思科监控 ISE 节点上发起 AuthStatus API 调用时返回的数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<authStatusOutputList>
-
<authStatusList key="00:0C:29:46:F3:B8"><authStatusElements>
-
<passed xsi:type="xs:boolean">true</passed>
<failed xsi:type="xs:boolean">false</failed>
<user_name>suser77</user_name>
<nas_ip_address>10.77.152.209</nas_ip_address>
<calling_station_id>00:0C:29:46:F3:B8</calling_station_id>
<identity_group>User Identity Groups:Guest</identity_group>
<acs_server>guest-240</acs_server>
<acs_timestamp>2012-10-05T10:50:56.515Z</acs_timestamp>
<execution_steps>5231</execution_steps>
<message_code>5231</message_code>
<id>1349422277270561</id>
<acsview_timestamp>2012-10-05T10:50:56.517Z</acsview_timestamp>
<identity_store>Internal Users</identity_store>
<response_time>146</response_time>
<other_attributes>ConfigVersionId=81,EndPointMACAddress=00-0C-29-46-F3-B8,PortalName=Defau
ltGuestPortal,
CPMSessionID=0A4D98D1000001F26F0C04D9,CiscoAVPair=</other_attributes>
</authStatusElements>
-
<authStatusElements>
<passed xsi:type="xs:boolean">true</passed>
<failed xsi:type="xs:boolean">false</failed>
<user_name>00:0C:29:46:F3:B8</user_name>
<nas_ip_address>10.77.152.209</nas_ip_address>
<calling_station_id>00:0C:29:46:F3:B8</calling_station_id>
<identity_group>Guest_IDG</identity_group>
<network_device_name>switch</network_device_name>
<acs_server>guest-240</acs_server>
<authentication_method>mab</authentication_method>
<authentication_protocol>Lookup</authentication_protocol>
<acs_timestamp>2012-10-05T10:49:47.915Z</acs_timestamp>
<execution_steps>11001,11017,11027,15049,15008,15048,15048,15004,15041,15006,15013,24209,2
421
1,22037,15036,15048,15004,15016,11022,11002</execution_steps>
<response>{UserName
=00:0C:29:46:F3:B8; User-Name=00-0C-29-46-F3-B8;
State=ReauthSession:0A4D98D1000001F26F0C04D9;
Class=CACS:0A4D98D1000001F26F0C04D9:guest-240/138796808/76;
Termination-Action=RADIUS-Request; Tunnel-Type=(tag=1) VLAN;
Tunnel-Medium-Type=(tag=1) 802; Tunnel-Private-Group-ID=(tag=1) 2;
cisco-av-pair=url-redirect-acl=ACL-WEBAUTH-REDIRECT;
cisco-av-pair=url-redirect=https://guest-240.cisco.com:8443/guestportal/gateway?
sessionId=0A4D98D1000001F26F0C04D9&action=cwa;
cisco-av-pair=ACS:CiscoSecure-Defined-ACL=#ACSACL#-IP-pre-posture-506e980a;
cisco-av-pair=profile-name=WindowsXP-Workstation;}</response
><audit_session_id>0A4D98D1000001F26F0C04D9</audit_session_id><nas_po
rt_id>GigabitEthernet1/0/17</nas_port_id><posture_status>Pending</posture_status>
<selected_azn_profiles>CWA_Redirect</selected_azn_profiles>
<service_type>Call Check</service_type>
<message_code>5200</message_code>
<nac_policy_compliance>Pending</nac_policy_compliance>
```

```
<id>1349422277270556</id>
<acsview_timestamp>2012-10-05T10:49:47.915Z</acsview_timestamp>
<identity_store>Internal Endpoints</identity_store>
<response_time>13</response_time>
<other_attributes>ConfigVersionId=81,DestinationPort=1812,Protocol=Radius,AuthorizationPol
icyMatchedRule=CWA_Redirect,
NAS-Port=50117,Framed-MTU=1500,NAS-Port-Type=Ethernet,EAP-Key-N
ame=,cisco-nas-port=GigabitEthernet1/0/17,AcsSessionID=guest-240/138796808/76,Us
eCase=Host Lookup,SelectedAuthenticationIdentityStores=Internal
Endpoints,ServiceSelectionMatchedRule=MAB,IdentityPolicyMatchedRule=Default,CPMS
essionID=0A4D98D1000001F26F0C04D9,EndPointMACAddress=00-0C-29-46-F3-B8,EndPointM
atchedProfile=WindowsXP-Workstation,ISEPolicySetName=Default,HostIdentityGroup=E
ndpoint Identity Groups:Guest_IDG,Device Type=Device Type#All Device
Types,Location=Location#All Locations,Device IP
Address=10.77.152.209,Called-Station-ID=00:24:F7:73:9A:91,CiscoAVPair=audit-sess
ion-id=0A4D98D1000001F26F0C04D9</other_attributes>
-
</authStatusElements>
-
</authStatusList>
-
</authStatusOutputList>
```

记帐状态 API 调用

您可以使用 AcctStatus API 调用检索目标节点上的最新设备和会话记帐状态。本节提供了架构文件输出示例、通过发起此 API 调用请求最新设备和会话信息的程序，以及发起此 API 调用后返回的数据示例。通过 AcctStatus API 调用，您可以配置与时间相关的参数：

- 持续时间 - 定义尝试搜索和检索与指定 MAC 地址关联的最新记帐设备记录所持续的秒数。用户可配置的有效值范围是 1 至 432000 秒（5 天）。例如，
  - 如果输入 2400 秒（40 分钟），则表示您想获得指定 MAC 地址在过去 40 分钟内可用的最新记帐设备记录。
  - 如果输入 0 秒，则指定默认持续时间 15 分钟（900 秒）。这表示您想获得指定 MAC 地址在此时间段内可用的最新记帐设备记录。

AcctList API 调用在 API 输出中提供以下记帐状态数据字段（请参阅表 3-3）：

表 3-3　　记帐状态数据字段

数据字段	说明
MAC 地址	客户端的 MAC 地址
审核会话 ID	身份验证会话 ID
传入的数据包数	接收的数据包总数
传出的数据包数	传输的数据包总数
收到的字节数	接收的总字节数
外发的字节数	传输的总字节数
会话时间	当前会话的持续时间

AcctStatus API 输出架构

此示例架构文件是将 AcctStatus API 调用发送到目标思科监控 ISE 节点上的指定会话后，此调用所返回的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="acctStatusOutputList" type="restAcctStatusOutputList"/>

  <xs:complexType name="restAcctStatusOutputList">
    <xs:sequence>
      <xs:element name="acctStatusList" type="restAcctStatusList" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="restAcctStatusList">
    <xs:sequence>
      <xs:element name="acctStatusElements" type="restAcctStatus" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="macAddress" type="xs:string"/>
    <xs:attribute name="username" type="xs:string"/>
  </xs:complexType>

  <xs:complexType name="restAcctStatus">
    <xs:sequence>
      <xs:element name="calling_station_id" type="xs:string" minOccurs="0"/>
      <xs:element name="audit_session_id" type="xs:string" minOccurs="0"/>
      <xs:element name="paks_in" type="xs:long" minOccurs="0"/>
      <xs:element name="paks_out" type="xs:long" minOccurs="0"/>
      <xs:element name="bytes_in" type="xs:long" minOccurs="0"/>
      <xs:element name="bytes_out" type="xs:long" minOccurs="0"/>
      <xs:element name="session_time" type="xs:long" minOccurs="0"/>
      <xs:element name="username" type="xs:string" minOccurs="0"/>
  <xs:element name="server" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

发起 AcctStatus API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

如果未能成功登录，请点击 Login 页面上的 **Problem logging in?** 链接，然后按照步骤 2 中的说明操作。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/<specific-api-call>/MACAddress/<macaddress>/<durationofcurrenttime>)，在目标节点的 URL 地址字段输入 AcctStatus API 调用：

```
https://acme123/admin/API/mnt/AcctStatus/MACAddress/00:26:82:7B:D2:51/1200
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

AcctStatus API 调用返回的数据示例

以下示例展示了在目标思科监控 ISE 节点上发起 AcctStatus API 调用时返回的数据：

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<acctStatusOutputList>
-
<acctStatusList macAddress="00:25:9C:A3:7D:48">
-
<acctStatusElements>
<calling_station_id>00:25:9C:A3:7D:48</calling_station_id>
<audit_session_id>0acb6b0b0000000B4D0C0DBD</audit_session_id>
<paks_in>0</paks_in>
<paks_out>0</paks_out>
<bytes_in>0</bytes_in>
<bytes_out>0</bytes_out>
<session_time>240243</session_time>
<server>HAREESH-R6-1-PDP1</server>
</acctStatusElements>
</acctStatusList>
</acctStatusOutputList>
```

授权更改 REST API

本章提供有关使用此版本思科身份服务引擎所支持的以下各个授权更改 (CoA) REST API 的示例和说明。

简介

CoA API 调用提供向 Cisco ISE 部署中的指定思科监控 ISE 节点发送会话身份验证和会话断开连接命令的途径。

CoA 会话管理 API 调用

通过 CoA 会话管理 API 调用，您可以向 Cisco ISE 部署中的目标思科监控 ISE 节点上的指定会话发送重新进行身份验证和断开连接命令：

- 默认重新身份验证 (Reauth)
- 会话断开连接 (Disconnect)

会话重新身份验证 API 调用

会话重新身份验证 API 调用包括以下类型：

- REAUTH_TYPE_DEFAULT = 0
- REAUTH_TYPE_LAST = 1
- REAUTH_TYPE_RERUN = 2

Reauth API 输出架构

此示例架构文件是将 Reauth API 调用发送到目标思科监控 ISE 节点上的指定会话后，此调用所返回的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="remoteCoA" type="coAResult"/>
<xs:complexType name="coAResult">
    <xs:sequence>
      <xs:element name="results" type="xs:boolean" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="requestType" type="xs:string"/>
  </xs:complexType>
</xs:schema>
```

调用 Reauth API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址 >/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/CoA/<specific-api-call>/<macaddress>/<reauthtype>)，在目标节点的 URL 地址字段输入 Reauth API 调用：：

```
https://acme123/admin/API/mnt/CoA/Reauth/server12/00:26:82:7B:D2:51/1
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

Reauth API 调用返回的数据示例

以下示例展示的是在目标思科监控 ISE 节点上发起 Reauth API 调用时返回的数据： 调用此命令可能返回两种结果：

- True 表示命令成功执行。
- False 表示命令未执行（由于各种情况）。

```
This XML file does not appear to have any style information associated with it. The
document tree is shown below.

-
<remoteCoA requestType="reauth">
<results>true</results>
</remoteCoA>
```

会话断开连接 API 调用

会话断开连接 API 调用包括以下断开连接端口选项类型：

- DYNAMIC_AUTHZ_PORT_DEFAULT = 0
- DYNAMIC_AUTHZ_PORT_BOUNCE = 1
- DYNAMIC_AUTHZ_PORT_SHUTDOWN = 2

Disconnect API 输出架构

此示例架构文件是将 Disconnect API 调用发送到目标思科监控 ISE 节点上的指定会话后，此调用所返回的输出：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="remoteCoA" type="coAResult"/>

  <xs:complexType name="coAResult">
    <xs:sequence>
      <xs:element name="results" type="xs:boolean" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="requestType" type="xs:string"/>
  </xs:complexType>
</xs:schema>
```

发起 Disconnect API 调用

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，*https://<ISE 主机名或 IP 地址>/admin/*）。

步骤 2　输入在 Cisco ISE 初始设置过程中指定和配置的用户名及密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

例如，如果您最初使用主机名 acme123 登录思科监控 ISE 节点，对于此节点，系统将显示以下 URL 地址字段：

```
https://acme123/admin/LoginAction.do#pageId=com_cisco_xmp_web_page_tmpdash
```

步骤 4　通过将“/admin/”部分替换为 API 调用部分 (/admin/API/mnt/CoA/<Disconnect>/<serverhostname>/<macaddress>/<portoptiontype>/<nasipaddress>/<destinationipaddress>），在目标节点的 URL 地址字段输入 Disconnect API 调用：

```
https://acme123/admin/API/mnt/CoA/Disconnect/server12/00:26:82:7B:D2:51/2/10.10.10.10/192.168.1.1
```

注意　由于这些调用区分大小写，所以在目标节点的 URL 地址字段输入每个 API 调用时请务必仔细。在 API 调用约定中使用“mnt”表示思科监控 ISE 节点。

步骤 5　按 **Enter** 将发出 API 调用。

相关主题

- 验证监控节点，第 1-2 页

Disconnect API 调用返回的数据示例

以下示例展示的是在目标思科监控 ISE 节点上发起 Disconnect API 调用时返回的数据： 调用此命令可能返回两种结果：

- True 表示命令成功执行。
- False 表示命令未执行（由于各种情况）。

```
This XML file does not appear to have any style information associated with it. The document tree is shown below.

-
<remoteCoA requestType="reauth">
<results>true</results>
</remoteCoA>
```

第 2 部分

Cisco ISE External RESTful Services APIs

第 5 章

ERS API 简介

概述

本章提供有关使用受支持的外部 RESTful 服务 API 和相关 API 调用的指南和示例。通过这些调用，您可以对 Cisco ISE 资源执行 CRUD（创建、读取、更新、删除）操作。

支持的 Cisco ISE 资源

通过 Cisco ISE 外部 RESTful 服务 API，您可以对以下类型的 ISE 资源执行操作：

- 终端设备
- 终端身份组
- 访客用户
- 身份组

- 内部用户
- 门户
- 分析器策略
- 网络设备
- 网络设备组
- 安全组

第 7 章，“外部 RESTful 服务 API 操作”中提供完整的请求和响应示例。

外部 RESTful 服务 API 身份验证和授权

外部 RESTful 服务 API 基于 HTTPS 协议和 REST 方法，并使用端口 9060。

外部 RESTful 服务 API 支持基本身份验证。身份验证凭证经过加密并且属于请求头的一部分。

ISE 管理员必须向用户分配特殊权限，用户才能使用外部 RESTful 服务 API 执行操作。ISE 管理员可以分配以下两种角色，以便使用外部 RESTful 服务 API 执行操作：

- 外部 RESTful 服务管理员 - 具有所有 ERS API 的完全访问权限（GET、POST、DELETE、PUT）。
- 外部 RESTful 服务操作员 - 具有只读访问权限（仅可执行 GET 请求）。

如果用户不具有所需的权限，但仍尝试使用外部 RESTful 服务 API 执行操作，则会收到错误响应。

相关主题

- 创建一个新的 Cisco ISE 管理员
- 发起人身份验证和授权，第 6-1 页

从 GUI 启用外部 RESTful 服务 API

要让为 Cisco ISE REST API 开发的应用能够访问 Cisco ISE，您必须启用 Cisco ISE REST API。Cisco REST API 使用 HTTPS 端口 9060，默认情况下，此端口处于关闭状态。如果未在 Cisco ISE 管理服务器上启用 Cisco ISE REST API，对于任何访客 REST API 请求，客户端应用都会从服务器收到超时错误。

所有类型的外部 RESTful 服务请求均仅对主 ISE 节点有效。辅助节点具有读取访问权限（GET 请求）。

操作步骤

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，`https://<ISE 主机名或 IP 地址>/admin/`）。

步骤 2　输入您的用户名和密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

步骤 4　选择 **Administration > Settings > ERS Settings**。

步骤 5　为 Primary Administration Node 选择 **Enable ERS for Read/Write**。

步骤 6　如果存在辅助节点，请为 All Other Nodes 选择 **Enable ERS for Read**。

步骤 7　点击**提交**。

注意　所有 REST 操作均要经过审核，并且在系统日志中记录日志。

相关主题

- 使用外部 RESTful 服务 API 调用的必备条件，第 7-1 页

外部 RESTful 服务 API 状态

您可以在 GUI 的 **Administration > Settings > ERS Settings** 页面上验证是否为主节点和辅助节点启用了外部 RESTful 服务 API。

默认情况下，外部 RESTful 服务 API 处于未启用状态。如果您在启用外部 RESTful 服务 API 调用之前尝试调用这些 API，将会收到错误响应。

外部 RESTful 服务 API 具有调试日志记录类别，您可以从 Cisco ISE GUI 的调试日志记录页面启用此类别。

注意　有关详细信息，请参阅*思科身份服务引擎管理员指南，版本 1.4* 的调试日志配置选项部分。

数据验证

发送到服务器的 CRUD 数据使用与 Cisco ISE 用于 GUI 的相同验证规则进行验证。所有验证在验证层集中进行。所要发布的所有 XML 数据都会根据架构进行验证。

系统会执行以下两种类型的验证：数据验证和结构验证。数据验证会验证数据是否符合 ISE 要求，例如，强制字段、字段长度及类型等。而结构验证会根据架构进行验证，例如，字段顺序、名称等。

命名空间

您必须在资源名称和 URI 中遵守严格的命名空间要求，如下所述：

- 内部用户、终端、终端组和身份组的身份
- SGA 的 SGT
- 所有其他对象资源的外部 RESTful 服务，如显示在响应消息中的搜索结果（客户端确实需要在请求中发送 ERS 命名空间）

Accept/Content-Type 标头必须包含以下命名空间：

```
application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major
version>.<minor version>+xml
```

例如：`application/vnd.com.cisco.ise.identity.internaluser.1.0+xml`

请求 XML 应包含如下命名空间定义：

```
identity.ers.ise.cisco.com

sga.ers.ise.cisco.com
```

例如：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<ns:endpoint xmlns:ns="identity.ers.ise.cisco.com" id="id">
```

