Cortex Cloud Platform APIs

1 | Cortex Cloud APIs

1.1 | Understand Cortex Cloud licenses

This Cortex Cloud documentation is relevant for Cortex Cloud Runtime Security and Cortex Cloud Posture Management. It is important to check the license requirements for each API. Each API includes information on which licenses and add-ons are necessary to run the API.

To read more about Cortex Cloud licensing, see Understand license plans in Cloud Runtime Security and Understand license plans in Cloud Posture Management.

1.2 | Get started with Cortex Cloud APIs

Before you can begin using Cortex Cloud APIs, you must generate the following items in Cortex Cloud:

Value	Description
API Key	The API Key is your unique identifier used as the `Authorization:{key}` header required for authenticating API calls. Depending on your desired security level, you can generate two types of API keys, Advanced or Standard, from Cortex.
API Key ID	The API Key ID is your unique token used to authenticate the API Key. The header used when running an API call is `x-xdr-auth-id:{key_id}`.
FQDN	The FQDN is a unique host and domain name associated with each tenant.

Cortex Cloud API URIs are made up of your tenant's FQDN, the API name, and endpoint path. For example, `https://api-{fqdn}/xsiam/public/v1/{endpoint_path}/.`

The following steps describe how to generate the necessary key values and run your first API call:

1. Create a new API key.
2. Get your Cortex Cloud API key ID.
3. Get your FQDN.
4. Make your first API call.

1.3 | Create a new API key

1. In Cortex, navigate to **Settings** > **Configurations** > **Integrations** > **API Keys** > **New Key**.
2. In the **Role** tab, perform the following:
   1. Under **Security Level**, select the type of API Key you want to generate: **Advanced** or **Standard**. The Advanced API key hashes the key using a nonce, a random string, and a timestamp to prevent replay attacks. cURL does not support this but it is suitable with scripts.
   2. Under **Role**, select the desired level of access for this key. You can select from predefined roles or custom roles. Roles are available according to what was defined in either the Cortex Gateway or Cortex XSIAM Access Management. You can view the configuration of the role selected by expanding the sections under Components. For more information, see Assign user roles and groups.
   3. (Optional) Under **Comment**, provide a comment that describes the purpose of the API key.
   4. (Optional) If you want to define a time limit on the API key authentication, select **Enable Expiration Date**, and select the expiration date and time. You can track the expiration date of each API key in the **API Keys** page. In addition, Cortex Cloud displays a API Key Expiration notification in the Notification Center one week and one day prior to the defined expiration date.
3. (Optional) To configure and manage granular scoping for Scope-Based Access Control (SBAC), click the **Scope** tab, and under **Scope Definition**, expand the scoping areas that you want to grant the user role access to for this API by clicking the chevron icon (>) beside the scoping area title. For more information, see Manage API keys.
4. **Generate** the API Key.
5. Copy the API key, and then click **Close**. This value represents your unique `Authorization:{key}`.

   warning

   You will not be able to view the API Key again after you complete this step. Ensure that you copy it before closing the notification.

1.4 | Get your Cortex API key ID

1. In the API Keys table, locate the **ID** field.
2. Note your corresponding **ID** number. This value represents the `x-xdr-auth-id:{key_id}` token.

1.5 | Get your FQDN

1. In Cortex Cloud, navigate to Settings > Configurations > Integrations > API Keys.
2. Click Copy API URL.
3. Your FQDN is saved in the clipboard. You must use the FQDN as part of the URL in every API call. For example: `https://api-company.us.com/xsoar/public/v1/{endpoint_path}/`.

You can use the CURL Example URL to run the APIs.

1.6 | Make your first API call

The following examples vary depending on the type of key you select.

You can test authentication with Advanced API keys using the provided Python 3 example. With Standard API keys, use either the cURL example or the Python 3 example. Don't forget to replace the example variables with your unique API key, API key ID, and FQDN tenant. After you verify authentication, you can begin making API calls.

Standard Key cURL Example

```
curl -X POST https://api-{fqdn}/public_api/v1/{name of api}/{name of call}/
-H "x-xdr-auth-id:{key_id}"
-H "Authorization:{key}"
-H "Content-Type:application/json"
-d '{}'
```

Standard Key Python 3 Example

```
import requests
    def test_standard_authentication(api_key_id, api_key):
    headers = {
        "x-xdr-auth-id": str(api_key_id),
        "Authorization": api_key
    }
    parameters = {}
    res = requests.post(url="https://api-{fqdn}/public_api/v1/{name of api}/{name of call}",
                                                headers=headers,
                                                json=parameters)
    return res
```

