External RESTful Services Calls

Invoking an External RESTful Services API Call

You should fulfill the following prerequisites before invoking an External RESTful Services API call:

- Enable External RESTful Services from the CLI.
- Enable External RESTful Services Admin privileges.

You can use any REST client like JAVA, cURL Linux command, python to invoke External RESTful Services API calls.

Related Topics

- Enabling the External RESTful Services API, page 5-2
- External RESTful Services Authorization, page 5-15

GetVersion API Call

The GetVersion API call is common to all available resources: endpoints, internal users, endpoint identity groups, and security group tags (SGTs). It fetches the version information of the required resource.

Table 6-1 GetVersion Characteristics

Characteristics	Description
Description	Retrieve the version information of the specified resource
Synopsis	GET/ers/config/<resource-name>/versioninfo
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type VersionInfo
Response Status	200, 400, 401,403,404,415,500

GetVersion Sample Request

```
Sample Request for Get Version Internal Users API
Method:GET
URI: https://10.56.13.196:9060/ers/config/internaluser/versioninfo
HTTP Accept header:
application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
```

GetVersion Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:versionInfo xmlns:ns2="ers.ise.cisco.com">
    <currentServerVersion>1.0</currentServerVersion>
    <link type="application/xml" href="link" rel="self"/>
    <supportedVersions>0.9,0.8</supportedVersions>
</ns2:versionInfo>
```

Internal User External RESTful Services API Calls

External RESTful Services API Calls for Internal users support full Create, Read, Update, Delete (CRUD) functionality. The internal user ID used in these API calls is the UUID type stored in the Cisco ISE database.

Table 6-2 External RESTful Services API Calls Available for Internal Users

API Call	HTTP Method	URL	Content	QueryString
Get All Users	GET	/ers/config/internaluser	—	Page, Size, sortacs or sortdsn, Filter
Get User	GET	/ers/config/internaluser/{internal user-id}	—	—
Create User	POST	/ers/config/internaluser/	internaluser	—
Update User	PUT	/ers/config/internaluser/{internal user-id}	internaluser	—
Delete User	DELETE	/ers/config/internaluser/{internal user-id}	—	—

Internal User Schema

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser name="name" id="id" description="description"
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@domain.com</email>
    <enabled>true</enabled>
    <firstName>firstName</firstName>
    <identityGroups>identityGroups</identityGroups>
    <lastName>lastName</lastName>
    <password>password</password>
</ns3:internaluser>
```

Get All Users API Call

You use Get All Users API call to retrieve all the internal users in Cisco ISE.

Table 6-3 Get All Users Characteristics

Characteristics	Description
Description	Retrieve collection of internal users
Synopsis	GET /ers/config/internaluser
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: name, description, firstName, lastName, identityGroup, email, enabled)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	200, 400, 401, 403, 404, 415, 500

Get All Users Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/internaluser
HTTP Accept header:
application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
Request Content:
N/A
```

Get All Users Sample Response

```
HTTP Status: 200 (OK)
Content:


<?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>
```

Get User API Call

You use the Get User API call to retrieve an internal user using the user ID.

Table 6-4 Get User Characteristics

Characteristics	Description
Description	Retrieve the specified internal user
Synopsis	GET /ers/config/internaluser/{internaluser-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type InternalUser
Response Status	200, 400, 401,403, 404, 415, 500

Get User Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/internaluser/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
Request Content:
N/A
```

Get User Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser name="name" id="id" description="description"
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@domain.com</email>
    <enabled>true</enabled>
    <firstName>firstName</firstName>
    <identityGroups>identityGroups</identityGroups>
    <lastName>lastName</lastName>
    <password>password</password>
</ns3:internaluser>}
```

Create User API Call

You use the Create User API call to create internal users in Cisco ISE. A password is mandatory for creating an internal user using the External RESTful Services API.

Table 6-5 Create User API Call Characteristics

Characteristic	Description
Description	Create the specified internal user
Synopsis	POST /ers/config/internaluser/
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	InternalUser
Response Headers	Content-Length, Content-Type, Location
Response Message Body	Resource of type InternalUser
Response Status	201, 400, 401, 403, 415, 500

Create User Sample Request

```
Method: POST
URI: https://10.56.13.196:9060/ers/config/internaluser
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.internaluser.1.0+xml; charset=utf-8
Request Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser name="name" id="id" description="description"
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@domain.com</email>
    <enabled>true</enabled>
    <firstName>firstName</firstName>
    <identityGroups>identityGroups</identityGroups>
    <lastName>lastName</lastName>
    <password>password</password>
</ns3:internaluser>
```

Create User Sample Response

```
HTTP Status: 201 (Created)
Content:
N/A
```

Update User API Call

You use the Update User API call to update internal users in Cisco ISE. While updating internal users using the External RESTful Services API, if you do not intend to change the existing password, you need not supply the current password. In case you want to change the existing password when you update internal users, you can specify it in plain text.

Table 6-6 Update User Characteristics

Characteristic	Description
Description	Update the specified internal user
Synopsis	PUT /ers/config/internaluser/{internaluser-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	InternalUser
Response Headers	Content-Length, Content-Type
Response Message Body	List of updated fields
Response Status	200, 400, 401, 403, 404, 415, 500

Update User Sample Request