```
<group>Profiled</group>
…
</ns:endpoint>
```

外部 RESTful 服务 SDK

您可以使用外部 RESTful 服务 SDK 开始构建自己的工具。您可以从以下 URL 访问外部 RESTful 服务 SDK：`https://<ISE-ADMIN-NODE>:9060/ers/sdk`。

只有外部 RESTful 服务管理员用户可以访问外部 RESTful 服务 SDK。SDK 包括以下组件：

- 快速参考 API 文档
- 所有可用 API 操作的列表
- 可下载的架构文件
- 可下载的 Java 示例应用
- 采用 curl 脚本格式的使用案例
- 使用 Chrome Postman 的说明

外部 RESTful 服务架构文件

外部 RESTful 服务 SDK 随附三个 XSD 架构文件，这些架构文件介绍有关 ISE ERS 界面所支持对象的结构。

三个 XSD 文件为：

- ers.xsd
- identity.xsd
- sga.xsd
- network.xsd

您可以结合使用架构和可用的工具（如 JAXB）来生成架构类。

您可以开发 HTTP 客户端或使用任何第三方 HTTP 客户端代码，并将其与通过 XSD 文件生成的架构类集成。

注意　内容中发送的 XML 根据架构进行验证，因此，XML 中的字段顺序和语法应该与其在架构中的显示相同。否则，您将会收到错误请求状态代码。

下载架构文件

操作步骤

步骤 1　在浏览器的地址栏输入以下 URL 以登录外部 RESTful 服务 SDK 页面：
`https://<ISE-ADMIN-NODE>:9060/ers/sdk`

步骤 2　输入外部 RESTful 服务管理员对应的用户名和密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

步骤 4　在 **Downloads** 类别中，点击 **Schema Jar (ers-schema-1.4-iteration-01-SNAPSHOT.jar)**。

步骤 5　将文件保存到您的本地计算机。

相关主题

- 外部 RESTful 服务 API 身份验证和授权，第 5-2 页

外部 RESTful 服务请求和响应

本节提供有关请求头和响应头的信息以及 ERS 返回的状态代码。

外部 RESTful 服务请求头

表 5-1　ERS 请求头

标头	支持的值	使用说明	必填
接受	访客 REST API 资源介质类型	向服务器指示此客户端愿意接受的介质类型，包括资源版本。	是，用于 GET/GET ALL/DELETE/GET VERSION 操作（这些操作不包含消息正文）。
Authorization	“Basic”加上用户名和密码（根据 RFC 2617）	识别提交此请求的授权用户。	是，用于所有请求。
Content-Length	访客 REST API 资源介质类型	向服务器指示此客户端愿意接受的介质类型，包括资源版本。	是，用于包含消息正文的请求。
Content-Type	描述请求消息正文的介质类型	描述请求消息正文的表示和语法。	是，用于包含消息正文的请求。

外部 RESTful 服务响应头

表 5-2　ERS 强制响应头

标头	支持的值	使用说明	必填
Content-Length	响应消息正文的长度 （以字节为单位）。	描述消息正文的大小。	是，用于包含消息正文的响应。
Content-Type	描述响应消息正文的介质类型。	描述响应消息正文的表示和语法。	是，用于包含消息正文的响应。
位置	新创建资源的规范 URI。	返回可用于请求新创建资源表示的新 URI。	是，用于新建可通过 URI 访问的服务器端资源的请求的响应。

常见外部 RESTful 服务 HTTP 状态代码

表 5-3　　ERS 返回的 HTTP 响应代码的说明

HTTP 状态	描述
200 OK	请求已成功完成。如果此请求创建了可使用 URI 表示地址的新资源，并且返回了包含新资源表示的响应正文，则系统将返回 200 状态，并带有包含新建资源的规范 URI 的位置标头。
201 Created	创建新资源的请求已完成，并且未返回包含新资源表示的响应正文。系统还应返回包含新建资源的规范 URI 的位置标头。
202 Accepted	请求已被接受并进行处理，但是处理尚未完成。根据 HTTP/1.1 规范，返回的实体（如有）应包含请求的当前状态的指示，以及状态监视器的指针或请求有望满足的预期时间。
204 No Content	服务器已执行请求，但是不需要返回响应消息正文。
400 Bad Request	由于请求包含缺失信息或无效信息（如输入字段验证错误、缺少所需的值等），无法处理请求。
401 Unauthorized	此请求中包含的身份验证凭证缺失或无效。
403 Forbidden	服务器已识别出您的凭证，但是您不具有执行此请求的授权。
404 Not Found	请求中指定了不存在资源的 URI。
405 Method Not Allowed	请求 URI 不支持此请求中指定的 HTTP 谓词（DELETE、GET、HEAD、POST、PUT）。
406 Not Acceptable	此请求标识的资源无法生成对应于请求 Accept 标头中的介质类型的表示。
409 Conflict	创建或更新请求无法完成，因为会导致服务器所支持资源的当前状态产生冲突（例如，尝试使用已经分配给某个现有资源的唯一标识符创建新资源）。
415 Unsupported Media Type	服务器不支持 Accept 标头中指定的介质类型。如果服务器不再支持客户端资源版本，通常会返回此响应。
429 Too many requests	并发 ERS 请求数过多。
500 Internal Server Error	服务器遇到阻止其执行请求的意外情况。
501 Not Implemented	服务器（当前）不支持执行请求所需的功能。
服务器（当前）不支持执行请求所需的功能。	由于服务器临时过载或处于维护状态，服务器当前无法处理请求。

注意　除了响应头中返回的状态代码，根据请求的性质，每个请求还可能会包含其他 xml 内容。

使用外部 RESTful 服务 API 进行版本控制

外部 RESTful 服务 API 提供与早期 Cisco ISE 版本的向后兼容。外部 RESTful 服务 API 具有用于 API 版本管理的版本控制机制。所有非访客资源为版本 1.0，无需向后兼容。

每个 RESTful 资源都具有一个模型版本 (major.minor)。版本必须包含在请求标头内，并采用如下语法：

```
application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major version>.<minor version>+xml
```

例如，要获取内部用户资源版本 1.0，需要通过以下请求：

```
DELETE https://<ISE-ADMIN-NODE>:9060/ers/config/internaluser/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
```

对请求进行身份验证和授权后，系统将执行版本匹配检查，并显示下表中提到的匹配结果：

版本匹配	结果
未发送版本	服务器返回状态 415“Unsupported Media Type”。
客户端版本等于服务器版本	服务器继续处理请求。
客户端次要版本不等于服务器次要版本	服务器添加描述版本差异的响应警告消息，并继续处理请求。
客户端和服务器主要版本不匹配	服务器返回状态 415 以及相应的错误消息。

注意　每种资源都具有检索服务器支持的版本列表的 API。

搜索和过滤

所有过滤和搜索操作都通过使用过滤完成。

您可通过向资源 URI 发送 GET 请求搜索资源。默认情况下，结果为第一页（页面索引 = 0），默认大小为 20。通过向 URI 添加以下部分介绍的过滤器、排序和分页参数，客户端可以控制此搜索。

从分页、过滤器或排序请求生成的资源绑定在 `<resources>` 集合中，此集合包含每个资源的名称、ID、说明及其完整表示的链接。这使客户端能够轻松地向下展开到资源。

外部 RESTful 服务 API 的过滤参数

您可以通过过滤器查询字符串参数执行简单的过滤操作。您可以发送多个过滤器。默认情况下，所有过滤器条件通用的逻辑运算符是 AND。您可以通过使用“filtertype=or”查询字符串参数更改此默认运算符。

每个资源数据模型说明应指定属性是否为已过滤字段。

例如，要获取名字开头为“a”且属于身份组“Finance”的内部用户，应通过以下请求：

```
GET
/ers/config/internaluser/?page=0&size=20&sortacs=name&filter=name.STARTSW.a&filter=identityGroup.EQ.Finance
```

下表显示可用的过滤器运算符：

参数	描述
EQ	等于
GT	大于
LT	小于
STARTSW	开头为
ENDSW	结尾为
CONTAINS	包含

下表显示每种资源的可过滤属性列表：

资源	可过滤属性
终端	mac、 portalUser、 profile、 profileId、 staticGroupAssignment、 staticProfileAssignment
内部用户	名称
终端身份组	名称
身份组	名称
SGT	名称

外部 RESTful 服务 API 的分页参数

所有外部 RESTful 服务搜索结果均经过分页。分页参数在使用查询参数的 URI 中传递。

例如，要获取按“name”字段以升序排列的前 20 条内部用户记录，需通过以下请求：

```
GET /ers/config/internaluser?page=0&size=20&sortasc=name
```

下表显示可用的分页参数：

参数	描述	默认值
page	页面起始索引	0
size	页面大小	20（最大值 = 100）
sortasc	按升序对字段排序，从一组可用的排序字段中选择字段。（对于字母字符，按 A 到 Z 排序。）	名称
sortdsc	按降序对字段排序，从一组可用的排序字段中选择字段。（对于字母字符，按 Z 到 A 排序。）	名称

外部 RESTful 服务系统流程

常见的外部 RESTful 服务流程由从客户端发送的 HTTPS 请求和来自服务器的 HTTPS 响应组成。流程因请求类型、URI、请求头、响应头和响应内容而异。

图 5-1　　ERS 成功流程

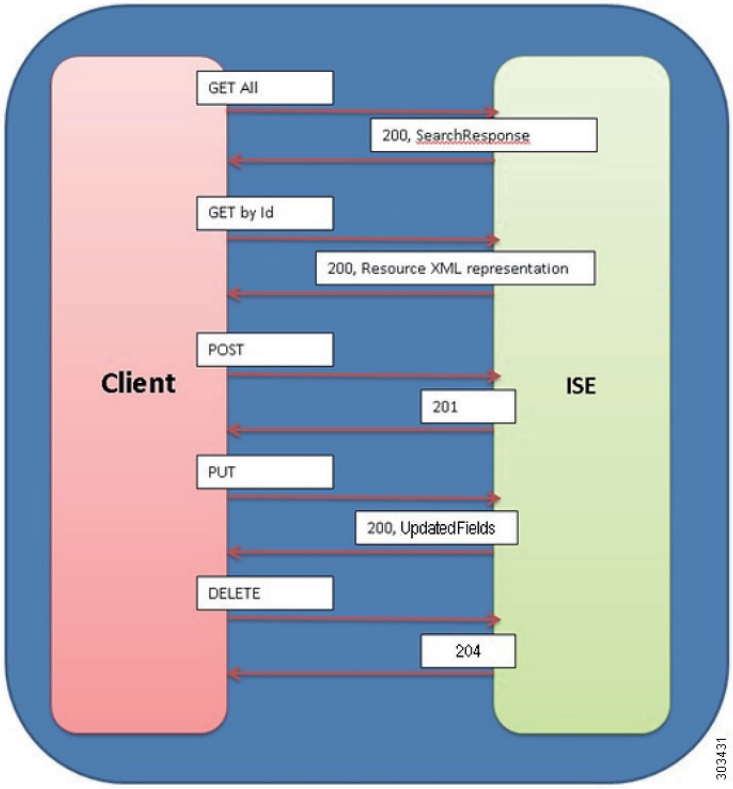
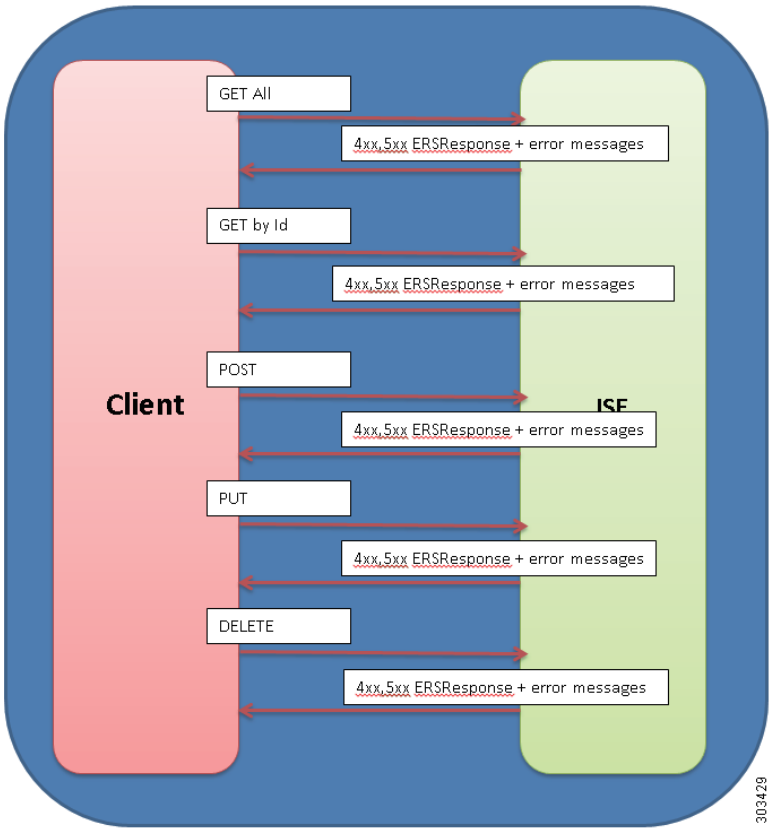


图 5-2　　ERS 失败流程



超链接

在 XML 表示中使用超链接是作为应用状态引擎的超媒体 (HATEOS) 的其中一个最强特性。超链接最常用于传达到资源在给定 URI 中具有表示的客户端。

外部 RESTful 服务链接符合以下命名空间所声明链接的 Atom 定义：

http://www.w3.org/2005/Atom namespace

下表显示外部 RESTful 服务的强制链接属性：

属性	说明
Href	此属性包含链接的 URI。
Rel	此属性的最初含义为“关系”，表示以下链接类型： • “self”- 刷新当前表示的链接 • “next”- 在获取下一个页面的集合中 • “info”- 获取资源的更多信息
类型	提示服务器可能为链接的 URI 返回的表示的介质类型，通常为“application/xml”。

搜索结果中的链接示例

以下示例显示搜索结果中的链接：

```
<ns2:searchResult xmlns:ns2="ers.ise.cisco.com"   total="1163">
<link rel="self"
href="http://cisco.com/ers/config/internaluser?page=0&size=20"
type="application/xml"/>
<link rel="next" href="http://cisco.com/ers/config/internaluser?page=20&size=20"
type="application/xml"/>
<resources>
<link rel="john doe" href="http://cisco.com/ers/config/internaluser/333"
type="application/xml"/>
<link rel="jeff smit" href="http://cisco.com/ers/config/internaluser/444"
type="application/xml"/>
.
.
.
</resources>
</ns2:searchResult>
```

批量操作

通过批量操作请求，您可以在单个请求中发送最多 500 项操作（或每个 ID 5000 项操作）。您可以对所有资源运行创建、更新和删除操作，也可以运行某些特定于资源的操作，如注册终端。

请求中的所有操作必须具有相同类型，也就是说您不能发送混合资源请求。

每种资源都有自己的事务，并且由于是多线程执行，因此不保证事务的顺序。

Cisco ISE 服务器解析请求并验证其结构。如果请求有效且未进行其他批量操作，系统将开始执行，且服务器将返回状态代码 `202 (ACCEPTED)` 并在 LOCATION 响应头中显示唯一的批量标识符。通过此 ID，您稍后可以使用获取批量状态操作跟踪批量状态。您可以获取操作开始之后至少 2 小时的状态报告。

如果对资源执行操作时出现故障，故障将记录在状态报告中，系统会继续执行到下一个资源。

一次只允许执行一个批量操作。如果在另一个批量操作仍在执行时发布批量操作请求，服务器将返回响应状态 `503 (Service Unavailable)`，并包含要求客户端稍后重试的消息。

访客 REST API

用于访客用户资源的 API

思科访客 API 是基于 REST（表述性状态转移）的操作集合，提供对思科访客用户进行经过身份验证的安全访问及管理功能。利用此 API，您可以创建、读取、更新、删除和搜索访客用户。

当调用此 API 时，您将作为使用 Cisco ISE 发起人门户的发起人，调用 API 来管理访客用户。要使用此 API，您必须先在 ISE 中启用相应的访问权限，然后设置发起人身份验证。

适用于访客用户的外部 RESTful 服务 API，第 7-24 页中提供完整的请求和响应示例。

发起人身份验证和授权

发起人是特殊类型的 Cisco ISE 用户，可以使用发起人门户创建并管理访客用户。访客 REST API 具有与发起人门户相同的功能。访客 REST API 的身份验证与发起人进行身份验证的过程类似。在发起人组的策略中指定的权限适用于访客 REST API。

注意 使用访客 REST API 的发起人不能属于 ERS-ADMIN 角色。

有关发起人和发起人组的详细信息，请参阅《Cisco ISE 用户指南》。

准备工作

与其他 ISE 用户相似，Cisco ISE 通过本地数据库或者外部轻量级目录访问协议 (LDAP) 或 Microsoft Active Directory 身份库对发起人进行身份验证。如果您不打算使用外部身份库，则必须在 Cisco ISE 中创建用户 (**Administration > Identity Management > Identities > Users**)。

操作步骤

步骤 1　在浏览器的地址栏内输入 Cisco ISE URL（例如，`https://<ISE 主机名或 IP 地址>/admin/`）。

步骤 2　输入您的用户名和密码（区分大小写）。

步骤 3　点击 **Login** 或按 **Enter**。

步骤 4　将用户分配到适当的身份组。

a. 选择 **Administration > Identity Management > Groups > Identity Groups**。

b. 创建新组或编辑现有组。Cisco ISE 包括以下默认发起人用户身份组：

- SponsorAllAccount
- SponsorGroupAccounts
- SponsorOwnAccounts

c. 将用户添加到成员列表中。

d. 点击**保存**。

步骤 5　将用户身份组添加到发起人组中。

a. 选择 **Guest Access > Configure > Sponsor Groups**。

b. 创建新发起人组或编辑现有发起人组。

c. 点击 **Members**。

d. 将用户的身份组添加到发起人组成员列表中，然后点击 **OK**。

e. 添加用户可以发起的访客类型和位置。

f. 选中 **Access ISE guest accounts via the programatic interface (REST API)** 复选框。

g. 点击**保存**。

步骤 6　确保 Sponsor_Portal_Sequence 可以访问用户的身份源。

a. 选择 **Administration > Identity Management > Identity Source Sequences > Sponsor_Portal_Sequence**。

b. 如果没有为 Authentication Search List 选择用户的身份源，请选择用户的身份源。

c. 点击**保存**。

相关主题

有关发起人和发起人组的详细信息，请参阅《Cisco ISE 用户指南》。

访客 REST API 请求

对 API 执行的请求具有以下特征：

- 用户请求通过 HTTPS 发送到 Cisco ISE 服务器，然后返回响应。
- 访问所有 API 的 URI 是主节点的域名 `https://<ISE-ADMIN-NODE>:9060/ers/config`
- API 请求区分大小写，应按照本手册所示进行输入。
- 每个访客请求需要将 Content Type 设置为：

  `application/vnd.com.cisco.ise.identity.guestuser.2.0+xml`
- 每个访客请求需要将 Accept 设置为：

  `application/vnd.com.cisco.ise.identity.guestuser.2.0+xml`
- 每个访客批量操作请求需要将 Content Type 设置为：

  `application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml`
- 每个访客批量操作请求需要将 Accept 设置为：

  `application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml`
- 您还可以包括可选的 `Accept-Encoding:gzip` 标头，用于压缩响应负载。

请求结构

下表列出可与思科访客 REST API 结合使用的请求操作类型：

表 6-1　　请求方法类型

请求类型	运营
GET	获取所有资源 （搜索）、根据资源 ID 获取某项资源，以及获取资源版本信息。
POST	创建新访客用户。
PUT	更新访客用户。
DELETE	删除访客用户

下表列出请求的强制性标头：

表 6-2　　强制性请求标头

标头	价值观	描述
ACCEPT	访客 REST API 资源介质类型	向服务器指示此客户端愿意接受的介质类型，包括资源版本。
AUTHORIZATION	“Basic”加上用户名和密码（根据 RFC 2617）	识别提交此请求的授权用户。
CONTENT-TYPE	描述请求消息正文的介质类型	描述请求消息正文的表示和语法。

相关主题

Content Type 和 Accept 标头，第 7-24 页

请求内容

以下 XML 内容显示访客用户帐户的结构，包括自定义字段：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:guestuser name="guestUser" id="123456789" description="ERS Example user "
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <customFields>
        <entry>
            <key>some key</key>
            <value>some value</value>
        </entry>
        <entry>
            <key>another key</key>
            <value>and its value</value>
        </entry>
    </customFields>
    <guestInfo>
        <emailAddress>email@some.uri.com</emailAddress>
        <enabled>true</enabled>
        <password>asdlkj324ew</password>
        <phoneNumber>3211239034</phoneNumber>
        <smsServiceProvider>GLobal Default</smsServiceProvider>
        <userName>DS3ewdsa34wWE</userName>
    </guestInfo>
    <guestType>Contractor</guestType>
    <portalId>23423432523</portalId>
    <sponsorUserName>Mr Spons</sponsorUserName>
</ns3:guestuser>
```