Advanced Key Python 3 Example

```
import requests

from datetime import datetime, timezone
import secrets
import string
import hashlib
import requests

def test_advanced_authentication(api_key_id, api_key):
   # Generate a 64 bytes random string
    nonce = "".join([secrets.choice(string.ascii_letters + string.digits) for _ in range(64)])
    # Get the current timestamp as milliseconds.
    timestamp = int(datetime.now(timezone.utc).timestamp()) * 1000
    # Generate the auth key:
    auth_key = "%s%s%s" % (api_key, nonce, timestamp)
    # Convert to bytes object
    auth_key = auth_key.encode("utf-8")
    # Calculate sha256:
    api_key_hash = hashlib.sha256(auth_key).hexdigest()
    # Generate HTTP call headers
    headers = {
        "x-xdr-timestamp": str(timestamp),
        "x-xdr-nonce": nonce,
        "x-xdr-auth-id": str(api_key_id),
        "Authorization": api_key_hash
    }
    parameters = {}
    res = requests.post(url="https://api-{fqdn}/public_api/v1/{name of api}/{name of call}",
                                                headers=headers,
                                                json=parameters)
    return res
```

1.7 | Run XQL query APIs

Cortex Cloud enables you to run XQL queries on your data sources using a series of APIs. To execute XQL APIs you must have:

- Valid API Key and API Key ID that include the Instance Administrator role permissions.
- Available query quota.

Query quota is made up of query units that enable you to run XQL APIs. Each XQL API query entails a cost of query units calculated according to the complexity and number of search results. The query cost for each API query is displayed in the **Get Query Results** API. You can also track the query cost per XQL API search, overall usage, and remaining quota in Cortex Cloud or by running a **Get XQL Query Quota** API. Cortex Cloud provides a free daily quota relative to your license size for you to run XQL API queries.

> info
>
> **Note**: You will be able to purchase additional query units in future Cortex Cloud versions.

To execute a XQL API, you need to run a series of APIs. Each API requires a response value from the previous API to continue. This allows you to track the number of XQL queries you want to run, which in turn helps you manage your daily quota. Queries called without enough quota will fail. To ensure you don't surpass your quota, Cortex Cloud allows you to run up to four API queries in parallel.

Run the following APIs to call an XQL query:

1. **Start an XQL Query**â€ Run an XQL query. Response returns a unique execution ID used to retrieve the results by the **Get XQL Query Results** API.
2. **Get XQL Query Results**â€ Retrieve XQL query results. API displays up to 1,000 results. If query generated more than 1,000 results, the response returns a unique stream ID used to retrieve additional results by the **Get XQL Query Results Stream** API.
3. **Get XQL Query Results Stream**â€ Retrieve XQL query with more than 1,000 results.

1.8 | Whats-new-in-this-release

What's new

Cortex Cloud version 1.3 includes the following new APIs.

Product Name	API Details

Application Security	Remediation APIs
	`/public_api/appsec/v1/issues/fix/trigger_fix_pull_request`
	`/public_api/appsec/v1/issues/fix/{issueId}/fix_suggestion`
	`/public_api/appsec/v1/issues/fix/{remediationId}`
	
	Criteria APIs
	`/public_api/appsec/v1/application/criteria`
	`/public_api/appsec/v1/application/criteria/all`
	`/public_api/appsec/v1/application/criteria/{criteriaId}`

Cloud Onboarding	`/public_api/v1/cloud_onboarding/get_identifiers`
Compliance	`/public_api/v1/compliance/get_assets`
Cortex Cloud Platform	Identity and Access Management (IAM) APIs `/platform/iam/v1/role` (GET, POST) `/platform/iam/v1/role/{role_id}` (DELETE) `/platform/iam/v1/role/permission-config` (GET) `/platform/iam/v1/user-group` (GET, POST) `/platform/iam/v1/user-group/{group_id}` (PATCH, DELETE) `/platform/iam/v1/scope/{entity_type}/{entity_id}` (GET, PUT) `/platform/iam/v1/user` (GET) `/platform/iam/v1/user/{user_email}` (GET, PATCH) `/platform/iam/v1/api-key/{api_key_id}` (GET, PUT)
Cloud CIEM	No new APIs in this release.
Compute (CWP)	No new APIs in this release.
DSPM	No new APIs in this release.
UVEM	No new APIs in this release.
Vulnerability	No new APIs in this release.

What's changed with version 1.3

Cortex Cloud version 1.3 includes the following updates.