```
Method: PUT
URI: https://10.56.13.196:9060/ers/config/internaluser/{id}
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.internaluser.1.0+xml; charset=utf-8
Request Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:internaluser name="name" id="id" description="description"
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@domain.com</email>
    <enabled>true</enabled>
    <firstName>firstName</firstName>
    <identityGroups>identityGroups</identityGroups>
    <lastName>lastName</lastName>
    <password>password</password>
</ns3:internaluser>
```

Update User Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
    <updatedField field="field1">
        <newValue>val_new</newValue>
        <oldValue>val_old</oldValue>
    </updatedField>
    <updatedField field="some other field">
        <newValue>val_new</newValue>
        <oldValue>val_old</oldValue>
    </updatedField>
</ns2:updatedFields>
```

Delete User API Call

You use the Delete User API call to delete internal users from Cisco ISE.

Table 6-7 Delete User Characteristics

Characteristic	Description
Description	Delete the specified internal user
Synopsis	DELETE /ers/config/internaluser/{internaluser-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type InternalUser
Response Status	204, 400, 401, 403, 404, 415, 500

Delete User Sample Request

```
Method: DELETE
URI: https://10.56.13.196:9060/ers/config/internaluser/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.internaluser.1.0+xml
Request Content:
N/A
```

Delete User Sample Response

```
HTTP Status: 204 (No Content)
Content:
N/A
```

Endpoint External RESTful Services API Calls

The internal user ID used in these API calls is the UUID type stored in the Cisco ISE database.

Note These examples are not meant to be used as is because they have references to DB data. You should treat it as a basic template and edit it before sending to server.

Table 6-8 External RESTful Services API Calls Available for Endpoints

Operation	Method	URL	Content	QueryString
Get All Endpoints	GET	/ers/config/endpoint	—	page, size, sortacs or sortdsn, filter
Get Endpoint	GET	/ers/config/endpoint/{endpoint-id}	—	—
Create Endpoint	POST	/ers/config/endpoint/	endpoint	—
Update Endpoint	PUT	/ers/config/endpoint/{endpoint-id}	endpoint	—
Delete Endpoint	DELETE	/ers/config/endpoint/{endpoint-id}	—	—
Register Endpoint	PUT	/ers/config/endpoint/register	endpoint	—
Deregister Endpoint	PUT	/ers/config/endpoint/{endpoint-id}/deregister	—	—

Endpoint Schema

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint name="name" id="id" description="description"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
```

Get All Endpoints API Call

You use the Get All Endpoints API call to retrieve a list of endpoints.

Table 6-9 Get All Endpoints Characteristics

Characteristic	Description
Description	Retrieve collection of endpoints
Synopsis	GET ers/config/endpoint
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: mac, portalUser, profileId, staticGroupAssignment, staticProfileAssignment)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	200, 400, 401, 403, 404, 415, 500

Get All Endpoints Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/endpoint
HTTP Accept header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
Request Content:
N/A
```

Get All Endpoints Sample Response

```
HTTP Status: 200 (OK)
Content:


<?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>
```

Note The endpoint do not have hyperlinks for the complete object as it is not supported. Only the name description and ID are presented.

Get Endpoint API Call

You use Get Endpoint API call to retrieve an endpoint using the user ID.

Table 6-10 Get Endpoint Characteristics

Characteristics	Description
Description	Retrieve the specified endpoint
Synopsis	GET /ers/config/endpoint/{endpoint-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type InternalUser
Response Status	200, 400, 401,403, 404, 415, 500

Get Endpoint Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/endpoint/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
Request Content:
N/A
```

Get Endpoint Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint name="name" id="id" description="description" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
```

Create Endpoint API Call

You use the Create Endpoint API call to create endpoints in Cisco ISE.

Table 6-11 Create Endpoint Characteristics

Characteristic	Description
Description	Create the specified endpoint
Synopsis	POST /ers/config/endpoint
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	Endpoint
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type InternalUser
Response Status	200, 400, 401,403, 404, 415, 500

Create Endpoint Sample Request

```
Method: POST
URI: https://10.56.13.196:9060/ers/config/endpoint
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml; charset=utf-8
Request Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint name="name" id="id" description="description"
xmlns:ns2="ers.ise.cisco.com" xmlns:ns3="identity.ers.ise.cisco.com">
<groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
```

Create Endpoint Sample Response

```
HTTP Status: 201 (Created)
Content:
N/A
```

Update Endpoint API Call

You use the Update Endpoint API call to update endpoints in Cisco ISE.

