

API Keys

Last Modified on 07/31/2024 2:08 pm EDT

Overview

- API keys are used to authenticate requests to the Resolver Core API without entering user credentials.
- API keys never expire, with no need to establish or maintain a session.
- API keys are tied to the user's org account. This means that if a user doesn't have permission to perform an action in Core, they won't be able to do it through the API.
- Only admins and super admins can create API keys. Admins can only create keys for orgs on which they're an admin.
- For security purposes, API keys are not stored. If you misplace the key, it cannot be retrieved and must be regenerated.
- It's possible to create API keys to impersonate other users; however, only super admins can enable this feature. See the **Impersonation** section of [Use an API Key](#) article for more information.
- Only active users of the same org as the API key user can be impersonated.
- Any call can be made using an API key while impersonating another user; however, only three endpoints in the Swagger user interface support this:
 - **POST** /data/file/file in the **file** resource;
 - **POST** /data/object/{objectID}/file/{fileID} in the **object** resource; and
 - **POST** /creation/import/json in the **dataImport** resource.

file Show/Hide | List Operations | Expand Operations

POST /creation/import/finishUpload Finishes uploading the data for an import, processing the data.

POST /data/file Add a file reference to the files table

POST /data/file/crop Post file and crop information to the files service and add reference to the files table

POST /data/file/file Post a file to the files service and add reference to the files table