Product Name	API And Schema Changes
Application Security	Schema Additions - ApplicationMetadataCloud - AppsecPolicyTriggersandActions - CloudCriteriaConfig - CreateCriteriaRequest - CreateCriteriaResponse - CriteriaCreationMethod - CriteriaDTOCloud - CriteriaDefinition - CriteriaDefinitionType - CriteriaMetadata - CriteriaType - CriteriaType.Cloud - FixStatus - GetIssueFixSuggestionResponse - PaginationResponse_CriteriaDTO_ - PolicyCondition - PolicyScope - RefreshStatus - SaveFixResult - SectionProvider - TriggerFixPrPayload - TriggerFixPrResponse - TriggeredPr Schema Updates - ConditionOperators: Updated enum `WILDCARD_NOT` â `WILDCARDNOT` - CortexCondition: `SEARCH_FIELD`, `SEARCH_TYPE`, `SEARCH_VALUE` made required - CreateRequest: Removed `enabled`, `overrideIssueSeverity`; added `assetGroupIds` - ExtendedFields: Properties and required fields updated - PolicyConditions & PolicyScope: Updated with available fields - PolicyTriggers: Updated with `details` property
Cloud Onboarding	Schema Additions - GetIdentifiersRequestData - GetIdentifiersResponse Schema Updates - InstanceAuditLogsConfig: Added new properties and updated required fields - InstanceAdditionalCapabilities: Added new properties and updated required fields - CreateInstanceTemplateRequestData: Added enum `OCI` to `cloud_provider`, added `scan_env_id`, updated required fields - EditInstanceRequestData, CreateOutpostTemplateRequestData, ListCloudProviderRegionsRequestData, ListCloudProviderRegionsResponse: Added enum `OCI` to `cloud_provider`
Cortex Cloud Platform	- Endpoint category renamed to Endpoint Management. - Issues endpoints updated to include `/public_api/` prefix for consistency with other API endpoints
Cloud CIEM	No updates in this release.
Compliance	Schema Additions - AssessmentProfileSortObject - AssessmentResultFilterObject - AssetFilterObject - AssetSortObject - AssetsResponse - ComplianceAsset - GetAssetsRequest Schemas Removed - FilterObject - SortObject
Compute (CWP)	No updates in this release.
DSPM	No updates in this release.
UVEM	No updates in this release.

Product Name	API And Schema Changes

Vulnerability	Added pagination properties to the following POST requests:
	`/public_api/uvem/v1/get_vulnerabilities`
	`/public_api/uvem/v1/get_affected_software`

2 | APIs

2.1 | Cortex Cloud Platform APIs

2.1.1 | Preface

You can export the OpenAPI specs in the Cortex Stoplight instance.

Using the Cortex Platform APIs, you can configure and manage authentication settings, datasets, cases, issues, and script executions. Additionally, the APIs allow you to run XQL queries and perform various other operations across the platform.

The license requirements for each API are listed individually.

Version: Cortex Cloud

2.1.1.1 | Servers

https://api-yourfqdn

2.1.2 | Cortex CLI

APIs for managing the Cortex CLI

2.1.2.1 | Get the latest version of the Cortex CLI

get /public_api/v1/cli/releases/version

Get the latest version of Cortex CLI

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'GET'
-H 'Accept: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/cli/releases/version
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE" } conn.request("GET", "/public_api/v1/cli/releases/version",
headers=headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/cli/releases/version") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Get.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' response = http.request(request) puts
response.read_body
const data = null; const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("GET", "https://api-yourfqdn/public_api/v1/cli/releases/version");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.send(data);
HttpResponse<String> response = Unirest.get("https://api-yourfqdn/public_api/v1/cli/releases/version")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE" ] let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/cli/releases/version")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "GET" request.allHTTPHeaderFields =
headers let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/cli/releases/version", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "GET", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "GET"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/cli/releases/version"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/cli/releases/version"); var request = new
RestRequest(Method.GET); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); IRestResponse response = client.Execute(request);
```

Responses

Success

Body
application/json
▼ versionstring

Cortex CLI version

Example:`"v1.2.3"`
RESPONSE