Table 6-12 Update Endpoint Characteristics

Characteristic	Description
Description	Update the specified endpoint
Synopsis	PUT /ers/config/endpoint/{endpoint-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	Endpoint
Response Headers	Content-Length, Content-Type
Response Message Body	List of updated fields
Response Status	200, 400, 401,403, 404, 415, 500

Update Endpoint Sample Request

```
Method: PUT
URI: https://10.56.13.196:9060/ers/config/endpoint/{id}
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml; charset=utf-8
Request Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint name="name" id="id" description="description" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
```

Update Endpoint Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
    <updatedField field="field1">
        <newValue>val_new</newValue>
        <oldValue>val_old</oldValue>
    </updatedField>
    <updatedField field="some other field">
        <newValue>val_new</newValue>
        <oldValue>val_old</oldValue>
    </updatedField>
</ns2:updatedFields>
```

Delete Endpoint API Call

You use the Delete Endpoint API call to delete endpoints in Cisco ISE.

Table 6-13 Delete Endpoint Characteristics

Characteristic	Description
Description	Delete the specified endpoint
Synopsis	DELETE /ers/config/endpoint/{endpoint-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	—
Response Status	200, 400, 401,403, 404, 415, 500

Delete Endpoint Sample Request

```
Method: DELETE
URI: https://10.56.13.196:9060/ers/config/endpoint/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml
Request Content:
N/A
```

Delete Endpoint Sample Response

```
HTTP Status: 204 (No Content)
Content:
N/A
```

Register Endpoint API Call

You use the Register Endpoint API call to register endpoints in Cisco ISE. If the endpoint already exists, it will be registered. If it does not exist, it will be created and then registered. In both scenarios, the return status will be 204, which means no content. Similar to the GUI registration flow, the endpoint is statically assigned to the Registered Devices group and the portal user and identity store will be set as specified in the content.

Table 6-14 Register Endpoint Characteristics

Characteristic	Description
Description	Register the specified endpoint
Synopsis	PUT /ers/config/endpoint/register
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	endpoint

Table 6-14 Register Endpoint Characteristics (continued)

Characteristic	Description
Response Headers	Content-Length, Content-Type
Response Message Body	List of updated fields
Response Status	202, 400, 401, 403, 404, 415, 500

Register Endpoint Sample Request

```
Method: PUT
URI: https://10.56.13.196:9060/ers/config/endpoint/register
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml; charset=utf-8
Request Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint name="name" id="id" description="description" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
```

Register Endpoint Sample Response

```
HTTP Status: null
Content:
N/A
```

Deregister Endpoint API Call

You use the Deregister Endpoint API call to deregister endpoints in Cisco ISE.

Table 6-15 Deregister Endpoint Characteristics

Characteristic	Description
Description	Deregister the specified endpoint
Synopsis	PUT /ers/config/endpoint/{endpoint-id}/deregister
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—

Table 6-15 Deregister Endpoint Characteristics (continued)

Characteristic	Description
Response Message Body	—
Response Status	202, 400, 401, 403, 404, 415, 500

Deregister Endpoint Sample Request

```
Method: PUT
URI: https://10.56.13.196:9060/ers/config/endpoint/{id}/deregister
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.endpoint.1.0+xml; charset=utf-8
Request Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:endpoint name="name" id="id" description="description" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="identity.ers.ise.cisco.com">
    <groupId>groupId</groupId>
    <identityStore>identityStore</identityStore>
    <identityStoreId>identityStoreId</identityStoreId>
    <mac>00:01:02:03:04:05</mac>
    <portalUser>portalUser</portalUser>
    <profileId>profileId</profileId>
    <staticGroupAssignment>true</staticGroupAssignment>
    <staticProfileAssignment>false</staticProfileAssignment>
</ns3:endpoint>
```

Deregister Endpoint Sample Response

```
HTTP Status: null
Content:
N/A
```

Endpoint Identity Group External RESTful Services API Calls

The internal user ID used in these API calls is the UUID type stored in the Cisco ISE database.

Note The examples shown in the subsequent sections are not meant to be used as is because they have references to DB data. You should treat it as a basic template and edit it before sending to server.

Table 6-16 External RESTful Services API Calls Available for Endpoint Identity Groups

Operation	Method	URL	Content	QueryString
Get All Endpoint Groups	GET	/ers/config/endpointgroup	—	page, size, sortacs or sortdsn, filter
Get Endpoint Group	GET	/ers/config/endpointgroup/{endpointgroup-id}	—	—
Create Endpoint Group	POST	/ers/config/endpointgroup/	Endpointgroup	—
Update Endpoint Group	PUT	/ers/config/endpointgroup/{endpointgroup-id}	Endpointgroup	—
Delete Endpoint Group	DELETE	/ers/config/endpointgroup/{endpointgroup-id}	—	—

Endpoint Identity Groups Schema

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

Get All Endpoint Identity Groups API Call

You use the Get All Endpoint Identity Groups API call to retrieve a collection of endpoint identity groups.

Table 6-17 Get All Endpoint Identity Groups Characteristics

Characteristic	Description
Description	Retrieve a collection of endpoint groups
Synopsis	GET /ers/config/endpointgroup
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	202, 400, 401, 403, 404, 415, 500

Get All Endpoint Identity Groups Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/endpointgroup
HTTP Accept header:
application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
```

```
Request Content:
N/A
```

Get All Endpoint Identity Groups Sample Response

```
HTTP Status: 200 (OK)
Content:


<?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>
```

Note The endpoint do not have hyperlinks for the complete object, only the name, description, and ID are presented.

Get Endpoint Identity Group API Call

You use the Get Endpoint Identity Group API call to retrieve the specified endpoint identity group.