以下请求示例显示有关使用 POST（创建）操作创建访客用户帐户所需的 XML 内容。

注意　创建访客用户帐户所需的字段无需与发起人门户中显示的必填字段对应。但是，如果发起人门户不提供创建访客用户帐户所需的信息，访客 REST API 会引发异常。

POST 请求示例

```
POST https://<ISE-Admin-Node>:9060/ers/config/guestuser/

Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Content-Type: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Authorization: Basic xxxxxxxxxxxxxxxx

<?xml version="1.0" encoding="UTF-8"?>
<ns2:guestuser xmlns:ns2="identity.ers.ise.cisco.com">
    <guestAccessInfo>
        <fromDate>08/06/2014 23:22</fromDate>
        <toDate>08/07/2014 23:22</toDate>
        <validDays>1</validDays>
    </guestAccessInfo>
    <guestInfo>
        <company>New Company</company>
        <emailAddress>john@example.com</emailAddress>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
        <notificationLanguage>English</notificationLanguage>
        <password>12345</password>
        <phoneNumber>9999998877</phoneNumber>
        <smsServiceProvider>Global Default</smsServiceProvider>
        <userName>autoguestuser1</userName>
```

```
    </guestInfo>
    <guestType>Daily</guestType>
    <personBeingVisited>sponsor</personBeingVisited>
    <portalId>portal101</portalId>
    <reasonForVisit>interview</reasonForVisit>
</ns2:guestuser>
```

注意　JSON 不用于访客 REST API。

批量执行

通过批量操作请求，您可以在单个请求中发送最多 500 项操作，或根据 ID 发送最多 5000 项操作。

请求中的所有操作必须具有相同类型，也就是说您不能发送混合资源请求。

每种资源都有自己的事务，并且由于是多线程执行，因此不保证事务的顺序。

Cisco ISE 服务器解析请求并验证其结构。如果请求有效且未进行其他批量操作，系统将开始执行，且服务器将返回状态代码 `202 (ACCEPTED)` 并在 LOCATION 响应头中显示唯一的批量标识符。通过此 ID，您稍后可以使用获取批量状态操作跟踪批量状态。您可以获取操作开始之后至少 2 小时的状态报告。

如果对资源执行操作时出现故障，故障将记录在状态报告中，系统会继续执行到下一个资源。

一次只允许执行一个批量操作。如果在另一个批量操作仍在执行时发布批量操作请求，服务器将返回响应状态 `503 (Service Unavailable)`，并包含要求客户端稍后重试的消息。

批量执行（包括获取批量状态）使用不同的标头：

- Content Type 为：`application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml`
- Accept 为：`application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml`

相关主题

- 启动访客用户的批量执行，第 7-39 页

访客 REST API 响应

每个请求随后都会收到服务器 HTTPS 响应，此响应包括标准标头和响应内容。

响应状态代码

响应包含一个 HTTP 状态代码。对于访客 REST API，这些代码可以是下列代码之一：

- 20x - 操作成功
- 4xx - 客户端错误
- 5xx - 服务器错误

下表提供有关状态代码的更多信息。

表 6-3　响应内容类型

请求类型	状态	响应负载
根据访客用户的 ID 获取访客用户	200	XML 表示的访客用户。
全部	4xx（客户端错误）	响应包含描述错误的消息，例如：版本不受支持或非法 URI。
全部	5xx（服务器错误）	响应包含描述错误的消息，例如，运行时异常 ...
创建 A 访客用户 [POST]	201	如果没有信息或警告，不会发回任何内容。新访客用户 ID 将在响应的“Location”标头中发送。如果还有其他信息或警告，将封装在响应中。
更新访客用户 [PUT]	200	返回更新字段的列表。
删除访客用户 [Delete]	204	无内容。

相关主题

响应错误消息，第 6-7 页

响应结构

下表列出响应的标头。

表 6-4　强制性响应标头

标头	价值观	说明
LOCATION	新建资源的 ID	仅适用于 POST - 获取有新资源 ID（以 URI 表示）
CONTENT-TYPE	描述响应消息正文的介质类型	描述响应消息正文的表示和语法

访客密码

创建访客时，ISE 自动生成密码。可以使用访客 REST API 通过调用 `resetpassword` 操作重置访客的密码。

您无法使用此 REST API 将访客的密码更改为指定字符串。

使用 GET 操作可检索访客用户的信息以及查看其密码。只要密码符合以下条件，就可以在 GET 操作的响应中显示 Cisco ISE 访客密码：

1. 由 ISE 自动生成。
2. 通过此 API 或发起人门户重置。

在有些访客流中，访客能够更改自己的密码。访客更改过的 Cisco ISE 访客密码在发起人门户中不可见，也无法通过 REST API 显示。

相关主题

- 重置访客用户帐户的密码，第 7-39 页
- 有关访客用户密码策略的更多详细信息，请参阅《Cisco ISE 管理员指南》。

响应错误消息

您在执行 POST 或 PUT 请求时，需要使用特定标头。如有遗漏，您会收到错误消息。

在 Cisco ISE UI 的以下位置可以找到包含详细消息的日志文件，您可以启用这些日志文件以更正操作：

Operations > Node > Debug Logs > guest.log

表 6-5　　错误代码

错误代码	说明	HTTP 状态
资源版本异常	服务器不支持请求内容中发送的资源版本。	415
资源介质类型异常	客户端在 ACCEPT 或 Content-Type 标头中发送的介质类型无效。	415
不受支持的资源异常	服务器不支持 URI 中列出的资源。	400
不受支持的方法异常	指定 URI 不支持请求方法类型。	400
查询字符串验证异常	搜索过滤器或分页参数无效。	400
架构验证异常	请求的 XML 内容不符合资源的 API 架构规则，例如包含 API 无法识别的字段或不包含必填字段。	400
应用资源验证异常	API 无法验证某些请求的内容，例如属性值中使用了无效字符。	400
未授权用户	发起人的用户名和密码无效。	401
内部异常	运行时发生的所有内部服务器意外错误。	500
转换异常	一些内部转换，应视为内部异常。	500
CRUD 操作异常	执行 CRUD 操作，应视为内部异常。	500

不支持的介质类型示例

以下示例展示的是客户端在 ACCEPT 标头中发送不受支持的介质类型时出现的响应。

响应

```
STATUS 415 Unsupported Media Type
Cache-Control: no-cache
Content-Length: 411
Content-Type: application/vnd.com.cisco.ise.ers.ersresponse.1.0+xml
Date: Wed, 11 Dec 2013 05:27:37 GMT
Expires: Wed, 31 Dec 2015 16:00:00 PST
Pragma: No-cache
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:ersResponse
    xmlns:ns2="ers.ise.cisco.com" operation="GET-getAll-guestuser">
    <link type="application/xml"
href="https://<ISE-Admin-Node>:9060/ers/config/guestuser/" rel="related"/>
    <messages>
        <ns2:message type="ERROR" code="Resource media type exception">
            <title>Wrong media type, check Accept request header.</title>
        </ns2:message>
    </messages>
</ns2:ersResponse>
```

版本

Cisco ISE 版本 1.4 的访客 REST API 是 2.0 版本。它可与未来版本兼容。

此 REST API 与早期 alpha 版本和 beta 版本的访客 REST API 不兼容。

每个 RESTful 资源都具有一个模型版本 (major.minor)。版本必须包含在请求标头内，并采用如下语法：

```
application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major version>.<minor
version>+xml
```

例如，要获取访客用户资源版本 2.0，需要通过以下请求：

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

对请求进行身份验证和授权后，系统将执行版本匹配检查，并具有下列匹配结果之一：

表 6-6　　版本匹配结果

版本匹配	结果
未发送版本	服务器返回状态 `415 "Unsupported Media Type"`
客户端版本等于服务器版本	服务器继续处理请求。
客户端次要版本不等于服务器次要版本	服务器添加描述版本差异的响应警告消息，并继续处理请求。
客户端和服务器主要版本不匹配	服务器返回状态 `415` 以及相应的错误消息。

搜索和过滤

所有过滤和搜索操作都通过使用过滤完成。

您可通过向资源 URI 发送 GET 请求搜索资源。默认情况下，结果为第一页（页面索引 = 0），默认大小为 20。通过向 URI 添加以下部分介绍的过滤器、排序和分页参数，客户端可以控制此搜索。

从分页、过滤器或排序请求生成的资源绑定在 `<resources>` 集合中，此集合包含每个资源的名称、ID、说明及其完整表示的链接。这使客户端能够轻松地向下展开到资源。

过滤参数

过滤功能可通过过滤器查询字符获取。过滤器的结构是字段运算符、参数和值三者的组合，以圆点分隔。例如，使用 `filter=name.STARTSW.g` 查找用户名以“g”开头的所有访客用户。

如果您在过滤器中使用多个参数，结果为对这些参数进行 AND 运算的结果。这意味着结果是与发送到 API 的所有参数匹配的用户。每个资源说明指定属性是否为已过滤字段。

注意　无效的过滤器值将生成状态 `400 (Bad Request)` 并显示相应的消息。

下表显示过滤器查询中可用的参数。

表 6-7　过滤参数

参数	描述
firstName	访客的名字
lastName	访客的姓氏
emailAddress	访客的邮件地址
userName	访客帐户用户名
creationTime	访客帐户的创建时间
toDate	访客帐户的到期日期
guestType	访客的类型
company	访客的公司
phoneNumber	访客的电话号码
groupTag	组
sponsor	访客的发起人
status	访客帐户的状态
名称	资源标识符

下表列出可以在过滤器查询中使用的操作。

表 6-8　可用的过滤器操作

参数	描述
EQ	等于
GT	大于
LT	小于
STARTSW	开头为
ENDSW	结尾为
CONTAINS	包含

过滤示例

使用 GET 请求获取名字的开头为“br”的访客用户

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/?filter=firstName.STARTSW.br
```

使用 GET 请求获取名字的开头为“b”且邮件地址包括“bob”的访客用户

```
GET
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser?filter=firstName.STARTSW.b&filter=email
Address.CONTAINS.bob
```

使用 GET 请求获取状态为“Approved”且到期日期为十月份的访客用户

```
GET
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser?filter=status.EQ.Approved&filter=toDate
.STARTSW.10
```

相关主题

获取访客用户，第 7-25 页

页面大小参数

搜索结果默认为每页 20 项资源。页面编号从“0”页开始。每页的最大资源数不能超过 100。非法参数值将产生状态 `400 (Bad Request)` 以及相应的消息。

通过使用表 6-9 中介绍的分页参数，页面大小参数可以覆盖默认值。

在下面的示例中，页面大小更改为 50 项资源：

```
GET
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser?filter=name.STARTSW.b&filter=emailAddre
ss.CONTAINS.bob&size=50&page=0
```

排序参数

默认情况下，排序列是 `name`，排序方向是 `sortasc`。您可以按照下例所示使用参数覆盖默认值：

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser?size=50&page=0&sortdsc=name
```

您可以在表 6-7（第 6-9 页） 中找到可用于排序的值列表。

表 6-9　　可用的分页和排序首选项

参数	描述	默认值
页	页面起始索引	0
size	页面大小	20（最大值 = 100）
sortasc	按升序对字段排序，从一组可用的排序字段中选择字段。（对于字母字符，按 A 到 Z 排序。）	名称
sortdsc	按降序对字段排序，从一组可用的排序字段中选择字段。（对于字母字符，按 Z 到 A 排序。）	名称

示例：获取前 20 个访客用户记录并根据姓氏按升序排序

请求

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser?page=0&size=20&sortasc=lastName
```

响应

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult
    xmlns:ns2="ers.ise.cisco.com" total="22">
    <nextPage type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser?page=1&size=20&sortasc=lastName"
rel="next"/>
    <resources>
        <resource name="aname001" id="8e4bf290-6229-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-Admin-Node:9060/ers/config/guestuser/8e4bf290-6229-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="aname002" id="8fe86480-6229-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-Admin-Node:9060/ers/config/guestuser/8fe86480-6229-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
...
        <resource name="aname020" id="908b5b40-6229-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-Admin-Node:9060/ers/config/guestuser/908b5b40-6229-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
    </resources>
</ns2:searchResult>
```

相关主题

- 通过开头为“ilucky”的用户名进行过滤的示例，第 7-26 页
- 通过开头为“ilucky”的用户名和开头为“J”的姓氏进行过滤的示例，第 7-27 页
- 通过名字“John”进行过滤并按用户名进行排序的示例，第 7-28 页

外部 RESTful 服务 API 操作

概述

本章提供有关外部 RESTful 服务 API 调用的示例，并介绍其使用方法。此外，本章还提供对发出外部 RESTful 服务 API 调用、API 输出架构文件示例以及返回的样本数据的说明。

使用外部 RESTful 服务 API 调用的必备条件

您必须满足以下必备条件，才能发起外部 RESTful 服务 API 调用：

- 您必须已通过 GUI 启用外部 RESTful 服务。
- 您必须具有外部 RESTful 服务管理员权限。

您可以使用 REST 客户端（如 JAVA）、curl linux 命令、python 或任何其他客户端来调用外部 RESTful 服务 API 调用。

相关主题

- 从 GUI 启用外部 RESTful 服务 API，第 5-2 页
- 外部 RESTful 服务 API 身份验证和授权，第 5-2 页

GetVersion

GetVersion 操作对所有可用资源是通用的。它可获取所需资源的版本信息。下表列出此操作的主要特征：

表 7-1　GetVersion 操作的主要特征

说明	检索指定资源的版本信息
摘要	GET/ers/config/<resource-name>/versioninfo
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	版本信息
响应状态	200、400、401、403、404、415、500

GetVersion 操作的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/<resource-type>/versioninfo
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.<resource-namespace>.1.0+xml
```

GetVersion 操作的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.<resource-namespace>.versoininfo.1.0+xml
Content-Length: 122
{
<?xml version="1.0" encoding="UTF-8"?>
<ns2:versionInfo xmlns:ns2="ers.ise.cisco.com">
    <currentServerVersion>1.2</currentServerVersion>
    <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/<resource-type>/versioninfo" rel="self"/>
    <supportedVersions>1.0,1.1,1.2,1.3<supportedVersion>
</ns2:versionInfo>
}
```