```
{ "version": "v1.2.3" }
```

Service had unexpected internal error

2.1.3 | XQL query

Run XQL queries on your data sources using a series of APIs.

2.1.3.1 | Start an XQL query

post /public_api/v1/xql/start_xql_query

Execute an XQL query.

For more information on how to run XQL queries, see *Running XQL query APIs*.

> Note
>
> To ensure you don't surpass your quota, Cortex Cloud allows you to run up to four API queries in parallel.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/start_xql_query'
```

```
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"query\":\"dataset=xdr_data | fields event_id, event_type, event_sub_type | limit 3\",\"tenants\":[],\"timeframe\":
{\"from\":1598907600000,\"to\":1599080399000}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/xql/start_xql_query",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/xql/start_xql_query")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"query\":\"dataset=xdr_data | fields
event_id, event_type, event_sub_type | limit 3\",\"tenants\":[],\"timeframe\":
{\"from\":1598907600000,\"to\":1599080399000}}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "query": "dataset=xdr_data | fields event_id, event_type,
event_sub_type | limit 3", "tenants": [], "timeframe": { "from": 1598907600000, "to": 1599080399000 } } }); const xhr
= new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/xql/start_xql_query"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/start_xql_query")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"query\":\"dataset=xdr_data | fields event_id, event_type,
event_sub_type | limit 3\",\"tenants\":[],\"timeframe\":{\"from\":1598907600000,\"to\":1599080399000}}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "query": "dataset=xdr_data | fields event_id,
event_type, event_sub_type | limit 3", "tenants": [], "timeframe": [ "from": 1598907600000, "to": 1599080399000 ] ]]
as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/xql/start_xql_query")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/start_xql_query", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"query\":\"dataset=xdr_data | fields event_id, event_type, event_sub_type |
limit 3\",\"tenants\":[],\"timeframe\":{\"from\":1598907600000,\"to\":1599080399000}}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/start_xql_query"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"query\":\"dataset=xdr_data | fields event_id, event_type, event_sub_type | limit 3\",\"tenants\":[],\"timeframe\":
{\"from\":1598907600000,\"to\":1599080399000}}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/start_xql_query"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"query\":\"dataset=xdr_data | fields event_id, event_type, event_sub_type | limit
3\",\"tenants\":[],\"timeframe\":{\"from\":1598907600000,\"to\":1599080399000}}}", ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired
▶ querystring

String of the XQL query.

▶ tenantsarray[string]

Note: This is only used when querying tenants managed by Managed Security Services Providers (MSSP).

List of strings used for running APIs on local and Managed Security tenants. Valid values:

- For single tenant (local tenant) query, enter a single-item list with your tenant_id. Additional valid values are, empty list ([]) or null (default).
- For multi-tenant investigations (Managed Security parent who investigate children and/or local), enter multi-item list with the required tenant_id. List of IDs can contain the parent, children, or both parent and children.

▶ timeframeobject

Integer in timestamp epoch milliseconds. Valid values include:

- Absolute Unix timestamp representing a date period: {"from" : 1598907600000, "to" : 1599080399000} = date period: 31/08/20 09:00:00 PM UTC - 02/09/20 8:59:59 PM UTC
- Relative Unix timestamp representing the last 24 hours: {"relativeTime": 86400000} = (24 * 60 * 60 * 1000 = 86400000).

REQUEST default

```
{ "request_data": { "query": "dataset=xdr_data | fields event_id, event_type, event_sub_type | limit 3", "tenants": [], "timeframe": { "from": 1598907600000, "to": 1599080399000 } } }
{ "request_data": { "query": "dataset=xdr_data | fields event_id, event_type, event_sub_type | limit 3", "tenants": [ "431509831", "401387390" ], "timeframe": { "from": 1598907600000, "to": 1599080399000 } } }
```

Responses

Successful response

Body
application/json
▼ replystring
RESPONSE Example 1

```
{ "reply": "ad21c1e1492d4c_667_inv" }
```

Bad Request. Invalid JSON.

Body
application/json
▼ replyobject
▶ err_codeinteger
▶ err_msgstring
▶ err_extraobject
RESPONSE Example 1

```
{ "reply": { "err_code": 500, "err_msg": "An error occurred while processing XDR - XQL query", "err_extra": { "err_msg": "reached max allowed amount of parallel running queries. please wait for some queries to finish and submit your query again", "query_cost": 0, "remaining_quota": 5, "total_daily_running_queries": 4, "total_daily_concurrent_rejected_queries": 1 } } }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json
▼ replyobject
▶ err_codeinteger
▶ err_msgstring
▶ err_extraobject
RESPONSE Example 1

```
{ "reply": { "err_code": 500, "err_msg": "An error occurred while processing XDR - XQL query", "err_extra": { "err_msg": "reached max allowed amount of parallel running queries. please wait for some queries to finish and submit your query again", "query_cost": 0, "remaining_quota": 5, "total_daily_running_queries": 4, "total_daily_concurrent_rejected_queries": 1 } } }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json
▼ replyobject
▶ err_codeinteger
▶ err_msgstring
▶ err_extraobject
RESPONSE Example 1

```
{ "reply": { "err_code": 500, "err_msg": "An error occurred while processing XDR - XQL query", "err_extra": { "err_msg": "reached max allowed amount of parallel running queries. please wait for some queries to finish and submit your query again", "query_cost": 0, "remaining_quota": 5, "total_daily_running_queries": 4, "total_daily_concurrent_rejected_queries": 1 } } }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

▼ replyobject
▶ err_codeinteger
▶ err_msgstring
▶ err_extraobject
RESPONSE Example 1
```
{ "reply": { "err_code": 500, "err_msg": "An error occurred while processing XDR - XQL query", "err_extra": {
"err_msg": "reached max allowed amount of parallel running queries. please wait for some queries to finish and submit
your query again", "query_cost": 0, "remaining_quota": 5, "total_daily_running_queries": 4,
"total_daily_concurrent_rejected_queries": 1 } } }
```

Internal server error. A unified status for API communication type errors.