Table 6-18 Get Endpoint Identity Group Characteristics

Characteristic	Description
Description	Retrieve the specified endpoint group
Synopsis	GET /ers/config/endpoint/{endpointgroup-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type Endpoint
Response Status	200, 400, 401,403, 404, 415, 500

Get Endpoint Identity Group Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/endpointgroup/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
Request Content:
N/A
```

Get Endpoint Identity Group Sample Response

```
HTTP Status: 200 (OK)
Content:


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

Create Endpoint Identity Group API Call

You use the Create Endpoint Identity Group API call to create a specified endpoint identity group.

:

Table 6-19 Create Endpoint Identity Group Characteristics

Characteristic	Description
Description	Create the specified endpoint group
Synopsis	POST /ers/config/endpoint/
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	Endpoint Group
Response Headers	Content-Length, Content-Type, Location
Response Message Body	Resource of type Endpoint Group
Response Status	201, 400, 401, 403, 415, 500

Create Endpoint Identity Group Sample Request

```
Method: POST
URI: https://10.56.13.196:9060/ers/config/endpointgroup
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml; charset=utf-8
Request Content:


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

Create Endpoint Identity Group Sample Response

```
HTTP Status: 201 (Created)
Content:
N/A
```

Update Endpoint Identity Group API Call

You use the Update Endpoint Identity Group API call to update a specified endpoint group.

Table 6-20 Update Endpoint Identity Group Characteristics

Characteristic	Description
Description	Update the specified endpoint group
Synopsis	PUT /ers/config/endpoint/{endpointgroup-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	Endpoint Group
Response Headers	Content-Length, Content-Type
Response Message Body	List of updated fields
Response Status	200, 400, 401, 403, 404, 415, 500

Update Endpoint Identity Group Sample Request

```
Method: PUT
URI: https://10.56.13.196:9060/ers/config/endpointgroup/{id}
HTTP Content-Type header:
application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml; charset=utf-8
Request Content:


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

Update Endpoint Identity Group Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:updatedFields xmlns:ns2="ers.ise.cisco.com">
    <updatedField field="field1">
        <newValue>val_new</newValue>
        <oldValue>val_old</oldValue>
    </updatedField>
    <updatedField field="some other field">
        <newValue>val_new</newValue>
        <oldValue>val_old</oldValue>
    </updatedField>
</ns2:updatedFields>
}
```

Delete Endpoint Identity Group API Call

You use the Delete Endpoint Identity Group API call to delete a specified endpoint identity group.

Table 6-21 Delete Endpoint Identity Group Characteristics

Characteristic	Description
Description	Delete the specified endpoint group
Synopsis	DELETE /ers/config/endpointgroup/{endpointgroup-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	—
Response Status	204, 400, 401, 403, 404, 415, 500

Delete Endpoint Identity Group Sample Request

```
Method: DELETE
URI: https://10.56.13.196:9060/ers/config/endpointgroup/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.endpointgroup.1.0+xml
Request Content:
N/A
```

Delete Endpoint Identity Group Sample Response

```
HTTP Status: 204 (No Content)
Content:
N/A
```

Profiler Profile External RESTful Services API Calls

Profiler Profile API calls enable you to search profiles. Profiler profile is the profile the endpoint is classified as or statically assigned to. Profiler Profile is also called profiler policy. The profile Id is an attribute on the endpoint. Endpoints can be filtered by names.

Note The examples shown in the following sections are not meant to be used as is because they have references to DB data. You should treat it as a basic template and edit it before sending to server.

Table 6-22 External RESTful Services API Calls Available for Profiler Profile

Operation	Method	URL	Content	QueryString
Get All Profiles	GET	/ers/config/profilerprofile	—	Start Size sortacs or sortdsn filter
Get Profile (Get by ID)	GET	/ers/config/profilerprofile/{id}	—	—

Profiler Profile Schema

```
<?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"/>
```

Get All Profiles API Call

You use the Get All Profiles API call to retrieve a list of Profiler profiles.

Table 6-23 Get All Profiles Characteristics

Characteristic	Description
Description	Retrieves a collection of profiler profiles
Synopsis	GET /ers/config/profilerprofile
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: name)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	200, 400, 401, 403, 404, 415, 500

Get All Profiles Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/profilerprofile
HTTP Accept header:
application/vnd.com.cisco.ise.identity.profilerprofile.1.0+xml
Request Content:
N/A
```

Get All Profiles Sample Response

```
HTTP Status: 200 (OK)

Content:



<?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>
```

Note The Profiler profile will not have the hyperlinks for the complete object as it is not supported in this release. And only name description and id will be presented.

Get Profile API Call

You use the Get Profile API call to retrieve a specific profiler profile.

Table 6-24 Get Profile Characteristics

Characteristic	Description
Description	Retrieves the specified profiler profile.
Synopsis	GET ers/config/profilerprofile/{id}
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: name)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	200, 400, 401, 403, 404, 415, 500

Get Profile Sample Request

```
Method: GET

URI: https://10.56.13.196:9060/ers/config/profilerprofile/{id}

HTTP Accept header:
```

```
application/vnd.com.cisco.ise.identity.profilerprofile.1.0+xml

Request Content:

N/A
```

Get Profile Sample Response

```
HTTP Status: 200 (OK)

Content:



<?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"/>
```

Identity Group External RESTful Services API Calls

External RESTful Services Identity Groups API calls enable you to search Identity Groups.

