

ISE身份组，通过REST API创建和修改用户

目录

[简介](#)

[先决条件](#)

[要求](#)

[使用的组件](#)

[配置](#)

[1.身份组创建](#)

[2.身份组详细信息检索](#)

[3.用户创建](#)

[4.用户详细信息检索](#)

[5.用户详细信息修改](#)

[验证](#)

简介

本文档介绍如何使用可用于身份管理自动化的Rest API创建和修改身份组 and 用户。本章中介绍的过程基于JSON格式的示例独立ISE部署和Rest API Firefox客户端(RES)。

先决条件

要求

Cisco 建议您了解以下主题：

- 思科身份服务引擎(ISE)
- REST API
- JSON

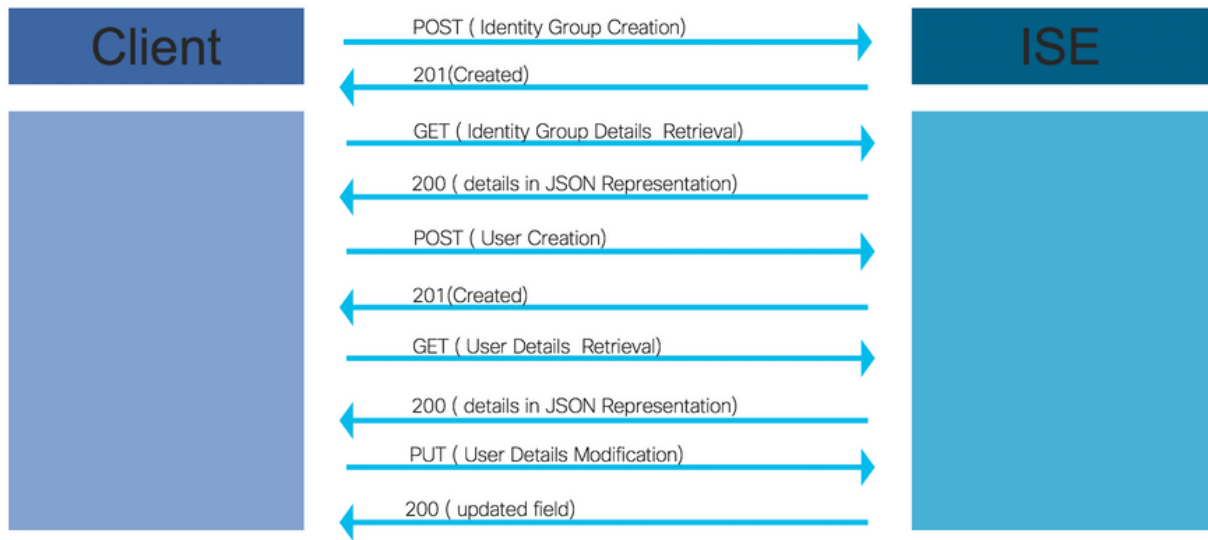
使用的组件

本文档不限于特定软件和硬件版本，只是通过REST API提供的示例配置指南。

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解所有命令的潜在影响。

配置

API Request and Response flow



这些是通过API执行操作的示例步骤，可用作构建您自己的调用的参考。

1.身份组创建

在POST方法的帮助下创建身份组。

API调用的URL:

<https://<ISE IP>:9060/ers/config/identitygroup>

API调用的报头：

HTTP“Content-Type”报头： application/json

HTTP“接受”报头： application/json

用于创建身份组的JSON代码

```
{ "IdentityGroup": { "name": "
```

示例：

Request ↗ ⚙ +

POST Send request

Headers ▾

Content-Type	application/json	🗑
Accept	application/json	🗑

[+Add header](#)

Basic auth ▾

ersadmin	<input type="checkbox"/> Show password?
----------	-------	---

Request body ▾

Type

```
{
  "IdentityGroup": {
    "name": "testusergroup",
    "description": " User group for testing",
    "parent": "NAC Group:NAC:IdentityGroups:User Identity Groups"
  }
}
```

Response (0.711s) - https://10.71.244.140:9060/ers/config/identitygroup

201 Created

[Headers >](#)

2. 身份组详细信息检索

借助GET方法获取身份组详细信息。

API调用的URL:

https://<ISE IP>:9060/ers/config/identitygroup?filter=name.CONTAINS.<身份组名称>

API调用的报头 :

HTTP“Content-Type”报头 : application/json

HTTP“接受”报头 : application/json

示例 :

GET

Headers ▾

Content-Type	application/json	🗑
Accept	application/json	🗑

[+Add header](#)

Basic auth ▾

ersadmin	<input type="checkbox"/> Show password?
----------	-------	---

Response (0.786s) - https://10.71.244.140:9060/ers/config/identitygroup?filter=name.CONTAINS.testusergroup