Body
application/json
▼ replyobject
▶ err_codeinteger
▶ err_msgstring
▶ err_extraobject
RESPONSE Example 1
```
{ "reply": { "err_code": 500, "err_msg": "An error occurred while processing XDR - XQL query", "err_extra": {
"err_msg": "reached max allowed amount of parallel running queries. please wait for some queries to finish and submit
your query again", "query_cost": 0, "remaining_quota": 5, "total_daily_running_queries": 4,
"total_daily_concurrent_rejected_queries": 1 } } }
```

2.1.3.2 | Get XQL query results

post /public_api/v1/xql/get_query_results

Retrieve results of an executed XQL query API.

Note: This endpoint only works on XQL queries initiated by `/public_api/v1/xql/start_xql_query/`.

Maximum result set size is 1000. The API does not support pagination, therefore, you can set values to determine the result size limitation and how to wait for the results. To view response with greater than 1000 results you must call Get XQL query results Stream.

For more information on how to run XQL queries, see *Running XQL query APIs*.

> Note
>
> To ensure you don't surpass your quota, Cortex Cloud allows you to run up to four API queries in parallel.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/get_query_results'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"query_id\":\"061880b4867446_4356_inv\",\"pending_flag\":true,\"limit\":100,\"format\":\"json\"}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/xql/get_query_results", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/xql/get_query_results") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
```

```
request.body = "{\"request_data\":
{\"query_id\":\"061880b4867446_4356_inv\",\"pending_flag\":true,\"limit\":100,\"format\":\"json\"}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "query_id": "061880b4867446_4356_inv", "pending_flag": true, "limit":
100, "format": "json" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/get_query_results");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/get_query_results")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"query_id\":\"061880b4867446_4356_inv\",\"pending_flag\":true,\"limit\":100,\"format\":\"json\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "query_id": "061880b4867446_4356_inv",
"pending_flag": true, "limit": 100, "format": "json" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/get_query_results")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/get_query_results", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"query_id\":\"061880b4867446_4356_inv\",\"pending_flag\":true,\"limit\":100,\"format\":\"json\"}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/get_query_results"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"query_id\":\"061880b4867446_4356_inv\",\"pending_flag\":true,\"limit\":100,\"format\":\"json\"}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/get_query_results"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":
{\"query_id\":\"061880b4867446_4356_inv\",\"pending_flag\":true,\"limit\":100,\"format\":\"json\"}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ query_idstring

String representing the unique execution ID generated by the response to Start an XQL query API. You can also enter the execution ID of a query generated in Cortex XDR and listed in the Query Center table.

▶ pending_flagboolean

Boolean flag indicating whether the API call should operate in synchronous/blocking mode, or in asynchronous/non-blocking mode. Valid Values:

- True (default): The call returns immediately with one of the following options:

1. PENDING status indicating query hasn't yet completed or results are not yet ready to be returned. Need to execute the API call again.
2. SUCCESS/FAIL status

- False: The API will block until query completes and results are ready to be returned.

▶ limitinteger

Integer representing the maximum number of results to return. If the 'limit' is not specified or if 'limit' is greater than 1000 and the query yields more than 1000 valid results, a `stream id` will be generated for use in the *Get XQL query results Stream** API. In the context of multi-tenant investigations, when you specify the parameter value (x), it will return x results across all tenants combined, rather than x results for each individual tenant. For example, if there are y tenants participating in the investigation, the maximum number of results returned can be x*y (up to the limit of 1,000,000).

▶ formatstring (Enum)

The type of response output.