Note The examples shown in the subsequent sections are not meant to be used as is because they have references to DB data. You should treat it as a basic template and edit it before sending to server.

Table 6-25 External RESTful Services API Calls Available for Identity Groups

Operation	Method	URL	Content	QueryString
Get Identity Group (Get by ID)	GET	/ers/config/identitygroup /{id}	—	—
List Identity Groups	GET	/ers/config/identitygroup	—	page, size, sortacs or sortdsn, filter
Get IdentityGroup Resource Version Info	GET	/ers/config/identitygroup /version	—	—

Identity Group Schema

```
<?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>
```

Get All Identity Groups API Call

You use the Get All Identity Groups API call to retrieve identity groups by ID in Cisco ISE.

Table 6-26 Get All Identity Groups Characteristics

Characteristic	Description
Description	Retrieve identity group resources by ID.
Synopsis	GET /ers/config/identitygroup/{id}
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: name, description)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	—
Response Status	200, 400, 401, 403, 404, 415, 500

Get All Identity Groups Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/identitygroup/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.identitygroup.1.0+xml
Request Content:
N/A
```

Get All Identity Groups Sample Response

```
HTTP Status: 200 (OK)

Content:



<?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>
```

Get Identity Group API Call

You use the Get Identity Group API call to retrieve a specific identity group in Cisco ISE.

Table 6-27 Get Identity Group Characteristics

Characteristic	Description
Description	Retrieves a specified identity group resource
Synopsis	GET /ers/config/identitygroup
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: name, description)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	200, 400, 401, 403, 404, 415, 500

Get Identity Group Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/identitygroup/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.identity.identitygroup.1.0+xml
Request Content:
N/A
```

Get Identity Group Sample Response

```
HTTP Status: 200 (OK)
Content:


<?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>
```

External RESTful Services API Calls for Security Group Tags

Security Group Tags (SGT) API calls enable you to search SGTs. The internal user ID used in these API calls is the UUID type stored in the Cisco ISE database.

Note The examples shown in the subsequent sections are not meant to be used as is because they have references to DB data. You should treat it as a basic template and edit it before sending to server.

Table 6-28 External RESTful Services API Calls Available for SGTs

Operation	Method	URL	Content	QueryString
Get All SGTs	GET	/ers/config/sgt	—	page, size, sortacs or sortdsn, filter
Get SGT (Get by ID)	GET	/ers/config/sgt/{sgt-id}	—	—

Get All SGTs API Call

You use the Get All SGTs API call to retrieve a collection of SGT resources.

Table 6-29 Get All SGTs Characteristics

Characteristic	Description
Description	Retrieve a collection of SGT resources
Synopsis	GET /ers/config/sgt
Request Headers	Accept, Authorization, Host
QueryString	start, size, sortbyacn, sortbydcn, filter (fields supported by filter: name)
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	SearchResult
Response Status	200, 400, 401, 403, 404, 415, 500

Get All SGTs Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/sgt
HTTP Accept header:
application/vnd.com.cisco.ise.sga.sgt.1.0+xml
Request Content:
N/A
```

Get All SGTs Sample Response

```
HTTP Status: 200 (OK)
Content:


<?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>
```

Get SGT API Call

You use the Get SGT API call to retrieve a specific SGT resource.

Table 6-30 Get SGT Characteristics

Characteristic	Description
Description	Retrieve the specified SGT
Synopsis	GET /ers/config/sgt/{sgt-id}
Request Headers	Accept, Authorization, Host
QueryString	—
Request Message Body	—
Response Headers	Content-Length, Content-Type
Response Message Body	Resource of type InternalUser
Response Status	200, 400, 401, 403, 404, 415, 500

Get SGT Sample Request

```
Method: GET
URI: https://10.56.13.196:9060/ers/config/sgt/{id}
HTTP Accept header:
application/vnd.com.cisco.ise.sga.sgt.1.0+xml
Request Content:
N/A
```

Get SGT Sample Response

```
HTTP Status: 200 (OK)
Content:


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:sgt name="name" id="id" description="description" xmlns:ns2="ers.ise.cisco.com"
xmlns:ns3="sga.ers.ise.cisco.com">
    <generationId>generationId</generationId>
    <isTagFromRange>isTagFromRange</isTagFromRange>
    <value>1</value>
</ns3:sgt>
}
```

Using a REST API Client

To build and test applications using the External RESTful Services API, you can use any industry-standard REST API client, such as the POSTMAN plugin for Google Chrome.

Designed according to REST architecture and principles, the POSTMAN Chrome plugin is an easy to use REST client. It allows you to save collections of requests and provides many features that are useful while working in a browser environment and have lightweight specific tasks. You can also use the POStMAN plugin for testing the External RESTful Services APIs.

POSTMAN enables you to send and retrieve standard HTTP and HTTPS requests and responses using the Google Chrome web browser. You can use the following standard HTTP methods to perform CRUD operations on Cisco ISE resources:

- GET
- POST
- PUT
- DELETE

The External RESTful Services API enables you to use these HTTP requests in various API calls, which in turn enable you to perform operations on Cisco ISE servers. For a comprehensive list of operations in which these HTTP requests are used, see External RESTful Services Calls, page 6-1.

Note To download the POSTMAN plugin, go to https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm?hl=en. For more information on using the POSTMAN plugin, go to https://github.com/a85/POSTMan-Chrome-Extension/wiki.

Making an External RESTful Services API Call Using GET

The GET method requests a representation of the specified resource. Requests using GET only retrieve data and do not have any other effect.

You can perform a search operation by sending a GET request to a resource URI. By default, the result is the first page (page index = 0) with default size of 20. By adding a filter, Sort, and paging parameters to the URI, you can control the results of the search operation. The search returns the following result type:

```
SearchResult: MediaType[MediaType:
[application/vnd.com.cisco.ise.ers.searchresult.1.0+xml]
```

The resulting resources are a collection of base resources that contain the name, id, description, and a link to the full representation of the resource.

Note An External RESTful Services API call uses the GET HTTP method in addition to other components of the External RESTful Services API, which are not described in this section. For more details on various External RESTful Services API components, such as the characteristics, requests, and responses, see External RESTful Services Calls.

The request body of the External RESTful Services API call that uses the GET HTTPS method contains the following three building blocks:

- URI
- Accept Header
- Authorization Header

URI

The GET method sends the Uniform Resource Identifier (URI) to the Cisco ISE server and the HTTP reply is the raw result data. A typical URI must adhere to the following format:

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

  Where *<Cisco ISE Server Address>* denotes the server address of the Cisco ISE server, *<port>* denotes port 9060, *<namespace>* denotes the namespace to which the Cisco ISE resource belongs to, and *<Cisco ISE Resource Name>* denotes the name of the Cisco ISE resource.

The following example shows a URI that requests data for the interaluser Cisco ISE resource:

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

Note The URI is not the request body; it is just a URL. This URL is sent to the server using the GET method.

Accept Header

The Accept header must adhere to the following format:

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

Where *<resource-namespace>* denotes the namespace to which the Cisco ISE resource belongs to, *<resource-type>* denotes the type of Cisco ISE resource, *<major-version>* denotes the major version number of the Cisco ISE deployment, and *<minor-version>* denotes the minor version number of the deployment.

The following example shows a typical Accept header:

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

Authorization Header

The Authorization header contains the encryption authorization key that is embedded in the GET request. After specifying the authorization credentials, you must generate the encryption key, which is then embedded in the request body.

Note For more information on generating an encryption key, see Making a GET Request Using POSTMAN, page 6-30.

Making a GET Request Using POSTMAN

Step 1 Open the POSTMAN plugin in the Google Chrome browser.

Step 2 Create a new collection using the options in the left pane.

Note For more information on using the POSTMAN plugin, go to https://github.com/a85/POSTMan-Chrome-Extension/wiki.

Step 3 From the drop-down menu, choose **GET**.

Step 4 In the URL address bar, enter a URI.

The URI specifies the Cisco ISE server with which you are trying to communicate and the Cisco ISE resource that you are trying to access. For more information on the format of the URI, see URI, page 6-29.

Step 5 Click the **Basic Auth** tab.

The options that enable you to specify the user access credentials appear.

Step 6 Specify the access credentials in the Username and Password fields and click **Refresh Headers**.

POSTMAN displays an Authorization header with an encryption key.

Step 7 Add an Accept header by specifying the following value:
application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

Note For more information on the Accept header, see Accept Header, page 6-30.

Step 8 Click **Send**.

The POSTMAN plugin displays a 200 OK status response indicating that the request is successful. The request also returns the details of the resources that you have specified in the URI.

Making an External RESTful Services API Call Using POST

The POST method requests that the server accept the entity enclosed in the request as a new subordinate of the web resource identified by the URI.

You can create a resource by sending a POST request to a resource URI. The request content holds XML representation of the resource, and must be well formatted according to the schema. The request header holds the encrypted authorization and the content-type that defines the resource content and its version.

Note An External RESTful Services API call uses the POST HTTP method in addition to other components of the External RESTful Services API, which are not described in this section. For more details on various components, such as characteristics, requests, and responses, see External RESTful Services Calls.

The request body of the External RESTful Services API call that uses the POST HTTP method contains the following three building blocks:

- URI
- Content-Type Header
- Authorization Header

URI

The POST method sends the Uniform Resource Identifier (URI) to the Cisco ISE server. A typical URI must adhere to the following format:

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

  Where *<Cisco ISE Server Address>* denotes the server address of the Cisco ISE server, *<port>* denotes port 9060, *<namespace>* denotes the namespace to which the Cisco ISE resource belongs to, and *<Cisco ISE Resource Name>* denotes the name of the Cisco ISE resource.

The following example shows a URI that requests data for the interaluser Cisco ISE resource:

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

Note The URI is not the request body; it is just a URL. This URL is sent to the server using the POST method.

Content-Type Header

The Content-Type header must adhere to the following format:

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

  Where *<resource-namespace>* denotes the namespace to which the Cisco ISE resource belongs to, *<resource-type>* denotes the type of Cisco ISE resource, *<major-version>* denotes the major version number of the Cisco ISE deployment, and *<minor-version>* denotes the minor version number of the deployment.

The following example shows a typical Content-Type header:

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

Authorization Header

The Authorization header contains the encryption authorization key that is embedded in the POST request. After specifying the authorization credentials, you must generate the encryption key, which is then embedded in the request body.