Response Class (Status 200)
OK

Model Example Value

```
{
  "id": 0,
  "displayName": "string",
  "fileType": "string",
  "contentType": "string",
  "fileUUID": "string",
  "fileUrl": "string"
}
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
x-api-key	<input type="text"/>		header	string
impersonate-user-id	<input type="text"/>		header	double
file	<input type="button" value="Choose File"/> <input type="text" value="No file chosen"/>		formData	file

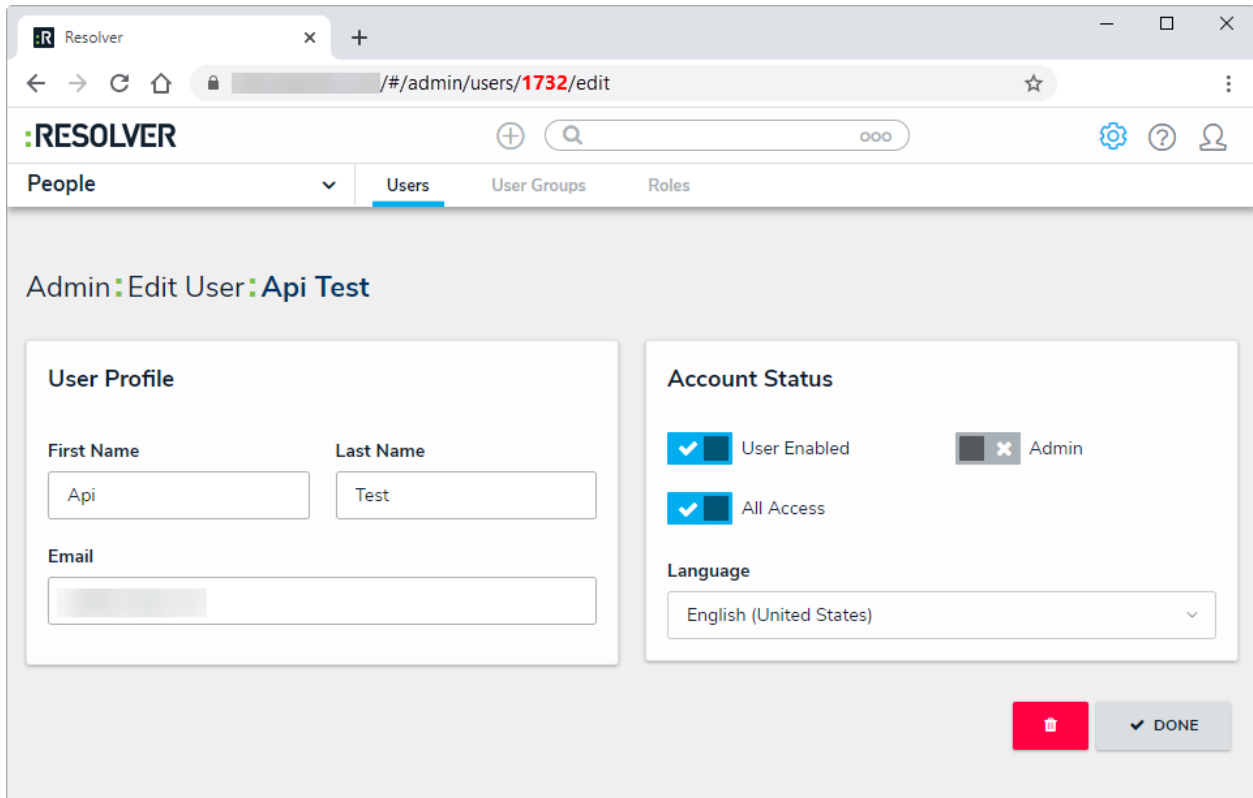
The file endpoint.

Impersonation

API keys can be used to authenticate requests in Core while impersonating another user; however, note the following:

- Impersonation can only be enabled by a member of Resolver Support at the time the key was created. Should you wish to enable impersonation, contact [Resolver Support](#) to create a new API key.
- Any user can be impersonated, provided you've obtained their user ID and the user is active in the org the API key was created for.
- Actions performed while impersonating using an API key are captured in the [audit trail](#) as "[API User's Name] impersonating [Impersonated User's Name]".
- If a user is impersonated using an external system (integration), the **Modified By** [property](#) on an object will show "[API User's Name]", but would still be captured in the audit trail as "[API User's Name] impersonating [Impersonated User's Name]".




To impersonate a user with an API key, open a supported endpoint in Swagger, enter the API key in the **x-api-key** field and the ID of the user to be impersonated in the **impersonate-user-id** field. The user ID can be obtained from the address bar of your browser after navigating to the **Edit User** page for the user (e.g., 1732).



The Edit User page. The user ID is displayed in the address bar.

Create an API Key

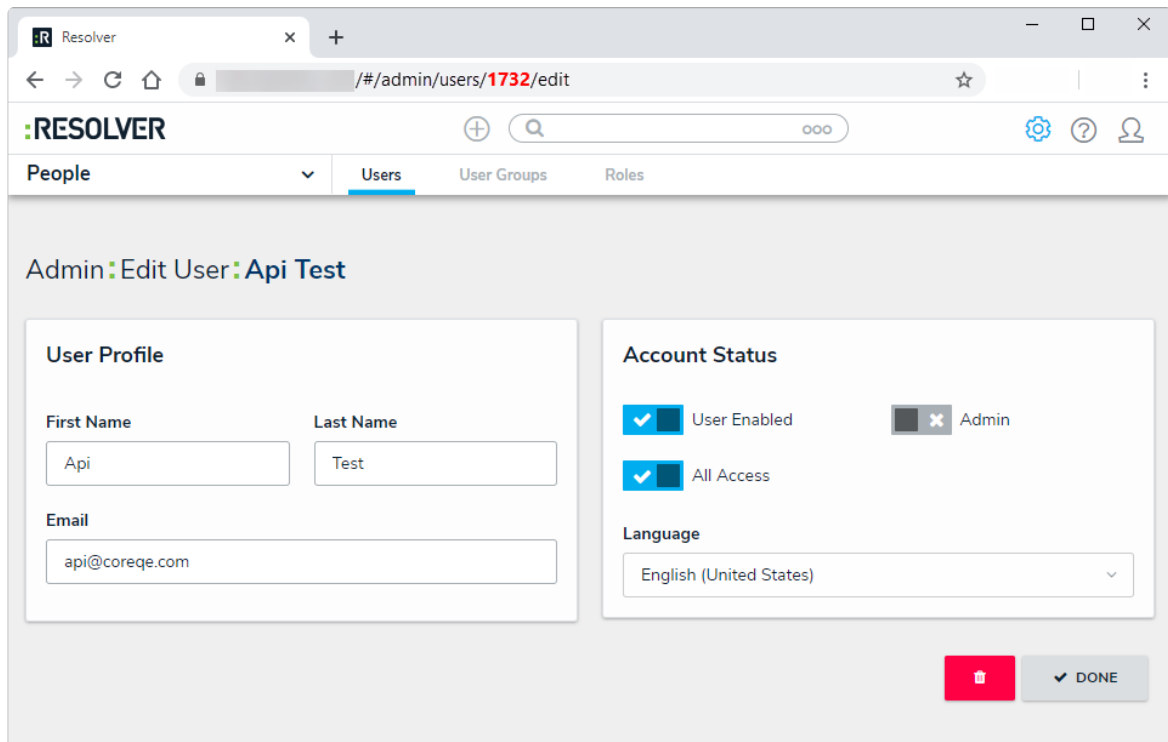
To create an API key:

1. Login as an admin and select the appropriate org, if required.
2. Click the  icon in the top bar > **Users** in the **People** section.
3. Click **Create User**.
4. Enter the user's name in the **First Name** and **Last Name** field.