REQUEST default

```
{ "request_data": { "query_id": "061880b4867446_4356_inv", "pending_flag": true, "limit": 100, "format": "json" } }
```

```
{ "request_data": { "query_id": "061880b4867446_4356_inv", "pending_flag": true, "limit": 100, "format": "json" } }
```

Responses

Successful response

Body
application/json

▼ replyobject
▶ statusstring
▶ number_of_resultsinteger
▶ query_costobject
▶ remaining_quotanumber
▶ resultsobject

RESPONSE pending_flag=true

```
{ "reply": { "status": "PENDING" } }
{ "reply": { "status": "SUCCESS", "number_of_results": 3, "query_cost": { "tenant_id_1": 0.001596388888888889 },
"remaining_quota": 4.998403611111111, "results": { "data": [ { "event_id": "eventID1", "_vendor": "PANW", "_product":
"Fusion", "insert_timestamp": 1621541825324, "_time": 1621541523000, "event_type": "STORY", "event_sub_type": "NULL"
}, { "event_id": "eventID2", "_vendor": "PANW", "_product": "Fusion", "insert_timestamp": 1621541825326, "_time":
1621541528000, "event_type": "STORY", "event_sub_type": "NULL" }, { "event_id": "eventID3", "_vendor": "PANW",
"_product": "Fusion", "insert_timestamp": 1621541825325, "_time": 1621541517000, "event_type": "STORY",
"event_sub_type": "NULL" } ] } } }
{ "reply": { "status": "SUCCESS", "number_of_results": 6, "query_cost": { "tenant_id_1": 0.001596388888888889,
"tenant_id_2": 0.00179989 }, "remaining_quota": 4.995007332222222, "results": { "data": [ { "event_id": "eventID1",
"_vendor": "PANW", "_product": "Fusion", "insert_timestamp": 1621541825324, "_time": 1621541523000, "event_type":
"STORY", "event_sub_type": "NULL", "tenant": "1723879655" }, { "event_id": "eventID2", "_vendor": "PANW", "_product":
"Fusion", "insert_timestamp": 1621541825326, "_time": 1621541528000, "event_type": "STORY", "event_sub_type": "NULL",
"tenant": "1723879655" }, { "event_id": "eventID3", "_vendor": "PANW", "_product": "Fusion", "insert_timestamp":
1621541825325, "_time": 1621541517000, "event_type": "STORY", "event_sub_type": "NULL", "tenant": "1723879655" }, {
"event_id": "eventID4", "_vendor": "PANW", "_product": "Fusion", "insert_timestamp": 1621541825324, "_time":
1621541523000, "event_type": "STORY", "event_sub_type": "NULL", "tenant": "1705396706" }, { "event_id": "eventID5",
"_vendor": "PANW", "_product": "Fusion", "insert_timestamp": 1621541825326, "_time": 1621541528000, "event_type":
"STORY", "event_sub_type": "NULL", "tenant": "1705396706" }, { "event_id": "eventID6", "_vendor": "PANW", "_product":
"Fusion", "insert_timestamp": 1621541825325, "_time": 1621541517000, "event_type": "STORY", "event_sub_type": "NULL",
"tenant": "1705396706" } ] } } }
{ "reply": { "status": "SUCCESS", "number_of_results": 1000000, "query_cost": { "tenant_id_1": 0.011742777777777777 },
"remaining_quota": 4.984442777777778, "results": { "stream_id": "streamID" } } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.3.3 | Get XQL query Quota

post /public_api/v1/xql/get_quota

Retrieve the amount of query quota available and used.

Note: This endpoint only works on XQL queries initiated by `/public_api/v1/xql/start_xql_query/`.

For more information on how to run XQL queries, see *Running XQL query APIs*.

> Note
>
> To ensure you don't surpass your quota, Cortex Cloud allows you to run up to four API queries in parallel.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/get_quota'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/xql/get_quota", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/xql/get_quota") http
= Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{}}" response = http.request(request)
puts response.read_body
const data = JSON.stringify({ "request_data": {} }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/get_quota");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/get_quota")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": []] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/get_quota")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/get_quota", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-
type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/get_quota"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/get_quota"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
Object
application/json
REQUEST default

```
{ "request_data": {} }
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ license_quotainteger
▶ additional_purchased_quotainteger
▶ used_quotanumber
▶ eval_quotainteger
▶ total_daily_running_queriesinteger

The number of daily active queries. This value is reset nightly.

▶ total_daily_concurrent_rejected_queriesinteger

The number of daily queries rejected due to too many concurrent XQL queries being run through the API. This value is reset nightly.

▶ current_concurrent_active_queriesobject

Currently running XQL queries with their current duration.

▶ current_concurrent_active_queries_countinteger

The number of active queries currently running.

▶ max_daily_concurrent_active_query_countinteger

The maximum number of queries that ran concurrently today on this tenant. This value is reset nightly.