适用于内部用户的外部 RESTful 服务 API

适用于内部用户的外部 RESTful 服务 API 支持完整的 CRUD 功能。下表列出适用于内部用户的外部 RESTful 服务 API：

表 7-2　适用于内部用户的外部 RESTful 服务 API

操作	HTTP 方法	URL	内容	查询字符串
获取所有用户	GET	/ers/config/internaluser	不适用	Page、Size、sortacs 或 sortdsn、Filter
获取用户	GET	/ers/config/internaluser/{internal user-id[1]}	不适用	
创建用户	POST	/ers/config/internaluser/	internaluser	
更新用户	PUT	/ers/config/internaluser/{internal user-id}	internaluser	
删除用户	DELETE	/ers/config/internaluser/{internal user-id}	不适用	
获取内部用户资源版本信息	GET	/ers/config/internaluser/version	不适用	

1. 内部用户 ID 是 Cisco ISE 数据库中存储的 UUID 类型。

检索所有内部用户

您可以使用此 API 调用检索 Cisco ISE 中存在的所有内部用户。下表列出此 API 调用的主要特征：

表 7-3　检索所有内部用户 API 调用的主要特征

描述	检索内部用户的集合
摘要	GET /ers/config/internaluser
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、429、500

检索所有内部用户 API 的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/internaluser?page=0&size=20&sortacs=name
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
```

检索所有内部用户 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult total="2" xmlns:ns2="ers.ise.cisco.com">
    <resources>
        <ns2:resource description="description1" name="name1" id="id1"/>
        <ns2:resource description="description2" name="name2" id="id2"/>
    </resources>
</ns2:searchResult>
}
```

通过 ID 获取内部用户

您可以使用此 API 调用通过 ID 获取 Cisco ISE 中的内部用户。下表列出此 API 调用的主要特征：

表 7-4　　读取内部用户 API 调用的主要特征

说明	检索指定的内部用户
摘要	GET /ers/config/internaluser/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	200、400、401、403、404、415、429、500

读取内部用户 API 的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/internaluser/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.internaluser.1.0+xml
```

读取内部用户 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser description="description" name="name" id="id"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <changePassword>true</changePassword>
    <customAttributes>
        <entry>
            <key>key2</key>
            <value>value3</value>
        </entry>
```

```
        <entry>
            <key>key1</key>
            <value>value1</value>
        </entry>
    </customAttributes>
    <email>email@example.com</email>
    <enabled>true</enabled>
    <firstName>John</firstName>
    <identityGroups>identityGroups</identityGroups>
    <lastName>Doe</lastName>
    <password>12345</password>
</ns3:internaluser>
}
```

创建内部用户

您可以使用此 API 调用在 Cisco ISE 中创建内部用户。使用外部 RESTful 服务 API 创建内部用户时，密码为必填内容。下表列出此 API 调用的主要特征：

表 7-5　　创建内部用户 API 调用的主要特征

说明	创建指定的内部用户
摘要	POST /ers/config/internaluser/
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	内部用户
响应头	Content-Length、Content-Type、Location
响应消息正文	内部用户类型的资源
响应状态	201、400、401、403、415、429、500

创建内部用户 API 的请求示例

```
POST https://<ISE-ADMIN-NODE>:9060/ers/config/internaluser/
Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser description="description" name="name" id="id"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <changePassword>true</changePassword>
    <customAttributes>
        <entry>
            <key>key2</key>
            <value>value3</value>
        </entry>
        <entry>
            <key>key1</key>
            <value>value1</value>
        </entry>
    </customAttributes>
    <email>email@example.com</email>
    <enabled>true</enabled>
    <firstName>John</firstName>
    <identityGroups>identityGroups</identityGroups>
```

```
    <lastName>Doe</lastName>
    <password>12345</password>
</ns3:internaluser>
}
```

创建内部用户 API 的响应示例

```
HTTP/1.1 201 OK  (see the location header for the new user's ID)
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
Location: https://<ISE-ADMIN-NODE>/ers/config/internaluser/444
```

更新内部用户

您可以使用此 API 调用更新 Cisco ISE 中的内部用户。如果在使用外部 RESTful 服务 API 更新内部时不更改密码，您必须将密码设置为“********”。下表列出此 API 调用的主要特征：

表 7-6　　更新内部用户 API 调用的主要特征

说明	更新指定的内部用户
摘要	PUT /ers/config/internaluser/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	内部用户
响应头	Content-Length、Content-Type
响应消息正文	更新字段的列表
响应状态	200、400、401、403、404、415、429、500

更新内部用户 API 的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/internaluser/333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser description="description" name="name" id="333"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <changePassword>true</changePassword>
    <customAttributes>
        <entry>
            <key>key2</key>
            <value>value3</value>
        </entry>
        <entry>
            <key>key1</key>
            <value>value1</value>
        </entry>
    </customAttributes>
    <email>email@example.com</email>
    <enabled>true</enabled>
    <firstName>John</firstName>
    <identityGroups>IdentityGroups</identityGroups>
    <lastName>Doe</lastName>
```

```
    <password>********</password>
</ns3:internaluser>
}
```

更新内部用户 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.updatedfields.1.0+xml
Content-Length: 529
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
    <updatedField field="lastname">
        <newValue>Doe</newValue>
        <oldValue>name</oldValue>
    </updatedField>
    <updatedField field="dentityGroups">
        <newValue>IdentityGroups</newValue>
        <oldValue>zzz</oldValue>
    </updatedField>
</ns2:updatedFields>
}
```

删除内部用户

您可以使用此 API 调用删除 Cisco ISE 中的内部用户。下表列出此 API 调用的主要特征：

表 7-7　　删除内部用户 API 调用的主要特征

说明	删除指定的内部用户
摘要	DELETE /ers/config/internaluser/{internaluser-id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	204、400、401、403、404、415、429、500

删除内部用户 API 的请求示例

```
DELETE https://<ISE-ADMIN-NODE>:9060/ers/config/internaluser/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
```

删除内部用户 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

适用于终端的外部 RESTful 服务 API

下表列出适用于终端的外部 RESTful 服务 API：

表 7-8　　适用于终端的外部 RESTful 服务 API

操作	方法	URL	内容	查询字符串
获取所有终端	GET	/ers/config/endpoint	不适用	page、size、sortacs 或 sortdsn、filter
获取终端	GET	/ers/config/endpoint/{id[1]}	不适用	
创建终端	POST	/ers/config/endpoint/	终端	
更新终端	PUT	/ers/config/endpoint/{id}	终端	
删除终端	DELETE	/ers/config/endpoint/{id}	不适用	
注册终端	PUT[2]	/ers/config/endpoint/register	终端	
撤销注册终端	PUT	/ers/config/endpoint/{id}/deregister	不适用	
获取终端资源版本信息	GET	/ers/config/endpoint/version	不适用	

1. 终端 ID 是 Cisco ISE 数据库中存储的 UUID 类型。
2. 如果终端已经存在，则会执行终端注册。如果终端不存在，则会先创建终端，再执行终端注册。在这两种情况下，返回的状态均为 204。

获取所有终端

获取所有终端的 API 仅用于检索与过滤器中指定的用户关联的终端。下表列出此 API 调用的主要特征：

表 7-9　　获取所有终端 API 调用的主要特征

说明	检索与指定内部用户关联的终端的集合
摘要	GET /ers/config/endpoint/
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、429、500

获取所有终端 API 的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint?filter=userid.EQ.123
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
```

获取所有终端 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult total="2" xmlns:ns2="ers.ise.cisco.com">
<resources>
<resource name=name1" id="id1">
   <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/id1" rel="self"/>
  </resource>
  <resource name="name2" id="id2">
   <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/id2" rel="self"/>
  </resource>
</resources>
</ns2:searchResult>
}
```

通过 ID 获取终端

您可以使用此 API 调用通过 ID 获取 Cisco ISE 中的终端。下表列出此 API 调用的主要特征：

表 7-10　　读取终端 API 调用的主要特征

说明	检索指定的终端
摘要	GET /ers/config/endpoint/{endpoint-id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	200、400、401、403、404、415、429、500

读取终端 API 的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
```

读取终端 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<ns3:endpoint description="description" name="name" id="id" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <group>group</group>
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profile>profile</profile>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
}
```

创建终端

您可以使用此 API 调用在 Cisco ISE 中创建终端。下表列出此 API 调用的主要特征：

表 7-11　　创建终端 API 调用的主要特征

描述	创建指定的终端
摘要	POST /ers/config/endpoint/
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	200、400、401、403、404、415、429、500

创建终端 API 的请求示例

```
POST https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/
Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint description="description" name="name" id="id" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <group>group</group>
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profile>profile</profile>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
}
```

创建终端 API 的响应示例

```
HTTP/1.1 201 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
Location: https://cisco.com/ers/config/endpoint/444
```

更新终端

您可以使用此 API 调用更新 Cisco ISE 中的终端。下表列出此 API 调用的主要特征：

表 7-12　　更新终端 API 调用的主要特征

说明	更新指定的终端
摘要	PUT /ers/config/endpoint/{endpoint-id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	200、400、401、403、404、415、429、500

更新终端 API 的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:endpoint xmlns:ns2="identity.ers.ise.cisco.com" description="updated"
name="Endpoint">
    <mac>AA:AA:AA:AA:AA:AA</mac>
    <staticGroupAssignment>false</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns2:endpoint>}
```

更新终端 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.updatedfields.1.0+xml
Content-Length: 529
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
<updatedField field="mac">
  <newValue>AA:AA:AA:AA:AA:AA</newValue>
  <oldValue>BB:BB:BB:BB:BB:BB</oldValue>
 </updatedField>
 <updatedField field="staticGroupAssignment">
  <newValue>false</newValue>
  <oldValue>true</oldValue>
```

```
 </updatedField>
<updatedField field="staticProfileAssignment">
  <newValue>false</newValue>
  <oldValue>true</oldValue>
 </updatedField>
</ns2:updatedFields>
}
```

删除终端

您可以使用此 API 调用更新 Cisco ISE 中的终端。下表列出此 API 调用的主要特征：

表 7-13　　删除终端 API 调用的主要特征

说明	删除指定的终端
摘要	DELETE /ers/config/endpoint/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	200、400、401、403、404、415、429、500

删除终端 API 的请求示例

```
DELETE https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
```

删除终端 API 的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

注册终端

您可以使用此 API 调用注册 Cisco ISE 中的终端。如果终端尚不存在，将创建终端。与 GUI 注册流程类似，终端静态分配给已注册的设备组，并按照内容中指定的值设置门户用户和身份库。

下表列出此 API 调用的主要特征：

表 7-14　　注册终端 API 调用的主要特征

说明	注册指定的终端
摘要	PUT /ers/config/endpoint/register
请求头	Accept、Authorization、Host
查询字符串	N/A

表 7-14　注册终端 API 调用的主要特征 （续）

请求消息正文	终端
响应头	Content-Length、Content-Type
响应消息正文	更新字段的列表
响应状态	202、400、401、403、404、415、429、500

注册终端 API 的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/register
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint description="description" name="name" id="id" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <group>group</group>
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profile>profile</profile>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
}
```

注册终端 API 的响应示例

```
HTTP/1.1 204 No Content
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

撤销注册终端

您可以使用此 API 调用对 Cisco ISE 中的终端撤销注册。结果中不显示任何内容。下表列出此 API 调用的主要特征：

表 7-15　撤销注册终端 API 调用的主要特征

说明	撤销注册指定的终端
摘要	PUT /ers/config/endpoint/{id}/deregister
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应消息正文	N/A
响应状态	202、400、401、403、404、415、429、500

撤销注册终端 API 调用的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/123/deregister
Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
```

撤销注册终端 API 调用的响应示例

```
HTTP/1.1 204 No Content
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

启动终端批量执行

通过批量执行，您可以在单个请求中发送最多 500 项类型相同的 CRUD 操作。

如果请求有效，服务器将返回状态代码 202 (ACCEPTED)，并在 LOCATION 响应头中包含唯一的批量标识符，您可以使用此标识符通过获取批量状态操作跟踪批量状态。

一次只允许运行一个批量操作。如果在另一个批量操作仍在执行时发布批量操作请求，服务器将返回响应状态 503 (Service Unavailable)，并包含相应的要求客户端稍后重试的描述性消息。

表 7-16　　启动终端批量执行的主要特征

说明	启动执行
摘要	PUT /ers/config/endpoint/bulk
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	批量请求
响应头	Content-Length、Content-Type
响应消息正文	不适用
响应状态	202、400、401、403、404、415、500

启动终端批量执行 API 调用的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/bulk HTTP/1.1
Host: {some-ise-node-ip}
Authorization: Basic xxxxxxxxxxxxxxxxxx Content-Type:
application/vnd.com.cisco.ise.ers.endpointbulkrequest.1.0+xml
{
<ns3:endpointBulkRequest
resourceMediaType = "vnd.com.cisco.ise.ers.identity.endpoint.1.0+xml" operationType =
"create"
xmlns:ns2 = "ers.ise.cisco.com"
xmlns:ns3 = "identity.ers.ise.cisco.com">
<resourcesList> <resource
xsi:type = "ns3:ersEndPoint"
description = "created by bulk request"
xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"> <mac>11:22:33:44:55:66</mac>
<staticGroupAssignment>false</staticGroupAssignment>
<staticProfileAssignment>false</staticProfileAssignment>
</resource>
```

```
. . .
<resource
xsi:type = "ns3:ersEndPoint"
description = "created by bulk request"
xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"> <group>Profiled</group>
<groupId>804f5350-7808-11e2-bdd0-0050568e01f0</groupId> <identityStore></identityStore>
<identityStoreId></identityStoreId> <mac>11:22:33:44:55:77</mac>
<portalUser></portalUser>
<profile>Apple-iPod</profile> <profileId>b8128870-7808-11e2-bdd0-0050568e01f0</profileId>
<staticGroupAssignment>true</staticGroupAssignment>
<staticProfileAssignment>true</staticProfileAssignment>
</resource> </resourcesList>
</ns3:endpointBulkRequest>
}
```

启动终端批量执行 API 调用的响应示例

```
HTTP/1.1 202 ACCEPTED
        Date: Thu, 12 Jul 2012 23:59:59 GMT
        Location: https://ise-node-ip:9060/ers/config/endpoint/123443545334
```

获取终端批量状态

如果批量执行请求有效且没有其他正在处理的批量操作，服务器将在 LOCATION 响应头中返回唯一的批量标识符。使用此 ID 可跟踪批量状态。您可以获取操作开始之后至少 2 小时的状态报告。

表 7-17　　获取批量状态的主要特征

说明	监控指定的批量执行进度
摘要	GET /ers/config/endpoint/bulk/{bulkid}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	批量状态
响应状态	200, 400, 401, 403, 404, 415, 500

获取终端批量状态示例

请求

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/bulk/53454354534 HTTP/1.1
    Host: {some-ise-node-ip}
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.bulkStatus.1.0+xml
```

响应

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.bulkStatus.1.0+xml Content-Length: 16347
{
<ns2:bulkStatus
```

```
            xmlns:ns2 = "ers.ise.cisco.com"
            successCount = "750"
            startTime = "Thu Mar 07 17:17:35 IST 2013"
            resourcesCount = "750"
            operationType = "create"
            mediaType =
    "vnd.com.cisco.ise.ers.identity.endpoint.1.0+xml"
            failCount = "0"
            executionStatus = "COMPLETED"
            bulkId = "1362669455284">
            <resourcesStatus>
                  <resourceStatus status = "SUCCUESS"
                        description = "created by bulk request"
                        id = "23d068d0-873a-11e2-bad4-00215edbb2a8" />
. . . .
                  <resourceStatus
                        status = "SUCCUESS"
                        description = "created by bulk request"
                        id = "23cfa580-873a-11e2-bad4-00215edbb2a8"/>
            </resourcesStatus>
    </ns2:bulkStatus>
}
}
```

适用于终端证书的外部 RESTful 服务 API

下表列出适用于终端证书的外部 RESTful 服务 API：

表 7-18　　适用于终端证书的 ERS API

操作	方法	URL	内容	查询字符串
创建终端证书	POST	/ers/config/endpointcert/certRequest/	表示 ZIP 文件的八位字节流	
获取证书资源版本信息	GET	/ers/config/endpointcert/version	不适用	

创建终端证书

下表列出创建终端证书 API 调用的主要特征：

:

表 7-19　　创建终端身份组 API 调用的主要特征

说明	创建终端证书
摘要	POST /ers/config/endpointcert/certRequest/
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	终端证书
响应头	Content-Length、Content-Type、Location
响应消息正文	表示 ZIP 文件的八位字节流
响应状态	201、400、401、403、415、429、500

创建终端证书 API 调用的请求示例

```
PUT https://<ISE-ADMIN-NODE>/ers/config/endpointcert/certRequest
HTTP Content-Type header:
application/vnd.com.cisco.ise.ca.endpointcert.1.0+xml; charset=utf-8

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpointcert xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="ca.ers.ise.cisco.com">
    <certTemplateName>Certificate_Template_Name</certTemplateName>
    <certificateRequest>
        <entry>
            <key>san</key>
            <value>11-22-33-44-55-66</value>
        </entry>
        <entry>
            <key>cn</key>
            <value>userName [or] machineName</value>
        </entry>
    </certificateRequest>
    <format>PKCS8 [or] PKCS8_CHAIN [or] PKCS12 [or] PKCS12_CHAIN</format>
    <password>password</password>
</ns3:endpointcert>
```

创建终端证书 API 调用的响应示例