5. Enter an email address in the **Email** field. If the API key is for a production environment, enter a valid company email address.
6. From the **Edit User** page, click the  icon next to **All Access** to allow the user to view, edit, and delete all data in the org. Alternatively, you may add the user to a [role](#) or [user group](#).
7. **Optional:** Click the  icon next to **Admin** to enable administrative rights.




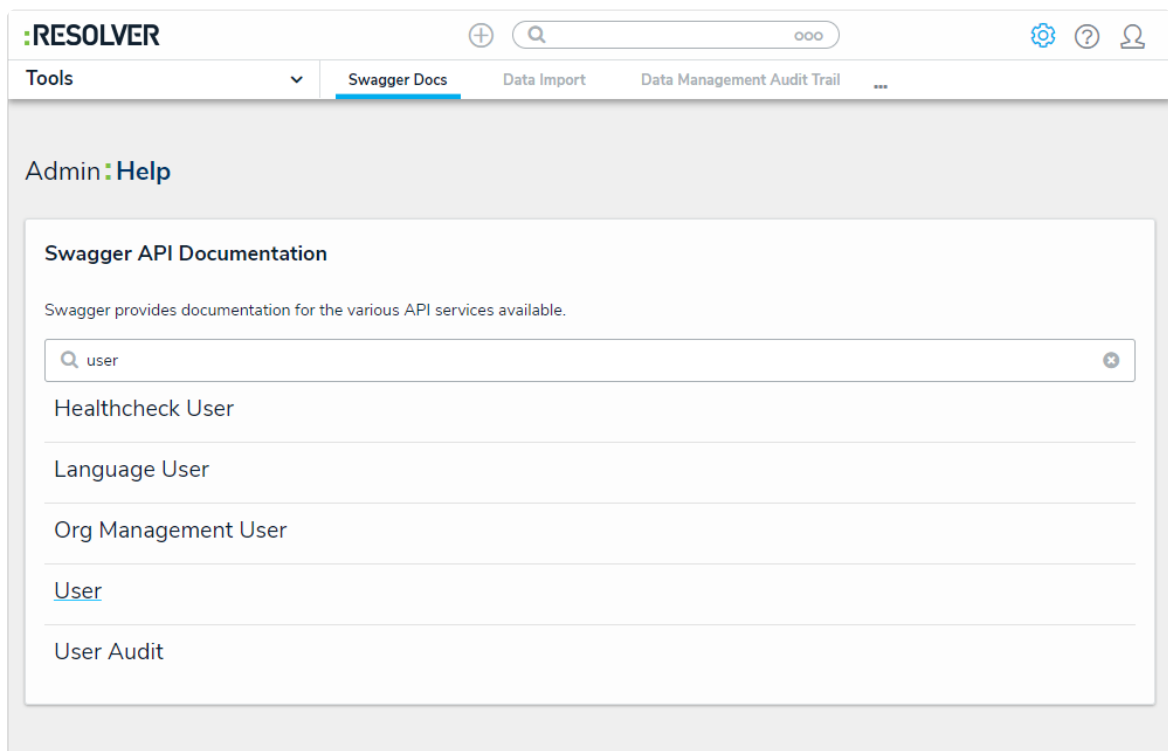
For added security, it's recommended that the **Admin** setting is not enabled for API users unless admin privileges are needed to complete the required API calls.

8. Click **Create**.
9. From the **Edit User** page, record the account's internal ID from the address bar of your browser (e.g., 1732).



The Edit User page. The user's internal ID is displayed in the address bar.

10. Click the  icon in the top bar > **Swagger Docs** in the **Tools** section.
11. Enter the keyword **user** in the search text box, then click **User** from the results to open the Swagger interface in a new tab.



The Swagger API Documentation page.

12. Scroll down to view the **user** resource.
13. Click **GET /user/users/me (who am I?)** to expand it.

user		Show/Hide	List Operations	Expand Operations
GET	/user/users			retrieve a set of users by id
POST	/user/users			add a new user
GET	/user/users/addedToRole			Retrieve users that have been added to the given role
POST	/user/users/email			load User By Email without org
GET	/user/users/email/{email}			Check if users exists by email address.
GET	/user/users/me			who am i?
GET	/user/users/page			retrieve a page of users
GET	/user/users/removedFromRole			Retrieve users that have lost access to the given role
GET	/user/users/updatedByRole			Retrieve updated users from the given role
DELETE	/user/users/{id}			delete a user
GET	/user/users/{id}			load users
PUT	/user/users/{id}			update a users
PUT	/user/users/{id}/membership			update a users membership
PUT	/user/users/{id}/termsOfService			update terms of service acceptance

The GET /user/users/me endpoint.

- Click **Try it out!**
- Record or copy the **id** number from the **currentOrg** section to your clipboard. This is the current org's internal ID.