Note For more information on generating an encryption key, see Making a POST Request Using POSTMAN, page 6-32.

Making a POST Request Using POSTMAN

Step 1 Open the POSTMAN plugin in the Google Chrome browser.

Step 2 Create a new collection using the options in the left pane.

Note For more information on using the POSTMAN plugin, go to https://github.com/a85/POSTMan-Chrome-Extension/wiki.

Step 3 From the drop-down menu, choose **POST**.

Step 4 In the URL address bar, enter the URI.

The URI specifies the Cisco ISE server with which you are trying to communicate and the Cisco ISE resource that you are trying to access. For more information on the format of the URI, see URI, page 6-31.

Step 5 Click the **Basic Auth** tab.

The options that enable you to specify the user access credentials appear.

Step 6 Specify the access credentials in the Username and Password fields and click **Refresh Headers**.

POSTMAN displays an Authorization header with an encryption key.

Step 7 Add a Content-Type header by specifying the following value:
application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

Note For more information on the Content-Type header, see Content-Type Header, page 6-32.

Step 8 From the drop-down menu that appears next to the **raw** button, choose **XML**.

Step 9 Click **raw**.

Step 10 The POSTMAN plugin opens an editing pane that enables you to specify the body of the POST request.

Step 11 Enter the message body of your POST request in the editing pane.

Note The message body must contain the details corresponding to the Cisco ISE resource that you trying to create on the Cisco ISE server. For example, while creating an internal user, you must specify details such as the name and description of the internal user, password, and so on. For more details on the message body of the External RESTful Services API calls that use the POST request and the details of the Cisco ISE resources that you need to specify, see External RESTful Services Calls.

Step 12 Click **Send**.

The POSTMAN plugin displays a 201 CREATED status response indicating that the request is successful. You can log on to Cisco ISE to verify whether the resource you added appears in the GUI.

Making an External RESTful Services API Call Using PUT

The PUT method requests that the enclosed entity be stored under the supplied URI. If the URI refers to an already existing resource, it is modified; if the URI does not point to an existing resource, then the server can create the resource with that URI.

You can update a resource by sending a PUT request to a resource URI. The request content holds XML representation of the resource, and must be well formatted according the schema. The request header holds the encrypted authorization and the content-type that defines the resource content and its version.

Note An External RESTful Services API call uses the PUT HTTP method in addition to other components of the External RESTful Services API, which are not described in this section. For more details on various components, such as characteristics, requests, and responses, see External RESTful Services Calls.

The request body of the External RESTful Services API call that uses the PUT HTTP method contains the following three building blocks:

- URI
- Content-Type Header
- Authorization Header

URI

The PUT method sends the Uniform Resource Identifier (URI) to the Cisco ISE server. A typical URI must adhere to the following format:

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

  Where *<Cisco ISE Server Address>* denotes the server address of the Cisco ISE server, *<port>* denotes the port 9060, *<namespace>* denotes the namespace to which Cisco ISE resource belongs to, and *<Cisco ISE Resource Name>* denotes the name of the Cisco ISE resource.

The following example shows a URI that requests data for the interaluser Cisco ISE resource:

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

Note The URI is not the request body; it is just a URL. This URL is sent to the server using the PUT method.

Content-Type Header

The Content-Type header must adhere to the following format:

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

  Where *<resource-namespace>* denotes the namespace to which the Cisco ISE resource belongs to, *<resource-type>* denotes the type of Cisco ISE resource, *<major-version>* denotes the major version number of the Cisco ISE deployment, and *<minor-version>* denotes the minor version number of the deployment.

The following example shows a typical Content-Type header:

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

Authorization Header

The Authorization header contains the encryption authorization key that is embedded in the PUT request. After specifying the authorization credentials, you must generate the encryption key, which is then embedded in the request body.

Note For more information on generating an encryption key, see Making a PUT Request Using POSTMAN, page 6-34.

Making a PUT Request Using POSTMAN

Step 1 Open the POSTMAN plugin in the Google Chrome browser.

Step 2 Create a new collection using the options in the left pane.

Note For more information on using the POSTMAN plugin, go to https://github.com/a85/POSTMan-Chrome-Extension/wiki.

Step 3 From the drop-down menu, choose **PUT**.

Step 4 In the URL address bar, enter the URI.

The URI specifies the Cisco ISE server with which you are trying to communicate and the Cisco ISE resource that you are trying to access. For more information on the format of the URI, see URI, page 6-33.

Step 5 Click the **Basic Auth** tab.

The options that enable you to specify the user access credentials appear.

Step 6 Specify the access credentials in the Username and Password fields and click **Refresh Headers**.

POSTMAN displays an Authorization header with an encryption key.

Step 7 Add a Content-Type header by specifying the following value:
application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

Note For more information on the Content-Type header, see Content-Type Header, page 6-34.

Step 8 From the drop-down menu that appears next to the **raw** button, choose **XML**.

Step 9 Click **raw**.

Step 10 The POSTMAN plugin opens an editing pane that enables you to specify the body of the POST request.

Step 11 Enter the message body of your POST request in the editing pane.