```
HTTP Status: 200 (OK)
Content:
[Response is returned as an Octet Stream representing a ZIP file.]
```

适用于终端身份组的外部 RESTful 服务 API

下表列出适用于终端身份组的外部 RESTful 服务 API：

表 7-20　　适用于终端身份组的 ERS API

操作	方法	URL	内容	查询字符串
获取所有终端组	GET	/ers/config/endpointgroup	不适用	page、size、sortacs 或 sortdsn、filter
获取终端组	GET	/ers/config/endpointgroup/{id[1]}	不适用	
创建终端组	POST	/ers/config/endpointgroup/	终端组	
更新终端组	PUT	/ers/config/endpointgroup/{id}	终端组	
删除终端组	DELETE	/ers/config/endpointgroup/{id}	不适用	
获取身份组资源版本信息	GET	/ers/config/ endpointgroup /version	不适用	

1. 终端组 ID 是 Cisco ISE 数据库中存储的 UUID 类型。

获取所有终端身份组

下表列出获取所有终端身份组 API 调用的主要特征：

表 7-21　　获取所有终端身份组 API 调用的主要特征

说明	检索终端组的集合
摘要	GET /ers/config/endpoint
请求头	Accept、Authorization、Host
查询字符串	page、size、sortasc、sortdsc、filter
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	更新字段的列表
响应状态	202、400、401、403、404、415、429、500

获取所有终端身份组 API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/endpointgroup
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
```

获取所有终端身份组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult total="2" xmlns:ns2="ers.ise.cisco.com">
<resources>
  <resource name="name1" id="id1" description="description1">
   <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/endpointgroup/id1" rel="self"/>
  </resource>
  <resource name="name2" id="id2" description="description2">
   <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/endpointgroup/id2" rel="self"/>
  </resource>
</resources>
</ns2:searchResult>
}
```

通过 ID 获取终端身份组

下表列出通过 ID 获取终端身份组 API 调用的主要特征：

表 7-22　　读取终端身份组 API 调用的主要特征

说明	检索指定的终端组
摘要	GET /ers/config/endpoint/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	终端资源类型
响应状态	200、400、401、403、404、415、429、500

读取终端身份组 API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
```

读取终端身份组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpointgroup description="description" name="name" id="id"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <systemDefined>true</systemDefined>
</ns3:endpointgroup>
}
```

创建终端身份组

下表列出创建终端身份组 API 调用的主要特征：

:

表 7-23　　创建终端身份组 API 调用的主要特征

说明	创建指定的终端组
摘要	POST /ers/config/endpoint/
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	终端组

表 7-23　创建终端身份组 API 调用的主要特征 （续）

响应头	Content-Length、Content-Type、Location
响应消息正文	终端组
响应状态	201、400、401、403、415、429、500

创建终端身份组 API 调用的请求示例

```
POST https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint
Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpointgroup description="description" name="name" id="id"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <systemDefined>true</systemDefined>
</ns3:endpointgroup>
}
```

创建终端身份组 API 调用的响应示例

```
HTTP/1.1 201 OK（请从 Location 标头查看新终端 ID）
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type:
Location: https://cisco.com/ers/config/endpointgroup/444
```

更新终端身份组

下表列出更新终端身份组 API 调用的主要特征：

表 7-24　更新终端身份组 API 调用的主要特征

说明	更新指定的终端组
摘要	PUT /ers/config/endpoint/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	终端组
响应头	Content-Length、Content-Type
响应消息正文	更新字段的列表
响应状态	200、400、401、403、404、415、429、500

更新终端身份组 API 调用的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/ endpoint /333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:endpointgroup xmlns:ns2="identity.ers.ise.cisco.com" description="updated" id="0"
name="Group">
    <systemDefined>false</systemDefined>
</ns2:endpointgroup>
}
```

更新终端身份组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.updatedfields.1.0+xml
Content-Length: 529
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
 <updatedField field="description">
  <newValue>updated</newValue>
  <oldValue>Group</oldValue>
 </updatedField>
</ns2:updatedFields>
}
```

删除终端身份组

下表列出删除终端身份 API 调用的主要特征：

表 7-25　　删除终端身份组 API 调用的主要特征

说明	删除指定的终端组
摘要	DELETE /ers/config/endpointgroup/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	不适用
响应状态	204、400、401、403、404、415、429、500

删除终端身份组 API 调用的请求示例

```
DELETE https://<ISE-ADMIN-NODE>:9060/ers/config/endpoint/333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
```

删除终端身份组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

适用于身份组的外部 RESTful 服务 API

下表列出适用于身份组的外部 RESTful 服务 API：

表 7-26　适用于身份组的 API

操作	方法	URL	内容	查询字符串
获取所有身份组	GET	/ers/config/identitygroup	不适用	page、size、sortacs 或 sortdsn、filter
通过 ID 获取身份组	GET	ers/config/identitygroup/{id}	不适用	
获取身份组资源版本信息	GET	/ers/config/identitygroup/version	不适用	

检索所有身份组

您可以使用此 API 调用检索 Cisco ISE 中的所有身份组。下表列出此 API 调用的主要特征：

表 7-27　检索所有身份组 API 调用的主要特征

说明	检索身份组资源的集合
摘要	GET /ers/config/identitygroup
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、429、500

检索所有身份组 API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/identitygroup?page=0&size=20&sortacs=name
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.identitygroup.1.0+xml
```

检索所有身份组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult total="2" xmlns:ns2="ers.ise.cisco.com">
    <resources>
      <resource name="name1" id="id1" description="description1">
        <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/identitygroup/id1" rel="self"/>
      </resource>
      <resource name="name2" id="id2" description="description2">
        <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/identitygroup/id2" rel="self"/>
      </resource>
    </resources>
</ns2:searchResult>
}
```

通过 ID 获取身份组

下表列出通过 ID 获取身份组 API 调用的主要特征：

表 7-28　　读取终端身份组 API 调用的主要特征

说明	检索指定的身份组
摘要	GET /ers/config/identitygroup/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	终端资源类型
响应状态	200、400、401、403、404、415、429、500

通过 ID 获取身份组 API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/identitygroup/333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.identitygroup.1.0+xml
```

通过 ID 获取身份组 API 调用的响应示例

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:identitygroup name="name" id="id" description="description"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <parent>parent</parent>
</ns3:identitygroup>
```

适用于访客用户的外部 RESTful 服务 API

下表列出适用于访客用户的外部 RESTful 服务 API：

表 7-29　　支持的场景

操作	方法	URL	内容
获取特定访客用户	GET	/ers/config/guestuser/{id}	不适用
获取所有访客用户	GET	/ers/config/guestuser/	不适用
创建访客用户	POST	/ers/config/guestuser /	访客用户信息 (XML)
更新访客用户	PUT	/ers/config/guestuser/{id}	部分或全部访客用户信息 (XML)
删除访客用户	DELETE	/ers/config/guestuser/{id}	不适用
暂停访客用户	PUT	/ers/config/guestuser/suspend/{id}	原因
恢复访客用户	PUT	/ers/config/guestuser/reinstate/{id}	不适用
发送电子邮件	PUT	/ers/config/guestuser/email/{id}/ portalId/{portalId}	发件人邮件
发送 SMS	PUT	/ers/config/guestuser/sms/{id}/portalId /{portalId}	不适用
批准访客用户	PUT	/ers/config/guestuser/approve/{id}	不适用
拒绝访客用户	PUT	/ers/config/guestuser/deny/{id}	不适用
重置密码	PUT	/ers/config/guestuser/resetpassword/ {id}	不适用
启动批量执行	PUT	/ers/config/ guestuser/bulk	批量请求
获取批量状态	GET	/ers/config/ guestuser/bulk/{bulkId}	不适用
更改发起人的密码	PUT	/ers/config/guestuser/changeSponsor Password/{portalId}	operationAdd itionalData
获取所有门户	GET	/ers/config/portal	不适用
通过 ID 获取门户	GET	/ers/config/portal/{id}	不适用
获取访客 API 信息	GET	/ers/config/guestuser/versioninfo	不适用

Content Type 和 Accept 标头

每个访客帐户请求都需要进行以下设置：

- Content Type 设置为：`application/vnd.com.cisco.ise.identity.guestuser.2.0+xml`
- Accept 设置为：`application/vnd.com.cisco.ise.identity.guestuser.2.0+xml`

每个批量执行请求都需要进行以下设置：

- Content Type 设置为：
  `application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml`
- Accept 设置为：`application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml`

获取访客用户

您可以通过 GET 操作，使用访客的用户名或数据库记录 ID 从 ISE 数据库检索特定访客用户。

表 7-30　　获取访客用户主要特征

说明	检索指定的访客用户
摘要	GET /ers/config/guestuser/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	访客用户类型的资源
响应状态	200, 400, 401, 403, 404, 415, 500

获取访客用户示例

- 通过 ID 获取访客用户的示例，第 7-25 页
- 通过开头为“ilucky”的用户名进行过滤的示例，第 7-26 页
- 通过开头为“ilucky”的用户名和开头为“J”的姓氏进行过滤的示例，第 7-27 页
- 通过名字“John”进行过滤并按用户名进行排序的示例，第 7-28 页
- 使用 curl 的访客用户请求和响应示例，第 7-28 页

通过 ID 获取访客用户的示例

请求

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/3333


Content-Type - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Accept - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Authorization - Basic xxxxxxxxxxxxxxxxxxx
```

响应

```
<?xml version="1.0" encoding="UTF-8"?>
<ns3:guestuser xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com"
name="fzvhervocnz" id="af827350-1f0f-11e4-b961-005056103001">
 <link type="application/xml" href="https://10.0.10.130:9060/ers/config/guestuser/3333"
rel="self"/>
 <customFields/>
 <guestAccessInfo>
  <fromDate>08/08/2014 8:21</fromDate>
  <toDate>08/09/2014 8:21</toDate>
  <validDays>1</validDays>
 </guestAccessInfo>
 <guestInfo>
  <company>Cisco</company>
  <creationTime>08/08/2014 08:21</creationTime>
  <emailAddress>doe@example.com</emailAddress>
  <enabled>true</enabled>
  <firstName>John</firstName>
```

```
  <lastName>Doe</lastName>
  <notificationLanguage>English</notificationLanguage>
  <password>12345</password>
  <phoneNumber>9999998877</phoneNumber>
  <smsServiceProvider>ATT</smsServiceProvider>
  <userName>Guest1</userName>
 </guestInfo>
 <guestType>Daily (default)</guestType>
 <personBeingVisited>sponsor@example.com</personBeingVisited>
 <reasonForVisit>Interview</reasonForVisit>
 <sponsorUserName>SponsoredUser1</sponsorUserName>
 <status>Awaiting Initial Login</status>
</ns3:guestuser>
```

相关主题

有关在 API 中是否显示密码的详细信息，请参阅访客密码，第 6-6 页。

通过开头为“ilucky”的用户名进行过滤的示例

请求

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/?filter=userName.STARTSW.ilucky
```

响应

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult
    xmlns:ns2="ers.ise.cisco.com" total="8">
    <resources>
        <resource name="ilucky101" id="a0957160-6224-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/a0957160-6224-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky102" id="e14f4460-6224-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/e14f4460-6224-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky201" id="123581f0-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/123581f0-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky301" id="154f9330-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/154f9330-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky401" id="172c6980-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/172c6980-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky501" id="19631fa0-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/19631fa0-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky601" id="1b44b0e0-6227-11e3-9bc2-000c2932c73c">
```

```
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/1b44b0e0-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky602" id="2e1ac600-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/2e1ac600-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
    </resources>
</ns2:searchResult>
```

通过开头为“ilucky”的用户名和开头为“J”的姓氏进行过滤的示例

请求

```
GET
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/?filter=userName.STARTSW.ilucky&filter=
lastName.STARTSW.j
```

响应

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult
    xmlns:ns2="ers.ise.cisco.com" total="8">
    <resources>
        <resource name="ilucky101" id="a0957160-6224-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/a0957160-6224-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky102" id="e14f4460-6224-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/e14f4460-6224-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky201" id="123581f0-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/123581f0-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky301" id="154f9330-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/154f9330-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky401" id="172c6980-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/172c6980-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky501" id="19631fa0-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/19631fa0-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky601" id="1b44b0e0-6227-11e3-9bc2-000c2932c73c">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/1b44b0e0-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
        <resource name="ilucky602" id="2e1ac600-6227-11e3-9bc2-000c2932c73c">
```

```
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/2e1ac600-6227-11e3-9bc2-000c2932c73
c" rel="self"/>
        </resource>
    </resources>
</ns2:searchResult>
```

通过名字“John”进行过滤并按用户名进行排序的示例

请求

```
GET
https://<ISE-Admin-node>:9060/ers/config/guestuser/?page=0&size=10&sortdsc=name&filter=fir
stName.eq.john
```

响应

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult
    xmlns:ns2="ers.ise.cisco.com" total="2">
    <resources>
        <resource name="jdoe0002" id="886f5b40-5ece-11e3-8faf-000c29c56fc6">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/886f5b40-5ece-11e3-8faf-000c29c56fc
6" rel="self"/>
        </resource>
        <resource name="jdoe0001" id="79e5a5a0-5df9-11e3-84f5-000c29c56fc6">
            <link type="application/xml"
href="https://ISE-ADMIN-NODE:9060/ers/config/guestuser/79e5a5a0-5df9-11e3-84f5-000c29c56fc
6" rel="self"/>
        </resource>
    </resources>
</ns2:searchResult>
```

使用 curl 的访客用户请求和响应示例

以下示例介绍使用 `curl` Linux 命令通过向 ISE 发送的 ID 获取访客用户的请求及其响应。

curl 命令

```
$ curl -v -k -H 'ACCEPT:application/vnd.com.cisco.ise.identity.guestuser.2.0+xml'
https://username:password@<ISE-ADMIN-NODE>:9060/ers/config/guestuser/user1
* About to connect() to <ISE-ADMIN-NODE> port 9060
*   Trying 111.11.11.111... * connected
* Connected to <ISE-ADMIN-NODE> (<ISE-ADMin-NODE-IP>) port 9060
* successfully set certificate verify locations:
*   CAfile: /usr/share/ssl/certs/ca-bundle.crt
  CApath: none
* SSL connection using DHE-RSA-AES256-SHA
* Server certificate:
*        subject: /CN=<ISE-ADMIN-NODE>
*        start date: 2013-11-26 00:56:55 GMT
*        expire date: 2014-11-26 00:56:55 GMT
*        common name: <ISE-ADMIN-NODE>
*        issuer: /CN=<ISE-ADMIN-NODE>
* Server auth using Basic with user 'username'
```

通过 ID 获取访客用户的请求

```
> GET /ers/config/guestuser/444
Authorization: Basic xxxxxxxxxxxxxxxx
User-Agent: curl/7.12.1 (i386-redhat-linux-gnu) libcurl/7.12.1 OpenSSL/0.9.7a zlib/1.2.1.2
libidn/0.5.6
Host: <ISE-ADMIN-NODE>:9060
Pragma: no-cache
ACCEPT:application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

响应

```
< HTTP/1.1 200 OK
< Pragma: No-cache
< Cache-Control: no-cache
< Expires: Wed, 31 Dec 1969 16:00:00 PST
< Set-Cookie: JSESSIONIDSSO=0FCBC2621A0897193FE3105B3FBA8F16; Path=/; Secure
< Set-Cookie: JSESSIONID=5B6092B3FCCE047F7282C52592FAFC7A; Path=/ers; Secure
< Date: Thu, 02 Jan 2014 23:01:59 GMT
< Content-Type: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
< Content-Length: 1162
< Server:
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:guestuser xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com"
name="user1" id="b4bdf2b0-73e1-11e3-8cdf-000c29c56fc6">
<link type="application/xml" href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/444"
rel="self"/>
<guestAccessInfo>
<fromDate>08/06/2014 23:26</fromDate>
<toDate>08/07/2014 23:26</toDate>
<validDays>1</validDays>
</guestAccessInfo>
<guestInfo>
<company>New Company</company>
<emailAddress>john@example.com</emailAddress>
<firstName>John</firstName>
<lastName>Doe</lastName>
<notificationLanguage>English</notificationLanguage>
<phoneNumber>9999998877</phoneNumber>
<smsServiceProvider>Global Default</smsServiceProvider>
<userName>user1</userName>
</guestInfo>
<guestType>Daily (default)</guestType>
<personBeingVisited>sponsor@example.com</personBeingVisited>
<portalId>ff2d99e0-2101-11e4-b5cf-005056bf2f0a</portalId>
<reasonForVisit>Interview</reasonForVisit>
</ns3:guestuser>
```

相关主题

有关在 API 中是否显示密码的详细信息，请参阅访客密码，第 6-6 页。

获取所有访客用户

您可以使用 GET 操作检索 ISE 数据库中的所有访客用户，并根据名称、用户名或邮件地址等条件过滤结果。响应包括访客的用户名、ID 和连接到其完整表示的链接。

表 7-31　　获取所有访客用户的主要特征

说明	检索访客用户的集合
摘要	GET /ers/config/guestuser/
请求头	Accept、Authorization、Host
查询字符串	page、size、sortasc、sortdsc、filter
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、500

相关主题

过滤参数，第 6-9 页

获取所有访客用户示例

在下面的示例中，GET 操作会检索用户名以 `ilu` 开头且名字以 `b` 开头的所有访客用户。

请求

```
GET
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/?page=0&size=10&sortasc=name&filter=nam
e.STARTSW.ilu&filter=firstName.STARTSW.b

Content-Type - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Accept - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Authorization - Basic xxxxxxxxxxxxxxxxxx
```

响应

```
HTTP/1.1 200 OK;
Date:Sat, 15 Dec 2012 21:55:05 GMT;
Content-Length:1439;
Content-Type:application/vnd.com.cisco.ise.ers.searchresult.1.0+xml;