```

Response Body
{
  "id": 353,
  "first": "John",
  "last": "Doe",
  "email": "john.doe@kroll.com",
  "modified": "2020-02-27T21:28:26.833Z",
  "externalRefId": "9a1a1c55-0179-4bb0-9722-0025c3194318",
  "isAdmin": true,
  "superAdmin": true,
  "acceptedTos": true,
  "lastLogin": "2020-02-27T21:28:26.833Z",
  "ssoBypass": false,
  "lang": "en-US",
  "isActive": true,
  "allAccess": true,
  "currentOrg": {
    "id": 92,
    "name": "Kroll (New & P)",
    "passwordExpiration": 100,
    "logoUrl": null,
  }
}

```

The Response Body section of the GET /user/users/me endpoint.

- Scroll up and expand the **org** resource to display the available endpoints.
- Click **GET /user/org/{orgId}/user/{userId} (load a user org membership)** endpoint to expand it.

The GET /user/org/{orgId}/user/{userId} endpoint.

18. Enter the org number copied during step 15 in the **orgId** field.
19. Enter the user ID, copied during step 9 from the **Edit User** page, in the **userId** field.

Parameter	Value	Description	Parameter Type	Data Type
orgId	92		path	integer
userId	1732		path	integer

The expanded GET /user/org/{orgId}/user{userId} endpoint.

20. Click **Try it out!**
21. From the **Response Body**, record or copy the **id** number to your clipboard. This is the user's org membership ID number.

The Response Body section of the GET /user/org/{orgId}/user/{userId} endpoint.

22. Scroll up and expand the **apiKey** resource to display the available endpoints.
23. Click **POST /user/apiKey (create an api key)** endpoint to expand it.

apiKey		Show/Hide	List Operations	Expand Operations
POST	/user/apiKey			create an api key
GET	/user/apiKey/user/{userId}			retrieve an api key by userId
DELETE	/user/apiKey/{id}			delete an api key
GET	/user/apiKey/{id}			retrieve an api key by id

The POST /user/apiKey (create an api key) endpoint.

24. In the **Parameters** section, click the **Example Value** box to populate the template in the **body** text box.

apiKey		Show/Hide	List Operations	Expand Operations
POST	/user/apiKey			create an api key
Response Class (Status 200)				
OK				
Model Example Value				
<pre>{ "id": 0, "apiKey": "string" }</pre>				
Response Content Type: application/json				
Parameters				
Parameter	Value	Description	Parameter Type	Data Type
body	<input type="text"/>		body	Model Example Value
	Parameter content type: application/json			<pre>{ "orgUserId": 0, "name": "string", "canImpersonate": false }</pre>
Try it out!				

The Example Value box.

25. In the **body** text box, delete the **0** in the **orgUserId** attribute, then enter the user's org membership ID number, obtained in step 21.
26. Enter a descriptive name for the API key in the **name** attribute (e.g., Integration Account).

Parameters		Show/Hide	List Operations	Expand Operations
Parameter	Value	Description	Parameter Type	Data Type
body	<pre>{ "orgUserId": 2257, "name": "Integration Account", "canImpersonate": false }</pre>		body	Model Example Value
	Parameter content type: application/json			<pre>{ "orgUserId": 0, "name": "string", "canImpersonate": false }</pre>
Try it out!				

A completed body text box.



The **canImpersonate** attribute makes it possible to impersonate another user while using the API key; however, only members of Resolver Support can enable impersonation through the API. Otherwise, setting this attribute to **true** will result in an error. Contact [Resolver Support](#) should you wish to enable this feature. For more information on impersonating a user with an API key, see [Impersonation with an API Key](#).

27. Click **Try it out!**

28. Copy the **apiKey** from the **Response Body** and store it for safekeeping.

Response Body