200 OK

Headers >

```
{
  "SearchResult": {
    "total": 1,
    "resources": [
      {
        "id": "b6ae7220-4289-11ea-a840-cee7c3fe1b0d",
        "name": "testusergroup",
        "description": " User group for testing",
        "link": {
          "rel": "self",
          "href": "https://10.71.244.140:9060/ers/config/identitygroup/b6ae7220-4289-11ea-a840-cee7c3fe1b0d",
          "type": "application/xml"
        }
      }
    ]
  }
}
```

注意：在此身份组中创建用户需要ID（在身份组详细信息中接收）。

3.用户创建

借助POST方法创建用户。

API调用的URL:

https://<ISE IP>:9060/ers/config/internaluser/

API调用的报头：

HTTP内容类型报头： application/json
 HTTP接受报头： application/json

用于用户创建的JSON代码：

```
{ "InternalUser": { "name": "
```

示例：

Request

POST [Send request](#)

Headers ▾

Content-Type	application/json	🗑
Accept	application/json	🗑

[+Add header](#)

Basic auth ▾

ersadmin	<input type="checkbox"/> Show password?
----------	-------	---

Request body ▾

Type

```
{
  "InternalUser": {
    "name": "test123",
    "email": "test@cisco.com",
    "enabled": true,
    "password": "Letmein!1",
    "firstName": "test",
    "lastName": "test",
    "changePassword": false,
    "identityGroups": "b6ae7220-4289-11ea-a840-cee7c3fe1b0d",
  }
}
```

Response (80.364s) - https://10.71.244.140:9060/ers/config/internaluser/

201 Created

[Headers >](#)

4.用户详细信息检索

借助GET方法获取用户详细信息。

API调用的URL:

https://<ISE IP>:9060/ers/config/internaluser

注意：此URL可用于过滤用户。可以使用firstName、lastName、identityGroup、name、description、email、enabled过滤用户。

建议使用邮件ID过滤用户详细信息，因为邮件ID对每个用户都是唯一的。

https://<ISE IP>:9060/ers/config/internaluser?filter=<用于过滤的字段名称>.CONTAINS.<过滤字段的值>

API调用的报头：

HTTP内容类型报头： application/json
 HTTP接受报头： application/json

示例：

The screenshot shows a REST client interface with the following elements:

- Request** tab selected.
- Method: **GET**
- URL: `140:9060/ers/config/internaluser?filter=email.CONTAINS.test@cisco.com`
- Send request** button.
- Headers** section:
 - Content-Type**: `application/json`
 - Accept**: `application/json`
 - +Add header** button.
- Basic auth** section:
 - Username: `ersadmin`
 - Password: `.....`
 - Show password?

Response (1.348s) - `https://10.71.244.140:9060/ers/config/internaluser?filter=email.CONTAINS.test@cisco.com`

200 OK

Headers >

```
{
  "SearchResult": {
    "total": 1,
    "resources": [
      {
        "id": "be9aee27-22f2-4a77-a8e9-3092ba3b636e",
        "name": "test123",
        "link": {
          "rel": "self",
          "href": "https://10.71.244.140:9060/ers/config/internaluser/be9aee27-22f2-4a77-a8e9-3092ba3b636e",
          "type": "application/xml"
        }
      }
    ]
  }
}
```

需要在此处:IDandNamereceived更新用户的密码或其他信息。hrefURL将用于更新用户信息。

5.用户详细信息修改

在PUT方法的帮助下修改用户密码。

API调用的URL:

`https://<ISE IP>:9060/ers/config/internaluser/<使用步骤4>中描述的过程接收的用户ID`

以上是使用步骤4中描述的流程收到的href URL。

API调用的报头：

HTTP内容类型报头：`application/json`
HTTP接受报头：`application/json`

用户凭证修改的JSON代码：

```
{ "InternalUser": { "id": "
```

示例：

The screenshot shows a REST client interface with the following sections:

- Request:** Method is PUT, URL is `140:9060/ers/config/internaluser/be9aee27-22f2-4a77-a8e9-3092ba3b636e`. A "Send request" button is present.
- Headers:** Two headers are defined: `Content-Type: application/json` and `Accept: application/json`. There is a "+Add header" button.
- Basic auth:** Username is `ersadmin`, password is masked with `.....`. A "Show password?" checkbox is present.
- Request body:** Type is set to "Custom". The body contains a JSON object:

```
{  "InternalUser": {    "id": "be9aee27-22f2-4a77-a8e9-3092ba3b636e",    "name": "test123",    "password": "Letmein!3",    "enablePassword": "Letmein!3"  }}
```
- Response (2.177s):** Status is **200 OK**. The URL is `https://10.71.244.140:9060/ers/config/internaluser/be9aee27-22f2-4a77-a8e9-3092ba3b636e`.

验证

要验证身份组，请导航至ISE GUI中的Administration > Identity Management > Groups > Identity Groups > User Identity Group。

要验证用户，请导航至Administration > Identity Management > Identities > Users in ISE GUI。