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult xmlns:ns2="ers.ise.cisco.com" total="6">
  <resources>
    <ns2:resource name="ilucky01" id="61dc9060-46a1-11e2-b141-000c290fcf9a">
      <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/61dc9060-46a1-11e2-b141-000c290fc
f9a" rel="self"/>
    </ns2:resource>
    <ns2:resource name="ilucky02" id="3f43bb40-468e-11e2-8f92-000c290fcf9a">
      <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/3f43bb40-468e-11e2-8f92-000c290fc
f9a" rel="self"/>
    </ns2:resource>
    <ns2:resource name="ilucky03" id="6c65d6d0-468e-11e2-8f92-000c290fcf9a">
```

```
      <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/6c65d6d0-468e-11e2-8f92-000c290fc
f9a" rel="self"/>
    </ns2:resource>
    <ns2:resource name="ilucky04" id="6948bdb0-46a1-11e2-b141-000c290fcf9a">
      <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/6948bdb0-46a1-11e2-b141-000c290fc
f9a" rel="self"/>
    </ns2:resource>
    <ns2:resource name="ilucky05" id="abbb6440-46a1-11e2-b141-000c290fcf9a">
      <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/abbb6440-46a1-11e2-b141-000c290fc
f9a" rel="self"/>
    </ns2:resource>
    <ns2:resource name="ilucky06" id="4d9a1530-46fd-11e2-b70b-000c290fcf9a">
      <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/4d9a1530-46fd-11e2-b70b-000c290fc
f9a" rel="self"/>
    </ns2:resource>
  </resources>
</ns2:searchResult>
```

创建访客用户

您可以使用 POST 操作创建新的访客用户帐户，此帐户使访客可以通过访客流程登录。

创建访客用户帐户时必须提供 guestType。

表 7-32　　创建访客用户的主要特征

说明	创建指定的内部用户
摘要	POST /ers/config/guestuser/
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	访客用户
响应头	Content-Length、Content-Type、Location
响应消息正文	访客用户类型的资源
响应状态	201、400、401、403、415、500

访客用户 XML 结构

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:guestuser name="guestUser" id="123456789" description="ERS Example user "
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <customFields>
        <entry>
            <key>some key</key>
            <value>some value</value>
        </entry>
        <entry>
            <key>another key</key>
            <value>and its value</value>
        </entry>
    </customFields>
    <guestInfo>
```

```
        <emailAddress>email@some.uri.com</emailAddress>
        <enabled>true</enabled>
        <password>asdlkj324ew</password>
        <phoneNumber>3211239034</phoneNumber>
        <smsServiceProvider>GLobal Default</smsServiceProvider>
        <userName>DS3ewdsa34wWE</userName>
    </guestInfo>
    <guestType>Contractor</guestType>
    <portalId>23423432523</portalId>
    <sponsorUserName>Mr Spons</sponsorUserName>
</ns3:guestuser>
```

创建访客用户示例

请求

```
POST https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/

Content-Type - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Accept - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Authorization - Basic xxxxxxxxxxxxxxxxxx

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:guestuser xmlns:ns2="identity.ers.ise.cisco.com">
    <guestAccessInfo>
        <fromDate>08/08/2014 08:15</fromDate>
        <toDate>08/09/2014 08:15</toDate>
        <validDays>1</validDays>
    </guestAccessInfo>
    <guestInfo>
        <company>New Company</company>
        <emailAddress>doe@example.com</emailAddress>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
        <notificationLanguage>English</notificationLanguage>
        <phoneNumber>9999998877</phoneNumber>
        <smsServiceProvider>Global Default</smsServiceProvider>
        <userName>guestuser1</userName>
    </guestInfo>
    <guestType>Daily (default)</guestType>
    <personBeingVisited>sponsor@example.com</personBeingVisited>
    <portalId>ff2d99e0-2101-11e4-b5cf-005056bf2f0a</portalId>
    <reasonForVisit>Interview</reasonForVisit>
</ns2:guestuser>
```

响应

```
HTTP/1.1 201 Created;
Date:Sat, 15 Dec 2012 21:20:51 GMT;
Content-Length:0;
Location:https://<ISE-ADMIN-NODE>/ers/config/guestuser/e1bb8290-6ccb-11e3-8cdf-000c29c56fc
6;
Set-Cookie:JSESSIONID=28CF43F1ACCC7448BED7255DC7B787EE; Path=/ers;
Secure;JSESSIONIDSSO=DB6D6900088D1863CA84863570392E4C; Path=/; Secure;
Content-Type:application/xml;
```

相关主题

有关在 API 中是否显示密码的详细信息，请参阅访客密码，第 6-6 页。

更新访客用户

通过使用 PUT 操作更新资源，您可以更改现有访客用户的属性。您可以全部或部分更新访客用户的属性。

表 7-33　　更新访客用户的主要特征

说明	更新指定的访客用户
摘要	PUT /ers/config/guestuser/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	访客用户
响应头	Content-Length、Content-Type
响应消息正文	更新字段的列表
响应状态	200、400、401、403、404、415、500

更新用户示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/name/ilucky101

Content-Type - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Accept - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Authorization - Basic xxxxxxxxxxxxxxxxxx

<?xml version="1.0" encoding="UTF-8"?>
<ns2:guestuser xmlns:ns2="identity.ers.ise.cisco.com">
    <portalId>ff2d99e0-2101-11e4-b5cf-005056bf2f0a</portalId>
    <reasonForVisit>Interview</reasonForVisit>
</ns2:guestuser>
```

响应

```
Status:200 OK

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
 <updatedField field="ReasonForVisit">
  <newValue>Interview</newValue>
  <oldValue>no reason</oldValue>
 </updatedField>
 <updatedField field="validDays">
  <newValue>0</newValue>
  <oldValue>1</oldValue>
 </updatedField>
</ns2:updatedFields>
```

删除访客用户

您可以使用数据库记录 ID 从 ISE 数据库删除访客用户的记录。用户在下次尝试时将无法登录。

表 7-34　　删除访客用户的主要特征

说明	删除指定的访客用户
摘要	DELETE /ers/config/guestuser/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	不适用
响应状态	204、400、401、403、404、415、500

删除访客用户示例

请求

```
DELETE https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/3333

Content-Type - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Accept - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Authorization - Basic xxxxxxxxxxxxxxxxxx
```

响应

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

暂停访客用户

使用 PUT 操作可暂停特定访客用户。用户在下次尝试时将无法登录。您必须包含暂停原因，原因可以包含空格。

表 7-35　　暂停访客用户的主要特征

说明	暂停指定的访客用户
摘要	PUT /ers/config/guestuser/suspend/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	原因
响应头	Content-Length、Content-Type
响应消息正文	访客用户类型的资源
响应状态	204、400、401、403、404、415、500

通过 ID 暂停访客用户示例

请求

```
PUT https:/<ISE-ADMIN-NODE>:9060/ers/config/guestuser/suspend/3333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
Content-Type - application/vnd.com.cisco.ise.identity.guestuser.2.0+xml

<?xml version="1.0" encoding="UTF-8"?>
<ns3:operationAdditionalData xmlns:ns2="identity.ers.ise.cisco.com"
xmlns:ns3="ers.ise.cisco.com">
    <requestAdditionalAttributes>
        <additionalAttribute name="reason" value="AUP not accepted"/>
    </requestAdditionalAttributes>
</ns3:operationAdditionalData>
```

响应

```
HTTP/1.1 204 No Content
Sat, 15 Dec 2012 10:14:38 GMT
```

恢复访客用户

使用 PUT 操作可恢复暂停的访客用户帐户。

表 7-36　　恢复访客用户的主要特征

说明	恢复指定的访客用户
摘要	PUT /ers/config/guestuser/reinstate/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	访客用户类型的资源
响应状态	204、400、401、403、404、415、500

恢复访客用户示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/reinstate/33
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2012 10:20:48 GMT
```

向访客用户发送邮件

使用 PUT 操作可向访客用户的邮件帐户发送邮件。此操作需要在 Cisco ISE 中配置 SMTP 服务器。请求需要使用门户 ID，因为门户配置包含邮件正文和主题所需的信息。

表 7-37　向访客用户发送邮件的主要特征

说明	向指定访客用户发送邮件
摘要	PUT /ers/config/guestuser/email/{id}/portalId/{portalID}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	发件人邮件
响应头	Content-Length、Content-Type
响应消息正文	不适用
响应状态	204、400、401、403、404、415、500

向访客用户发送邮件示例

请求

```
PUT
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/email/4444/portalId/ff2d99e0-2101-11e4-
b5cf-005056bf2f0a
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept:
    application/vnd.com.cisco.ise.identity.guestuser.2.0+xml

<?xml version="1.0" encoding="UTF-8"?>
 <ns3:operationAdditionalData xmlns:ns2="identity.ers.ise.cisco.com"
xmlns:ns3="ers.ise.cisco.com">
       <requestAdditionalAttributes>
           <additionalAttribute name="senderEmail" value="sender Email"/>
       </requestAdditionalAttributes>
</ns3:operationAdditionalData>
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2012 10:20:48 GMT
```

向访客用户发送 SMS 文本

使用 PUT 操作可向访客用户的移动电话发送文本消息。此操作需要在 Cisco ISE 中配置 SMTP 服务器。

请求需要使用门户 ID，因为门户配置包含文本正文所需的信息。

表 7-38　向访客用户发送邮件的主要特征

说明	向指定访客用户发送 SMS。
摘要	PUT /ers/config/guestuser/sms/{id}/portalId/{portalID}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	不适用
响应状态	204、400、401、403、404、415、500

发送 SMS 示例

请求

```
PUT
https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/sms/444/portalId/ff2d99e0-2101-11e4-b5cf-005056bf2f0a
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept:
    application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2012 10:20:48 GMT
```

批准访客用户

此操作允许您批准访客用户帐户。此操作需要使用访客帐户 ID。

表 7-39　获取 API 版本的主要特征

说明	批准指定的访客用户
摘要	PUT /ers/config/guestuser/approve/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	API 版本
响应状态	200, 400, 401, 403, 404, 415, 500

批准访客用户示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/approve/3333
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2012 10:20:48 GMT
```

拒绝批准访客用户帐户

此操作允许您拒绝批准访客用户帐户。此操作需要使用访客帐户 ID。

表 7-40　　获取 API 版本的主要特征

说明	拒绝指定的访客用户
摘要	PUT /ers/config/guestuser/deny/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	API 版本
响应状态	200、400、401、403、404、415、500

拒绝批准访客用户示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/deny/7777
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2012 10:20:48 GMT
```

重置访客用户帐户的密码

此操作允许您重置访客用户帐户的密码。此操作需要使用访客帐户 ID。此操作会返回生成的新密码，您无法使用 REST API 指定自己的密码。

表 7-41　获取 API 版本的主要特征

说明	重置指定访客用户的密码
摘要	PUT /ers/config/guestuser/resetpassword/{id}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	新密码
响应状态	200, 400, 401, 403, 404, 415, 500

重置访客用户的密码示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/resetpassword/7777
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2014 10:20:48 GMT
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:operationResult xmlns:ns2="ers.ise.cisco.com">
    <attributesList>
        <attribute value="DdsAASDs%$##@ssds12" name="password"/>
    </attributesList>
</ns2:operationResult>
```

启动访客用户的批量执行

通过批量操作请求，您可以在单个请求中发送最多 500 项操作，或根据 ID 发送最多 5000 项操作。

如果请求有效，服务器将返回状态代码 `202 (ACCEPTED)`，并在 LOCATION 响应头中包含唯一的批量标识符，您可以使用此标识符通过获取批量状态操作跟踪批量状态。

一次只允许运行一个批量操作。如果在另一个批量操作仍在执行时发布批量操作请求，服务器将返回响应状态 `503 (Service Unavailable)`，并包含相应的要求客户端稍后重试的描述性消息。

表 7-42　启动批量执行的主要特征

说明	启动执行
摘要	PUT /ers/config/guestuser/bulk
请求头	Accept、Authorization、Host
查询字符串	不适用

表 7-42　　启动批量执行的主要特征

请求消息正文	批量请求
响应头	Content-Length、Content-Type
响应消息正文	不适用
响应状态	202、400、401、403、404、415、500

创建访客批量执行示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/bulk
Authorization: Basic
Content-Type: application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:bulkRequest xsi:type="ns2:guestUserBulkRequest"
                 resourceMediaType="vnd.com.cisco.ise.identity.guestuser.1.0+xml"
                 operationType="create"
                 xmlns:ns2="identity.ers.ise.cisco.com"
     xmlns:ns3="ers.ise.cisco.com">
 <resourcesList>
  <resource xsi:type="ns2:GuestUser" description="created by bulk">
  <portalId>6ab68890-d0f1-11e3-a1d5-005056bf4687</portalId>
   <guestAccesstInfo>
        <groupTag>group</groupTag>
        <validDays>2</validDays>
        <location>London</location>
        <ssid>guest_ssid</ssid>
    </guestAccesstInfo>
<guestInfo>
        <company>new company</company>
        <emailAddress>joe@example.com</emailAddress>
        <enabled>true</enabled>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
        <phoneNumber>6033203311</phoneNumber>
        <userName>lucky7</userName>
        <password>1234</password>
        <notificationLanguage>English</notificationLanguage>
        <smsServiceProvider>ATT</smsServiceProvider>
    </guestInfo>
    <guestType>DAILY</guestType>
    <reasonForVisit>interview</reasonForVisit>
    <personBeingVisited>sponsor@cisco.com</personBeingVisited>
  </resource>

...
  <resource xsi:type="ns2:GuestUser" description="created by bulk">
  <portalId>6ab68890-d0f1-11e3-a1d5-005056bf4687</portalId>
   <guestAccesstInfo>
        <groupTag>group</groupTag>
        <validDays>3</validDays>
        <location>London</location>
        <ssid>guest_ssid</ssid>
    </guestAccesstInfo>
    <guestInfo>
        <company>new company</company>
        <emailAddress>mary@example.com</emailAddress>
        <enabled>true</enabled>
```

```
        <firstName>Mary</firstName>
        <lastName>Sue</lastName>
        <phoneNumber>6039990000</phoneNumber>
        <userName>lucky13</userName>
        <password>1234</password>
        <notificationLanguage>English</notificationLanguage>
        <smsServiceProvider>ATT</smsServiceProvider>
    </guestInfo>
    <guestType>DAILY</guestType>
    <reasonForVisit>interview</reasonForVisit>
    <personBeingVisited>sponsor@cisco.com</personBeingVisited>
  </resource>
 </resourcesList>
</ns3:bulkRequest>
```

响应

```
HTTP/1.1 202 ACCEPTED
        Date: Thu, 12 Jul 2012 23:59:59 GMT
        Location: https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/123443545334
```

相关主题

获取终端批量状态，第 7-15 页

获取访客用户的批量状态

如果批量执行请求有效且没有其他正在处理的批量操作，服务器将在 LOCATION 响应头中返回唯一的批量标识符。使用此 ID 可跟踪批量状态，您可以获取操作开始之后至少 2 小时的状态报告。

表 7-43　　获取批量状态的主要特征

说明	监控指定的批量执行进度
摘要	GET /ers/config/guestuser/bulk/{bulkid}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	批量状态
响应状态	200、400、401、403、404、415、500

获取访客用户的批量状态示例

请求

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/bulk/53454354534 HTTP/1.1
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.guestuserbulkrequest.1.0+xml
```

响应

```
HTTP/1.1 200 OK
Date: Thu Mar 07 18:17:35 IST 2013 GMT
Content-Type: application/vnd.com.cisco.ise.ers.guestuserbulkrequest.1.0+xml
Content-Length: 16347
```

```
{
<ns2:bulkStatus
  xmlns:ns2 = "ers.ise.cisco.com"
  successCount = "50"
  startTime = "Thu Mar 07 17:17:35 IST 2013"
  resourcesCount = "50"
  operationType = "create"
  resourceMediaType = "vnd.com.cisco.ise.ers.identity.guestuser.1.0+xml"
  failCount = "0"
  executionStatus = "COMPLETED"
  bulkId = "53454354534">

  <resourcesStatus>
    <resourceStatus
     status = "SUCCUESS"
     description = "created by bulk request"
     id = "23d068d0-873a-11e2-bad4-00215edbb2a8"/>

...