```
{
  "id": 125,
  "apiKey": "cSqzQ0Cud0Hp7AZ4k7at_EU0U11S4BCqo2tH73x/QAuTKR9qegaK4mHaxBcpv8dTqrGrugLH1w"
}
```


The Response Body displaying the ID and API key.



For security purposes, once an API key is generated, it cannot be retrieved. If you misplace an API key, a new key must be generated.

Delete an API Key

To delete an API key:

1. Log in as an admin and select the appropriate org, if required.
2. Click the  icon in the top bar > **Swagger Docs** in the **Tools** section.
3. Click any resource to open the Swagger interface in a new tab.
4. Click the **apiKey** service to display its endpoints.
5. Click **GET /user/apiKey/{id} (retrieve an api key by id)** to expand it.

apiKey		Show/Hide	List Operations	Expand Operations
POST	/user/apiKey			create an api key
GET	/user/apiKey/user/{userId}			retrieve an api key by userId
DELETE	/user/apiKey/{id}			delete an api key
GET	/user/apiKey/{id}			retrieve an api key by id

The apiKey service.

6. Enter the user ID of the account the API key was created under in the **id** field. The user ID can be obtained from the address bar of your browser after navigating to the **Edit User** page for the user.

GET /user/apiKey/{id} retrieve an api key by id

Response Class (Status 200)
OK

Model | Example Value

```

{
  "id": 0,
  "orgUserId": 0,
  "name": "string",
  "create": "2020-03-13"
}

```

Response Content Type

Parameters

Parameter	Value	Description	Parameter Type	Data Type
id	<input type="text" value="1732"/>		path	integer

The GET /user/apiKey/{id} (retrieve an api key by id) endpoint.

- Click **Try it out!**
- Copy the **id** from the **Response Body** to your clipboard. This is the internal ID for the API key.

Response Body