RESPONSE example-1

```
{ "reply": { "license_quota": 5, "additional_purchased_quota": 0, "used_quota": 0, "eval_quota": 0,
"total_daily_running_queries": 4, "total_daily_concurrent_rejected_queries": 8, "current_concurrent_active_queries": {
"debee6b0c41f47_911_inv": { "xql": "config timeframe = 1mo | dataset=xdr_data | limit 1000000", "duration": 61 } },
"current_concurrent_active_queries_count": 1, "max_daily_concurrent_active_query_count": 4 } }
```

Bad Request. Got an invalid JSON.

Body
application/json
▼ replyobject
▶ err_codeinteger
▶ err_msgstring
▶ err_extraobject
RESPONSE Example 1

```
{ "reply": { "err_code": 500, "err_msg": "An error occurred while processing XDR - XQL query", "err_extra": {
"err_msg": "reached max allowed amount of parallel running queries. please wait for some queries to finish and submit
your query again", "query_cost": 0, "remaining_quota": 5, "total_daily_running_queries": 4,
"total_daily_concurrent_rejected_queries": 1 } } }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.3.4 | Get XQL query results Stream

post /public_api/v1/xql/get_query_results_stream

Retrieve XQL query results with more than 1000 results.

Note: This endpoint only works on XQL queries initiated by `/public_api/v1/xql/start_xql_query/`.

Response is returned as chunked (Transfer-Encoding: chunked). To retrieve a compressed gzipped response (Content-Encoding: gzip), in your header add Accept-Encoding: gzip.

For more information on how to run XQL queries, see *Running XQL query APIs*.

> Note
>
> To ensure you don't surpass your quota, Cortex Cloud allows you to run up to four API queries in parallel.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
▼ Accept-Encoding String

For retrieving a compressed gzipped response

Example: `acceptEncoding_example`
Default: `gzip`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example' -H 'Accept-Encoding:
acceptEncoding_example'
'https://api-yourfqdn/public_api/v1/xql/get_query_results_stream'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"stream_id\":\"563c5e24-===-9a1f8139d3c5\",\"is_gzip_compressed\":true}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'Accept-Encoding': "SOME_STRING_VALUE", 'content-type':
"application/json" } conn.request("POST", "/public_api/v1/xql/get_query_results_stream", payload, headers) res =
conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/xql/get_query_results_stream") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["Accept-Encoding"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"stream_id\":\"563c5e24-
===-9a1f8139d3c5\",\"is_gzip_compressed\":true}}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "stream_id": "563c5e24-===-9a1f8139d3c5", "is_gzip_compressed": true } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/get_query_results_stream"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("Accept-Encoding", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/get_query_results_stream") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("Accept-Encoding", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{\"stream_id\":\"563c5e24-===-9a1f8139d3c5\",\"is_gzip_compressed\":true}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "Accept-Encoding": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": [ "stream_id": "563c5e24-===-9a1f8139d3c5", "is_gzip_compressed": true ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/xql/get_query_results_stream")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/xql/get_query_results_stream", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"stream_id\":\"563c5e24-===-9a1f8139d3c5\",\"is_gzip_compressed\":true}}", CURLOPT_HTTPHEADER => [ "Accept-Encoding: SOME_STRING_VALUE", "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/xql/get_query_results_stream"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "Accept-Encoding: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"stream_id\":\"563c5e24-===-9a1f8139d3c5\",\"is_gzip_compressed\":true}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/get_query_results_stream"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("Accept-Encoding", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"stream_id\":\"563c5e24-===-9a1f8139d3c5\",\"is_gzip_compressed\":true}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ stream_idstring

String representing the unique ID generate by the response to Get XQL query results API.

▶ is_gzip_compressedboolean

A boolean flag.

REQUEST default

```
{ "request_data": { "stream_id": "563c5e24-===-9a1f8139d3c5", "is_gzip_compressed": true } }
```

Responses

Successful response

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4 | Endpoint Management

APIs for managing endpoints

2.1.4.1 | Get Distribution version

post /public_api/v1/distributions/get_versions

Get a list of all the agent versions to use for creating a distribution list.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/distributions/get_versions'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/distributions/get_versions", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/distributions/get_versions") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{}" response = http.request(request) puts response.read_body
const data = JSON.stringify({}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/distributions/get_versions"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/distributions/get_versions")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = [] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/distributions/get_versions")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/distributions/get_versions", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type:
application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/distributions/get_versions"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/distributions/get_versions"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
Object
application/json
REQUEST

```
{}
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ windowsarray[string]

List of Windows agent versions.

▶ linuxarray[string]

List of Linux agent versions.

▶ macosarray[string]

List of Mac agent versions.