    <resourceStatus
     status = "SUCCUESS"
     description = "created by bulk request"
     id = "23cfa580-873a-11e2-bad4-00215edbb2a8"/>
  </resourcesStatus>
</ns2:bulkStatus>
}
```

更改发起人的密码

此操作允许您更改当前登录的发起人的密码。此操作需要使用门户 ID。

表 7-44　　获取 API 版本的主要特征

说明	更新已登录发起人的密码
摘要	PUT /ers/config/guestuser/changeSponsorPassword/{portalId}
请求头	Accept、Authorization、Host
查询字符串	不适用
请求消息正文	不适用
响应头	Content-Length、Content-Type
响应消息正文	API 版本
响应状态	200、400、401、403、404、415、500

更改发起人的密码示例

请求

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/guestuser/changeSponsorPassword/88888
    Host: cisco.com
    Authorization: Basic xxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.identity.guestuser.2.0+xml

<?xml version="1.0" encoding="UTF-8"?>
```

```
<ns3:operationAdditionalData xmlns:ns2="identity.ers.ise.cisco.com"
xmlns:ns3="ers.ise.cisco.com">
    <requestAdditionalAttributes>
        <additionalAttribute name="newPassword" value="Cisco1234"/>
        <additionalAttribute name="currentPassword" value="Autom8me"/>
    </requestAdditionalAttributes>
</ns3:operationAdditionalData>
```

响应

```
HTTP/1.1 204 OK
Date: Sat, 15 Dec 2012 10:20:48 GMT
```

适用于门户的外部 RESTful 服务 API

下表列出适用于门户的外部 RESTful 服务 API：

表 7-45　　适用于门户的 API

操作	方法	URL	内容	查询字符串
获取所有门户	GET	/ers/config/portal	不适用	Page、Size、sortacs 或 sortdsn、Filter
通过 ID 获取门户	GET	/ers/config/portal/{id}	不适用	

获取所有门户

下表列出获取所有门户 API 调用的主要特征：

表 7-46　　获取所有门户 API 调用的主要特征

说明	检索门户的集合
摘要	GET /ers/config/portal
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415,500

获取所有门户调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/portal
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.portal.1.0+xml
```

获取所有门户调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="utf-8" standalone="yes"?> <ns2:searchResult total="2"
xmlns:ns2="ers.ise.cisco.com">
        <resources>
            <resource name="portal1" id="id1">
                <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/portal/id1" rel="self"/>
            </resource>
            <resource name="portal2" id="id2">
                <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/portal/id2" rel="self"/>
            </resource>
        </resources>
</ns2:searchResult>
```

通过 ID 获取门户

下表列出通过 ID 获取门户 API 调用的主要特征：

表 7-47　　通过 ID 获取门户 API 调用的主要特征

说明	检索指定的门户
摘要	GET /ers/config/portal/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	门户资源类型
响应状态	200、400、401,403、404、415、500

通过 ID 获取门户调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/portal/333
    Authorization: Basic xxxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.portal.1.0+xml
```

通过 ID 获取门户调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.identity.portal.1.0+xml Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:portal name="sponsor" id="d7b703f0-b073-11e3-bd6c- 005056a15fa7"
        xmlns:ns2="ers.ise.cisco.com"
        xmlns:ns3="identity.ers.ise.cisco.com">
```

```
        <link type="application/xml"
href="https://<ISE-ADMIN-NODE>:9060/ers/config/portal/333" rel="self"/>
    <allowSponsorToChangeOwnPassword>false</allowSponsorToChangeOwnPassword>
        <GuestUserFieldList>
            <GuestUserField>
                <customType>false</customType>
                <dataType>DROPDOWN</dataType>
    <dictionaryLabelKey>ui_sms_provider_label</dictionaryLabelKey>
                <labelName>SMS Service Provider</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>TEXT</dataType>
    <dictionaryLabelKey>ui_company_label</dictionaryLabelKey>
                <labelName>Company</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>TEXT</dataType>
    <dictionaryLabelKey>ui_first_name_label</dictionaryLabelKey>
                <labelName>First name</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>TEXT</dataType>
    <dictionaryLabelKey>ui_reason_visit_label</dictionaryLabelKey>
                <labelName>Reason for visit</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>true</customType>
                <dataType>TEXT</dataType>
    <dictionaryLabelKey>ui_ssn-number_text_label</dictionaryLabelKey>
                <instructionText>social </instructionText>
                <labelName>ssn-number</labelName>
                <required>false</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>PHONE</dataType>
    <dictionaryLabelKey>ui_phone_number_label</dictionaryLabelKey>
                <labelName>Phone number</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>EMAIL</dataType>
    <dictionaryLabelKey>ui_person_visited_label</dictionaryLabelKey>
                <labelName>Person being visited</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>EMAIL</dataType>
    <dictionaryLabelKey>ui_email_address_label</dictionaryLabelKey>
                <labelName>Email address</labelName>
                <required>true</required>
            </GuestUserField>
            <GuestUserField>
                <customType>false</customType>
                <dataType>TEXT</dataType>
```

```
    <dictionaryLabelKey>ui_last_name_label</dictionaryLabelKey>
                <labelName>Last name</labelName>
                <required>true</required>
            </GuestUserField>
    </GuestUserFieldList>
</ns3:portal>
}
```

适用于配置文件的外部 RESTful 服务 API

下表列出适用于配置文件的外部 RESTful 服务 API：

表 7-48　　适用于门户的 API

操作	方法	URL	内容	查询字符串
获取所有配置文件	GET	/ers/config/profilerprofile	不适用	Page、Size、sortacs 或 sortdsn、Filter
通过 ID 获取配置文件	GET	/ers/config/profilerprofile/{id}	不适用	

获取所有配置文件

下表列出获取所有门户 API 调用的主要特征：

表 7-49　　获取所有门户 API 调用的主要特征

说明	检索配置文件的集合
摘要	GET /ers/config/profilerprofile
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415,500

获取所有配置文件调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/profilerprofile
    Authorization: Basic xxxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.profilerprofile.1.0+xml
```

获取所有配置文件调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2014 23:59:59 GMT

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult total="2" xmlns:ns2="ers.ise.cisco.com">
    <nextPage type="application/xml" href="link-to-next-page" rel="next"/>
    <previousPage type="application/xml" href="link-to-previous-page" rel="previous"/>
    <resources>
        <resource name="name1" id="id1" description="description1"/>
        <resource name="name2" id="id2" description="description2"/>
    </resources>
</ns2:searchResult>
```

通过 ID 获取门户

下表列出通过 ID 获取配置文件 API 调用的主要特征：

表 7-50　　通过 ID 获取门户 API 调用的主要特征

说明	检索指定的配置文件
摘要	GET /ers/config/profilerprofile/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	配置文件资源类型
响应状态	200、400、401,403、404、415、500

通过 ID 获取门户调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/profilerprofile/333
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.identity.profilerprofile.1.0+xml
```

通过 ID 获取门户调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2014 23:59:59 GMT

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:profilerprofile name="name" id="id" description="description"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com"/
```

适用于网络设备的外部 RESTful 服务 API

下表列出适用于网络设备的外部 RESTful 服务 API：

表 7-51　　适用于门户的 API

操作	方法	URL	内容	查询字符串
获取所有网络设备	GET	/ers/config/networkdevice	不适用	Page、Size、sortacs 或 sortdsn、Filter
获取网络设备	GET	/ers/config/networkdevice /{id}	不适用	
创建网络设备	POST	/ers/config/networkdevice	网络设备	
更新网络设备	PUT	/ers/config/networkdevice /{id}	网络设备	
删除网络设备	DELETE	/ers/config/networkdevice /{id}	不适用	
获取网络设备资源版本信息	GET	/ers/config/ networkdevice /versioninfo	不适用	

获取所有网络设备

下表列出获取所有网络设备 API 调用的主要特征：

表 7-52　　获取所有网络设备 API 调用的主要特征

说明	检索网络设备资源的集合
摘要	GET /ers/config/networkdevice
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、500

获取所有网络设备调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevice?page=1&size=20&sortacs=name
    Authorization: Basic xxxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml
```

获取所有网络设备调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <ns2:searchResult
        xmlns:ns2="ers.ise.cisco.com" total="1">
        <resources>
            <resource name="nd1" id="0d008bb0-2539-11e3-84ad-
    00215edbb2a8">
                <link type="application/xml"
    href="https://10.56.13.196:9060/ers/config/networkdevice/0d0
    08bb0-2539-11e3-84ad-00215edbb2a8" rel="self"/>
            </resource>
        </resources>
    </ns2:searchResult>
}
```

通过 ID 获取网络设备

下表列出通过 ID 获取网络设备 API 调用的主要特征：

表 7-53　　通过 ID 获取网络设备 API 调用的主要特征

说明	检索指定的网络设备
摘要	GET /ers/config/networkdevice/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	网络设备类型的资源
响应状态	200、400、401、403、404、415、500

通过 ID 获取网络设备调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevice/333
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml
```

通过 ID 获取网络设备调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml Content-Length:
16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <ns3:networkdevice
        xmlns:ns2="ers.ise.cisco.com"
        xmlns:ns3="network.ers.ise.cisco.com" name="nd1"
    id="0d008bb0-2539-11e3-84ad-00215edbb2a8">
        <link type="application/xml"
    href="https://10.56.13.196:9060/ers/config/networkdevice/0d0
```

```
    08bb0-2539-11e3-84ad-00215edbb2a8" rel="self"/>
        <authenticationSettings>
            <enableKeyWrap>false</enableKeyWrap>
            <keyInputFormat>ASCII</keyInputFormat>
            <networkProtocol>RADIUS</networkProtocol>
            <radiusSharedSecret>*****</radiusSharedSecret>
        </authenticationSettings>
<NetworkDeviceIPList>
        <NetworkDeviceIP>
            <ipaddress>1.2.3.4</ipaddress>
            <mask>32</mask>
        </NetworkDeviceIP>
    </NetworkDeviceIPList>
    <modelName>Unknown</modelName>
    <NetworkDeviceGroupList>
        <NetworkDeviceGroup>1d8c62b0-2539-11e3-84ad-
00215edbb2a8</NetworkDeviceGroup>
        <NetworkDeviceGroup>37053aa0-2539-11e3-84ad-
00215edbb2a8</NetworkDeviceGroup>
    </NetworkDeviceGroupList>
    <softwareVersion>Unknown</softwareVersion>
</ns3:networkdevice>
}
```

创建网络设备

下表列出创建网络设备 API 调用的主要特征：

表 7-54　　创建网络设备 API 调用的主要特征

说明	创建指定的网络设备
摘要	POST /ers/config/networkdevice/
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	网络设备
响应头	Content-Length、Content-Type、Location
响应消息正文	N/A
响应状态	200、400、401、403、415、500

创建网络设备调用的请求示例

```
POST https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevice/
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml {
    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <ns3:networkdevice
        xmlns:ns2="ers.ise.cisco.com"
        xmlns:ns3="network.ers.ise.cisco.com" name="nd2">
        <authenticationSettings>
            <enableKeyWrap>false</enableKeyWrap>
            <keyInputFormat>ASCII</keyInputFormat>
            <networkProtocol>RADIUS</networkProtocol>
            <radiusSharedSecret>acsi</radiusSharedSecret>
        </authenticationSettings>
        <NetworkDeviceIPList>
```

```
            <NetworkDeviceIP>
                <ipaddress>1.2.3.4</ipaddress>
                <mask>32</mask>
            </NetworkDeviceIP>
        </NetworkDeviceIPList>
        <modelName>Unknown</modelName>
        <NetworkDeviceGroupList>
            <NetworkDeviceGroup>1d8c62b0-2539-11e3-84ad-
    00215edbb2a8</NetworkDeviceGroup>
            <NetworkDeviceGroup>37053aa0-2539-11e3-84ad-
    00215edbb2a8</NetworkDeviceGroup>
        </NetworkDeviceGroupList>
        <softwareVersion>Unknown</softwareVersion>
    </ns3:networkdevice>
}
```

创建网络设备调用的响应示例

```
HTTP/1.1 201 OK (see location header for the ID of the new device)
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml
Location: https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevice/444
```

更新网络设备

下表列出更新网络设备 API 调用的主要特征：

表 7-55　　更新网络设备 API 调用的主要特征

说明	更新指定的网络设备
摘要	PUT /ers/config/networkdevice/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	网络设备
响应头	Content-Length、Content-Type
响应消息正文	更新字段的列表
响应状态	200、400、401、403、415、500

更新网络设备调用的请求示例

```
PUT https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevice/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Content-Type: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml {
    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <ns3:networkdevice
        xmlns:ns2="ers.ise.cisco.com"
        xmlns:ns3="network.ers.ise.cisco.com"
    name="nd2_updated">
        <authenticationSettings>
            <enableKeyWrap>true</enableKeyWrap>
        </authenticationSettings>
    </ns3:networkdevice>
}
```

更新网络设备调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.updatedfields.1.0+xml Content-Length: 529
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <ns2:updatedFields
xmlns:ns2="ers.ise.cisco.com">
    <updatedField field="name">
        <newValue>nd2_updated</newValue>
        <oldValue>nd2</oldValue>
    </updatedField>
    <updatedField field="enableKeywrap">
        <newValue>true</newValue>
        <oldValue>false</oldValue>
    </updatedField>
</ns2:updatedFields>
}
```

删除网络设备

下表列出删除网络设备 API 调用的主要特征：

表 7-56　　删除网络设备 API 调用的主要特征

说明	删除指定的网络设备
摘要	DELETE /ers/config/networkdevice/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	N/A
响应状态	200、204、400、401、403、404、415、500

更新网络设备调用的请求示例

```
DELETE https://<ISE-ADMIN-NODE>:9060/ers/config/networjdevice/333
    Authorization: Basic xxxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.network.networkdevice.1.0+xml
```

更新网络设备调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
```

适用于网络设备组的外部 RESTful 服务 API

下表列出适用于网络设备组的外部 RESTful 服务 API：

表 7-57　　适用于 SGT 的 API

操作	方法	URL	内容	查询字符串
获取所有网络设备组	GET	/ers/config/networkdevicegroup	不适用	page、size、sortacs 或 sortdsn、filter
获取网络设备组	GET	/ers/config/networkdevicegroup/{id}	不适用	
获取网络设备组资源版本信息	GET	/ers/config/networkdevicegroup/versioninfo	不适用	

获取所有网络设备组

下表列出获取所有网络设备组 API 调用的主要特征：

表 7-58　　获取所有网络设备组 API 调用的主要特征

说明	检索网络设备组资源的集合
摘要	GET /ers/config/networkdevicegroup
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、500

获取所有网络设备组 API 调用的请求示例

```
GET
https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevicegroup?page=1&size=20&sortacs=name
    Authorization: Basic xxxxxxxxxxxxxxxxxx
    Accept: application/vnd.com.cisco.ise.network.networkdevicegroup.1.0+xml
```

获取所有网络设备组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml Content-Length: 16347
{
   <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
                      <ns2:searchResult
            xmlns:ns2="ers.ise.cisco.com" total="0">
                          <resources>
  <resource name="Location#All Locations#loc1" id="1d8c62b0-2539-11e3-84ad-00215edbb2a8"
description="xxx">
```

```
                <link type="application/xml"
href="https://10.56.13.196:9060/ers/config/networkdevicegroup/1d8c62b0-2539-11e3-84ad-0021
5edbb2a8" rel="self"/>
                        </resource>
   <resource name="Device Type#All Device Types#device type 555"
id="37053aa0-2539-11e3-84ad-00215edbb2a8" description="vvv">
                <link type="application/xml"
 href="https://10.56.13.196:9060/ers/config/networkdevicegrou
     p/37053aa0-2539-11e3-84ad-00215edbb2a8" rel="self"/>
                        </resource>
                        </resources>
                    </ns2:searchResult>
}
```

获取网络设备组

下表列出获取网络设备组 API 调用的主要特征：

表 7-59　　获取网络设备组 API 调用的主要特征

说明	检索指定的网络设备组
摘要	GET /ers/config/networkdevicegroup/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	网络设备组类型的资源
响应状态	200、400、401、403、404、415、429、500

获取网络设备组 API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/networkdevicegroup/333
Authorization: Basic xxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.network.networkdevicegroup.1.0+xml
```

获取网络设备组 API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.network.networkdevicegroup.1.0 +xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?> <ns3:networkdevicegroup
        xmlns:ns2="ers.ise.cisco.com"
        xmlns:ns3="network.ers.ise.cisco.com" name="Location#All
    Locations#loc1" id="1d8c62b0-2539-11e3-84ad-00215edbb2a8"
    description="xxx">
        <link type="application/xml"
    href="https://10.56.13.196:9060/ers/config/networkdevicegrou
    p/1d8c62b0-2539-11e3-84ad-00215edbb2a8" rel="self"/>
        <type>Location</type>
</ns3:networkdevicegroup>
}
```

适用于 SGT 的外部 RESTful 服务 API

下表列出适用于 SGT 的外部 RESTful 服务 API：

表 7-60　　适用于 SGT 的 API

操作	方法	URL	内容	查询字符串
获取所有 SGT	GET	/ers/config/sgt	不适用	page、size、sortacs 或 sortdsn、filter
获取 SGT	GET	/ers/config/sgt/{id[1]}	不适用	
获取 GST 资源版本信息	GET	/ers/config/sgt/versioninfo	不适用	

1. SGT ID 是 Cisco ISE 数据库中存储的 UUID 类型。

获取所有 SGT

下表列出获取所有 SGT API 调用的主要特征：

表 7-61　　主获取所有 SGT API 调用的主要特征

说明	检索 SGT 资源的集合
摘要	GET /ers/config/sgt
请求头	Accept、Authorization、Host
查询字符串	page、size、sortbyacn、sortbydcn、filter
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	搜索结果
响应状态	200、400、401、403、404、415、429、500

获取所有 SGT API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/sgt?page=0&size=20&sortacs=name
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.sga.sgt.1.0+xml
```

获取所有 SGT API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.ers.searchresult.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:searchResult total="2" xmlns:ns2="ers.ise.cisco.com">
<resources>
  <resource name="name1" id="id1" description="description1">
```

```
   <link type="application/xml" href="https://<ISE-ADMIN-NODE>:9060/ers/config/sgt/id1"
rel="self"/>
  </resource>
 </resources>
</ns2:searchResult>
}
```

通过 ID 获取 SGT

下表列出通过 ID 获取 SGT API 调用的主要特征：

表 7-62　　获取 SGT API 调用的主要特征

说明	检索指定的 SGT
摘要	GET /ers/config/sgt/{id}
请求头	Accept、Authorization、Host
查询字符串	N/A
请求消息正文	N/A
响应头	Content-Length、Content-Type
响应消息正文	内部用户类型的资源
响应状态	200、400、401、403、404、415、429、500

通过 ID 获取 SGT API 调用的请求示例

```
GET https://<ISE-ADMIN-NODE>:9060/ers/config/sgt/333
Authorization: Basic xxxxxxxxxxxxxxxxxxx
Accept: application/vnd.com.cisco.ise.sga.sgt.1.0+xml
```

通过 ID 获取 SGT API 调用的响应示例