Note The message body must contain the details corresponding to the Cisco ISE resource that you trying to update on the Cisco ISE server. For example, while updating an internal user, you must specify details such as the name and description of the internal user, password, and so on. For more details on the message body of the External RESTful Services API calls that use the PUT request and the details of the Cisco ISE resources that you need to specify, see External RESTful Services Calls.

Step 12 Click **Send**.

The POSTMAN plugin displays a 201 CREATED status response indicating that the request is successful. You can log on to Cisco ISE to verify whether the resource you added appears in the GUI.

Making an External RESTful Services API Call Using DELETE

The DELETE method deletes the specified resource.

You can delete a resource by sending a DELETE request to a resource URI. The request header holds the encrypted authorization and the content-type that defines the resource content and its version.

Note An External RESTful Services API call uses the DELETE HTTP method in addition to other components of the External RESTful Services API, which are not described in this section. For more details on various components, such as characteristics, requests, and responses, see External RESTful Services Calls.

The request body of the External RESTful Services API call that uses the DELETE HTTP method contains the following three building blocks:

- URI
- Accept Header
- Authorization Header

URI

The DELETE method sends the Uniform Resource Identifier (URI) to the Cisco ISE server. A typical URI must adhere to the following format:

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

  Where *<Cisco ISE Server Address>* denotes the server address of the Cisco ISE server, *<port>* denotes the port 9060, *<namespace>* denotes the namespace to which the Cisco ISE resource belongs to, and *<Cisco ISE Resource Name>* denotes the name of the Cisco ISE resource.

The following example shows a URI that requests data for the interaluser Cisco ISE resource:

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

Note The URI is not the request body; it is just a URL. This URL is sent to the server using the GET method.

Accept Header

The Accept header must adhere to the following format:

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

Where *<resource-namespace>* denotes the namespace to which the Cisco ISE resource belongs to, *<resource-type>* denotes the type of Cisco ISE resource, *<major-version>* denotes the major version number of the Cisco ISE deployment, and *<minor-version>* denotes the minor version number of the deployment.

The following example shows a typical accept header:

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

Authorization Header

The Authorization header contains the encryption authorization key that is embedded in the DELETE request. After specifying the authorization credentials, you must generate the encryption key, which is then embedded in the request body.

Note For more information on generating an encryption key, see Making a DELETE Request Using POSTMAN, page 6-36.

Making a DELETE Request Using POSTMAN

Step 1 Open the POSTMAN plugin in the Google Chrome browser.

Step 2 Create a new collection using the options in the left pane.

Note For more information on using the POSTMAN plugin, go to https://github.com/a85/POSTMan-Chrome-Extension/wiki.

Step 3 From the drop-down menu, choose **DELETE**.

Step 4 In the URL address bar, enter the URI.

The URI specifies the Cisco ISE server with which you are trying to communicate and the Cisco ISE resource that you are trying to access. For more information on the format of the URI, see URI, page 6-36.

Step 5 Click the **Basic Auth** tab.

The options that enable you to specify the user access credentials appear.

Step 6 Specify the access credentials in the Username and Password fields and click **Refresh Headers**.

POSTMAN displays an Authorization header with an encryption key.

Step 7 Add an accept header by specifying the following value:
application/vnd.com.cisco.ise.ers.<namespace>.<ise resource>.1.0+xml

Note For more information on the Accept Header, see Accept Header, page 6-36.

Step 8 Click **Send**.

The POSTMAN plugin displays a 200 OK status response indicating that the request is successful. You can log on to Cisco ISE to verify whether the resource you added appears in the GUI.

Java External RESTful Services Demo Application

The External RESTful Services Java Demo application is a simple External RESTful Services client code that uses the External RESTful Services API calls to configure internal users and endpoints. The main Cisco ISE dependency for this Java client code is the ers-schema.jar file. The Java Demo application consists of a generic HTTP client that is based on the Apache DefaultHttpClient and two main classes. These classes execute REST calls in a sequential order.

The source code and the required libraries are available for your reference in the demo application zip file. You can download the Java External RESTful Services Demo application and the ers-schema.jar file from the External RESTful Services Online SDK.

Related Topics

- Downloading the Java External RESTful Services Demo Application, page 6-37

Downloading the Java External RESTful Services Demo Application

Step 1 Enter the External RESTful Services SDK URL in the address bar of your browser. For example, *https://<ise hostname or ip address>:9060/ers/sdk*, where *<ise hostname or ip address>* denotes the ip address of the Cisco ISE host.

Step 2 Enter the username and case-sensitive password that was specified and configured during the initial Cisco ISE setup.

Step 3 Click **Login** or press **Enter**.

Step 4 From the Navigation pane of the External RESTful Services Online SDK page, choose **Developer Resources > Downloads**.

Step 5 Click **Schema Jar (ers-schema-1.2.0-896.jar)**.

The *ers-schema-1.0-892.jar* file is downloaded to your local machine.

Step 6 Click **Java ERS Demo Application**.

The *ers-demo-app.zip* file is downloaded to your local machine.

After you Finish

Note Refer to the *readme.txt* file that is available in the demo application zip file (ers-demo-app.zip) for all the information that is required to use the Java External RESTful Services demo application.