```

{
  "data": [
    {
      "id": 130,
      "orgUserId": 93768,
      "name": "API Token",
      "accessKey": "48mw0qSsANirPIrrgJRh",
      "canImpersonate": false,
      "created": "2020-02-20T21:33:07.303Z"
    }
  ]
}

```

The id in the Response Body.

- Click **DELETE /user/apiKey/{id} (delete an api key)** to expand it.
- Paste the internal API key ID obtained from step 8 above in the **id** field.

DELETE /user/apiKey/{id} delete an api key

Parameters

Parameter	Value	Description	Parameter Type	Data Type
id	<input type="text" value="130"/>		path	integer

Response Messages

HTTP Status Code	Reason	Response Model	Headers
204	Delete success		

The DELETE /user/apiKey/{id} (delete an api key) endpoint.

- Click **Try it out!** to delete the API key.

IP Token Validation

IP token validation protects the Core API (Swagger) from unauthorized users and can be used in conjunction with IP authorization to manage organizational access. The token validation process

accomplishes this by periodically validating the IP associated with the token. If there is any change to the IP during the session, the user will be logged out and asked to log in again.

IP token validation can be used with one of the following options:

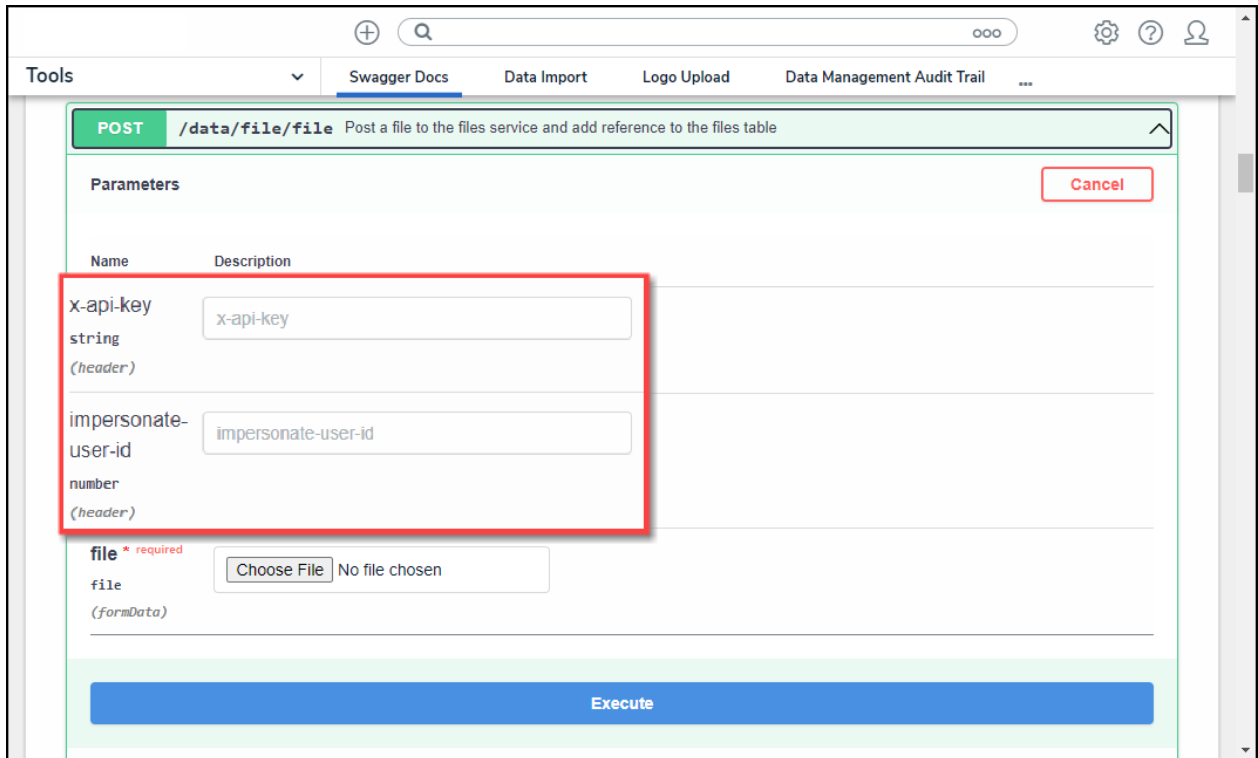
- **off:** No IP token validation is performed for any authentication requests. This is the default setting for IP token validation.
- **on:** IP token validation is performed for all SSO and basic API access requests.

The above options can be enabled and disabled by [Resolver Support](#). See the [IP Authorization Logins](#) article for login functionality when this feature is enabled.

Make Calls

To use an API key in the Swagger interface:

1. Log in as an admin and select the appropriate org, if required.
2. Navigate to **Admin > Swagger Docs**.
3. Click the resource from the list to open the Swagger interface in a new tab.
4. Expand a supported endpoint.
5. Enter the API key in the **x-api-key** field to authenticate the call.
6. **Optional:** If impersonation is enabled, enter the ID of the user who will be impersonated. Impersonation can only be enabled by a member of Resolver Support. See [Impersonation with an API Key](#) for more details.



*The POST /data/file/file endpoint in the **file** service*

7. Complete the remainder of the fields as required.

Example

```
curl -X POST --header 'Content-Type: multipart/form-data' --header 'Accept: application/json' --header 'x-api-key: YOUR_
```