```
HTTP/1.1 200 OK
Date: Thu, 12 Jul 2012 23:59:59 GMT
Content-Type: application/vnd.com.cisco.ise.sga.sgt.1.0+xml
Content-Length: 16347
{
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:sgt description="description" name="name" id="id" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="sga.ers.ise.cisco.com">
    <generationId>generationId</generationId>
    <isTagFromRange>isTagFromRange</isTagFromRange>
    <value>1</value>
</ns3:sgt>
}
```

REST API 客户端

通过外部 RESTful 服务 API，您可以对 Cisco ISE 资源执行 CRUD（创建、读取、更新、删除）操作。要使用与 Cisco ISE 服务器通信并在其上执行操作的外部 RESTful 服务 API 构建和测试应用，您可以使用任何行业标准 REST API 客户端，如 Google Chrome 的 POSTMAN 插件。

POSTMAN 根据 REST 架构和基本原则进行设计，使您可以利用 Google Chrome Web 浏览器发送并检索标准 HTTP 和 HTTPS 请求和响应。您可以使用以下标准 HTTP 方法对 Cisco ISE 资源执行 CRUD 操作：

- GET
- POST
- PUT
- DELETE

通过 ERS API，您可以在各种 API 调用中使用这些 HTTP 请求，从而使您可以在 Cisco ISE 服务器上执行操作。有关使用这些 HTTP 请求的操作的完整列表，请参阅 <ERS API 操作 >。

注意 要下载 POSTMAN 插件，请访问 https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm?hl=en。有关使用 POSTMAN 插件的详细信息，请访问 https://github.com/a85/POSTMan-Chrome-Extension/wiki。

GET 方法

请求指定资源的表示。使用 GET 的请求只会检索数据，不会产生任何其他影响。

注意 本节介绍如何使用 POSTMAN 插件发起 ERS API 调用。此 API 调用使用 GET HTTP 方法和 ERS API 的其他部分（本节不做介绍）。有关各个 ERS API 部分（如特征、请求和响应）的详细信息，请参阅外部 RESTful 服务 API 操作。

使用 GET HTTPS 方法的 ERS API 调用的请求正文包括以下三个构成要素：

- URI
- Accept 标头
- Authorization 标头

URI

GET 方法向 Cisco ISE 服务器发送 URI，HTTP 回复为原始结果数据。典型的 URI 必须遵守以下格式：

- *https://<Cisco ISE Server address:<port>/<namespace>/config/<Cisco ISE Resouce Name>*

  其中，*<Cisco ISE Server Address>* 表示 Cisco ISE 服务器的服务器地址，*<port>* 表示端口 9060，*<namespace>* 表示 ISE 资源所属的命名空间，*<Cisco ISE Resource Name>* 表示 Cisco ISE 资源的名称。

以下示例显示的 URI 用于请求 *interaluser* ISE 资源的数据：

- *https://10.56.13.196:9060/ers/config/internaluser*。

注意　URI 不是请求正文；它只是一个 URL。此 URL 使用 GET 方法发送到服务器。

Accept 标头

Accept 标头必须遵守以下格式：

- *application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major version>.<minor version>+xml*

其中，*<resource-namespace>* 表示 ISE 资源所属的命名空间，*<resource-type>* 表示 ISE 资源的类型，*<major-version>* 表示 ISE 部署的主要版本号，*<minor-version>* 表示 ISE 部署的次要版本号。

以下示例显示典型的 Accept 标头：

- *application/vnd.com.cisco.ise.identity.internaluser.1.0+xml*

Authorization 标头

Authorization 标头包含嵌入到 GET 请求中的加密授权密钥。指定授权凭证后，您必须生成加密密钥，此加密密钥之后将嵌入到请求正文中。

注意　有关生成加密密钥的详细信息，请参阅使用 POSTMAN 提出 GET 请求，第 7-58 页。

使用 POSTMAN 提出 GET 请求

操作步骤

步骤 1　在 Google Chrome 浏览器中打开 POSTMAN 插件。

步骤 2　使用左侧窗格中的选项创建新集合。

注意　有关使用 POSTMAN 插件的详细信息，请访问 https://github.com/a85/POSTMan-Chrome-Extension/wiki。

步骤 3　从下拉菜单中，选择 **GET**。

步骤 4　在 URL 栏，输入 URI。

URI 指定您尝试与之通信的 Cisco ISE 服务器和您尝试访问的 ISE 资源。有关 URI 格式的详细信息，请参阅 URI，第 7-57 页。

步骤 5　点击 **Basic Auth** 选项卡。

通过此选项，您可以指定显示的用户访问凭证。

步骤 6　在 Username 和 Password 字段指定您的访问凭证，然后点击 **Refresh Headers**。

POSTMAN 会显示 Authorization 标头与加密密钥。

步骤 7　通过指定以下值添加 Accept 标头：application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

注意　有关 Accept 标头的详细信息，请参阅 Accept 标头，第 7-58 页。

步骤 8　点击 **Send**。

POSTMAN 插件会显示 200 OK 状态响应，表示请求成功。此请求也会返回您在 URL 指定的资源的详细信息。

POST 方法

请求服务器接受请求中包含的实体作为 URI 所标识 Web 资源的新附属资源。

注意　本节介绍如何使用 POSTMAN 插件发起 ERS API 调用。此 API 调用使用 POST HTTP 方法和 ERS API 的其他部分（本节不做介绍）。有关各个 ERS API 部分（如特征、请求和响应）的详细信息，请参阅外部 RESTful 服务 API 操作。

使用 POST HTTP 方法的 ERS API 调用的请求正文包含以下三个构成要素：

- URI
- Content-Type 标头
- Authorization 标头

URI

POST 方法向 Cisco ISE 服务器发送 URI。典型的 URI 必须遵守以下格式：

- *https://<Cisco ISE Server address:<port>/<namespace>/config/<Cisco ISE Resouce Name>*

  其中，*<Cisco ISE Server Address>* 表示 Cisco ISE 服务器的服务器地址，*<port>* 表示端口 9060，*<namespace>* 表示 ISE 资源所属的命名空间，*<Cisco ISE Resource Name>* 表示 Cisco ISE 资源的名称。

以下示例显示的 URI 用于请求 *interaluser* ISE 资源的数据：

- *https://10.56.13.196:9060/ers/config/internaluser*。

注意　URI 不是请求正文；它只是一个 URL。此 URL 使用 POST 方法发送到服务器。

Content-Type 标头

Content-Type 标头必须遵守以下格式：

- *application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major version>.<minor version>+xml*

  其中，*<resource-namespace>* 表示 ISE 资源所属的命名空间，*<resource-type>* 表示 ISE 资源的类型，*<major-version>* 表示 ISE 部署的主要版本号，*<minor-version>* 表示 ISE 部署的次要版本号。

以下示例显示典型的 Accept 标头：

- *application/vnd.com.cisco.ise.identity.internaluser.1.0+xml*

Authorization 标头

Authorization 标头包含嵌入到 POST 请求中的加密授权密钥。指定授权凭证后，您必须生成加密密钥，此加密密钥之后将嵌入到请求正文中。

注意　有关生成加密密钥的详细信息，请参阅使用 POSTMAN 提出 POST 请求，第 7-60 页。

使用 POSTMAN 提出 POST 请求

操作步骤

步骤 1　在 Google Chrome 浏览器中打开 POSTMAN 插件。

步骤 2　使用左侧窗格中的选项创建新集合。

注意　有关使用 POSTMAN 插件的详细信息，请访问 https://github.com/a85/POSTMan-Chrome-Extension/wiki。

步骤 3　从下拉菜单中，选择 **POST**。

步骤 4　在 URI 栏，输入 URI。

URI 指定您尝试与之通信的 Cisco ISE 服务器和您尝试访问的 ISE 资源。有关 URI 格式的详细信息，请参阅URI，第 7-59 页。

步骤 5　点击 **Basic Auth** 选项卡。

通过此选项，您可以指定显示的用户访问凭证。

步骤 6　在 Username 和 Password 字段指定您的访问凭证，然后点击 **Refresh Headers**。

POSTMAN 会显示 Authorization 标头与加密密钥。

步骤 7　通过指定以下值添加 Content-Type 标头：application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

注意　有关 Accept 标头的详细信息，请参阅Content-Type 标头，第 7-59 页。

步骤 8　从显示在 raw 按钮旁边的下拉菜单，选择 **XML**。

步骤 9　点击 **raw**。

步骤 10　POSTMAN 插件会打开一个编辑窗格，您可以利用此窗格指定 POST 请求的正文。

步骤 11　在编辑窗格中输入 POST 请求的消息正文。

注意　此消息正文必须包含您尝试在 ISE 服务器上创建的 ISE 资源对应的详细信息。例如，创建内部 interaluser 时，您必须指定详细信息，如 interaluser 的名称、interaluser 的说明、密码等。有关使用 POST 请求的 ERS API 的消息正文的详细信息，以及需要指定的 ISE 资源的详细信息，请参阅外部 RESTful 服务 API 操作。

步骤 12　点击 **Send**。

POSTMAN 插件会显示 201 CREATED 状态响应，表示请求成功。您可以转到 ISE GUI 以验证您添加的 ISE 资源是否显示在 ISE GUI 中。

PUT 方法

请求包含的实体存储在提供的 URI 下。如果 URI 指向已经存在的现有资源，系统将修改此资源；如果 URI 指向的不是现有资源，服务器可以使用此 URI 创建资源。

注意　本节介绍如何使用 POSTMAN 插件发起 ERS API 调用。此 API 调用使用 PUT HTTP 方法和 ERS API 的其他部分（本节不做介绍）。有关各个 ERS API 部分（如特征、请求和响应）的详细信息，请参阅外部 RESTful 服务 API 操作。

使用 POST HTTP 方法的 ERS API 调用的请求正文包含以下三个构成要素：

- URI
- Content-Type 标头
- Authorization 标头

URI

PUT 方法向 Cisco ISE 服务器发送 URI。典型的 URI 必须遵守以下格式：

- *https://<Cisco ISE Server address:<port>/<namespace>/config/<Cisco ISE Resouce Name>*

  其中，*<Cisco ISE Server Address>* 表示 Cisco ISE 服务器的服务器地址，*<port>* 表示端口 9060，*<namespace>* 表示 ISE 资源所属的命名空间，*<Cisco ISE Resource Name>* 表示 Cisco ISE 资源的名称。

以下示例显示的 URI 用于请求 *interaluser* ISE 资源的数据：

- *https://10.56.13.196:9060/ers/config/internaluser*。

注意　URI 不是请求正文；它只是一个 URL。此 URL 使用 PUT 方法发送到服务器。

Content-Type 标头

Content-Type 标头必须遵守以下格式：

- *application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major version>.<minor version>+xml*

  其中，*<resource-namespace>* 表示 ISE 资源所属的命名空间，*<resource-type>* 表示 ISE 资源的类型，*<major-version>* 表示 ISE 部署的主要版本号，*<minor-version>* 表示 ISE 部署的次要版本号。

以下示例显示典型的 Accept 标头：

- *application/vnd.com.cisco.ise.identity.internaluser.1.0+xml*

Authorization 标头

Authorization 标头包含嵌入到 PUT 请求中的加密授权密钥。指定授权凭证后，您必须生成加密密钥，此加密密钥之后将嵌入到请求正文中。

注意　有关生成加密密钥的详细信息，请参阅使用 POSTMAN 提出 PUT 请求，第 7-62 页。

使用 POSTMAN 提出 PUT 请求

操作步骤

步骤 1　在 Google Chrome 浏览器中打开 POSTMAN 插件。

步骤 2　使用左侧窗格中的选项创建新集合。

注意　有关使用 POSTMAN 插件的详细信息，请访问 https://github.com/a85/POSTMan-Chrome-Extension/wiki。

步骤 3　从下拉菜单中，选择 **PUT**。

步骤 4　在 URI 栏，输入 URI。

URI 指定您尝试与之通信的 Cisco ISE 服务器和您尝试访问的 ISE 资源。有关 URI 格式的详细信息，请参阅 URI，第 7-61 页。

步骤 5　点击 **Basic Auth** 选项卡。

通过此选项，您可以指定显示的用户访问凭证。

步骤 6　在 Username 和 Password 字段指定您的访问凭证，然后点击 **Refresh Headers**。

POSTMAN 会显示 Authorization 标头与加密密钥。

步骤 7　通过指定以下值添加 Content-Type 标头：application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

注意　有关 Accept 标头的详细信息，请参阅 Content-Type 标头，第 7-61 页。

步骤 8　从显示在 raw 按钮旁边的下拉菜单，选择 **XML**。

步骤 9　点击 **raw**。

步骤 10　POSTMAN 插件会打开一个编辑窗格，您可以利用此窗格指定 POST 请求的正文。

步骤 11　在编辑窗格中输入 POST 请求的消息正文。

注意　此消息正文必须包含您尝试在 ISE 服务器上更新的 ISE 资源对应的详细信息。例如，更新 interaluser 时，必须指定详细信息，如 interaluser 的名称、interaluser 的说明、密码等。有关使用 POST 请求的 ERS API 的消息正文的详细信息，以及需要指定的 ISE 资源的详细信息，请参阅外部 RESTful 服务 API 操作。

步骤 12　点击 **Send**。

POSTMAN 插件会显示 201 CREATED 状态响应，表示请求成功。您可以转到 ISE GUI 以验证您添加的 ISE 资源是否显示在 ISE GUI 中。

Delete 方法

删除指定资源。

注意　本节介绍如何使用 POSTMAN 插件发起 ERS API 调用。此 API 调用使用 DELETE HTTP 方法和 ERS API 的其他部分（本节不做介绍）。有关各个 ERS API 部分（如特征、请求和响应）的详细信息，请参阅外部 RESTful 服务 API 操作。

使用 DELETE HTTP 方法的 ERS API 调用的请求正文包含以下三个构成要素：

- URI
- Accept 标头
- Authorization 标头

URI

DELETE 方法向 Cisco ISE 服务器发送 URI。典型的 URI 必须遵守以下格式：

- *https://<Cisco ISE Server address:<port>/<namespace>/config/<Cisco ISE Resouce Name>*

  其中，*<Cisco ISE Server Address>* 表示 Cisco ISE 服务器的服务器地址，*<port>* 表示端口 9060，*<namespace>* 表示 ISE 资源所属的命名空间，*<Cisco ISE Resource Name>* 表示 Cisco ISE 资源的名称。

以下示例显示的 URI 用于请求 *interaluser* ISE 资源的数据：

- *https://10.56.13.196:9060/ers/config/internaluser*。

注意　URI 不是请求正文；它只是一个 URL。此 URL 使用 GET 方法发送到服务器。

Accept 标头

Accept 标头必须遵守以下格式：

- *application/vnd.com.cisco.ise.<resource-namespace>.<resource-type>.<major version>.<minor version>+xml*

其中，*<resource-namespace>* 表示 ISE 资源所属的命名空间，*<resource-type>* 表示 ISE 资源的类型，*<major-version>* 表示 ISE 部署的主要版本号，*<minor-version>* 表示 ISE 部署的次要版本号。

以下示例显示典型的 Accept 标头：

- *application/vnd.com.cisco.ise.identity.internaluser.1.0+xml*

Authorization 标头

Authorization 标头包含嵌入到 DELETE 请求中的加密授权密钥。指定授权凭证后，您必须生成加密密钥，此加密密钥之后将嵌入到请求正文中。

注意　有关生成加密密钥的详细信息，请参阅使用 POSTMAN 提出 DELETE 请求，第 7-64 页。

使用 POSTMAN 提出 DELETE 请求

操作步骤

步骤 1　在 Google Chrome 浏览器中打开 POSTMAN 插件。

步骤 2　使用左侧窗格中的选项创建新集合。

注意　有关使用 POSTMAN 插件的详细信息，请访问 https://github.com/a85/POSTMan-Chrome-Extension/wiki。

步骤 3　从下拉菜单中，选择 **DELETE**。

步骤 4　在 URL 栏，输入 URI。

URI 指定您尝试与之通信的 Cisco ISE 服务器和您尝试访问的 ISE 资源。有关 URI 格式的详细信息，请参阅 URI，第 7-63 页。

步骤 5　点击 **Basic Auth** 选项卡。

通过此选项，您可以指定显示的用户访问凭证。

步骤 6　在 Username 和 Password 字段指定您的访问凭证，然后点击 **Refresh Headers**。

POSTMAN 会显示 Authorization 标头与加密密钥。

步骤 7　通过指定以下值添加 Accept 标头：application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

注意　有关 Accept 标头的详细信息，请参阅 Accept 标头，第 7-63 页。

步骤 8　点击 **Send**。

POSTMAN 插件会显示 200 OK 状态响应，表示请求成功。指定的 ISE 资源将从 ISE 服务器删除。

Cisco ISE 故障原因报告

本附录提供可用于访问 Cisco ISE 故障原因报告的程序。通过 Cisco ISE 故障原因报告，您可以查看故障原因列表。

简介

Cisco ISE 故障原因报告是 Cisco ISE 用户界面中的一个选项，提供可能遇到的所有故障原因的信息。您可以使用此报告检查使用 Cisco ISE 查询故障排除 API 时从 Get Failure Reason Mapping 调用返回的输出。

通过 Cisco ISE 故障原因报告，您可以访问 Cisco ISE 软件定义的适用于思科监控 ISE 节点操作的故障原因完整列表。您可以通过以下程序查看或编辑定义的故障原因的列表。您必须登录到思科监控 ISE 目标节点的 Cisco ISE 用户界面，才能查看和访问故障原因。有关登录的详细信息，请参阅验证监控节点，第 1-2 页。

查看故障原因

步骤 1 选择 **Operations > Reports > Authnetication Summary** 报告。

步骤 2 在导航面板中，展开 **Monitoring**，然后选择 **Failure Reason Editor**。

步骤 3 从提供的过滤器列表中选择 Failure Reasons。

步骤 4 提供您要查找的故障原因。

步骤 5 点击 Run。

故障原因列表将显示在右侧面板中。

步骤 6 点击任意故障原因以在新窗口中显示详细报告。