RESPONSE

```
{ "reply": { "windows": [ "example" ], "linux": [ "example" ], "macos": [ "example" ] } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body

application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.2 | Get all Endpoints

post /public_api/v1/endpoints/get_endpoints

Gets a list of all of your endpoints. The response is concatenated using AND condition (OR is not supported).

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/get_endpoints'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/get_endpoints", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/get_endpoints") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{}" response = http.request(request) puts response.read_body
const data = JSON.stringify({}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/get_endpoints");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/get_endpoints")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = [] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/endpoints/get_endpoints")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/endpoints/get_endpoints", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/endpoints/get_endpoints"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/get_endpoints"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
Object
application/json
REQUEST

```
{}
```

Responses

Successful response

Body
application/json
▼ replyarray
[
▶ agent_idstring
▶ agent_statusstring
▶ operational_statusstring
▶ host_namestring
▶ agent_typestring
▶ iparray[string]
▶ last_seeninteger
▶ tagsobject
▶ usersarray[string]
]
RESPONSE example-1

```
{ "reply": [ { "agent_id": "<agent_id>", "agent_status": "DISCONNECTED", "operational_status": "PROTECTED", "host_name": "<hostname>", "agent_type": "Workstation", "ip": [ "<ip_address>" ], "last_seen": 1678012587521, "tags": { "server_tags": [], "endpoint_tags": [] }, "users": [ "user" ] } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body

application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.3 | Get Policy

post /public_api/v1/endpoints/get_policy

Get the policy name for a specific endpoint.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/get_policy'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"endpoint_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/endpoints/get_policy", payload, headers) res
= conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/get_policy") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"endpoint_id\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "endpoint_id": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/get_policy");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/get_policy")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"endpoint_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["endpoint_id": "string"]] as [String : Any]
let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/endpoints/get_policy")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/get_policy", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"endpoint_id\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/get_policy"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"endpoint_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/get_policy"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"endpoint_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ endpoint_idstring

Endpoint ID.

REQUEST example-1

```
{ "request_data": { "endpoint_id": "<endpoint ID>" } }
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ policy_namestring

Name of the policy allocated with the endpoint.

RESPONSE example-1

```
{ "reply": { "policy_name": "Windows Default" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.4 | Delete Endpoints

post /public_api/v1/endpoints/delete

Delete selected endpoints in Cortex Cloud. You can delete up to 1000 endpoints.

Note: Endpoints are deleted from Cortex Cloud UI, however they still exist in the database.

When filtering by multiple fields:

- Response is concatenated using AND condition (OR is not supported).
- Maximum result set size is 1000.
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/delete'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"string\",\"operator\":\"in\",\"value\":[\"string\"]}]}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/delete", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/endpoints/delete")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"string\",\"operator\":\"in\",\"value\":[\"string\"]}]}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "string", "operator": "in", "value": [
"string" ] } ] } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/delete");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/delete")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":[{\"field\":\"string\",\"operator\":\"in\",\"value\":
[\"string\"]}]}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["filters": [ [ "field": "string", "operator":
"in", "value": ["string"] ] ]]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters,
options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/endpoints/delete")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0)
request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session =
URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response,
error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/endpoints/delete", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"string\",\"operator\":\"in\",\"value\":[\"string\"]}]}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/endpoints/delete"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"string\",\"operator\":\"in\",\"value\":[\"string\"]}]}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/delete"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"filters\":[{\"field\":\"string\",\"operator\":\"in\",\"value\":[\"string\"]}]}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
▶ filtersarray

Array of filter fields.

REQUEST example-1

```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID>" ] } ] } }
```

Responses

Successful response

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.5 | Create distributions

post /public_api/v1/distributions/create

Create an installation package. This is an async call that returns the distribution ID; it does not mean that the creation succeeded. To confirm the package has been created, check the status of the distribution by running the **Get Distribution Status** API.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/distributions/create'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"name\":\"string\",\"platform\":\"windows\",\"package_type\":\"string\",\"agent_version\":\"string\",\"windows_version\":\"st
{\"property1\":null,\"property2\":null},\"proxy\":
[\"string\"],\"cluster_name\":\"string\",\"run_on_master_node\":true,\"run_on_all_nodes\":true}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/distributions/create", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/distributions/create") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":
{\"name\":\"string\",\"platform\":\"windows\",\"package_type\":\"string\",\"agent_version\":\"string\",\"windows_version\":\"st
{\"property1\":null,\"property2\":null},\"proxy\":
[\"string\"],\"cluster_name\":\"string\",\"run_on_master_node\":true,\"run_on_all_nodes\":true}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "name": "string", "platform": "windows", "package_type": "string",
"agent_version": "string", "windows_version": "string", "linux_version": "string", "macos_version": "string",
"deployment_platform": "string", "default_namespace": "string", "node_selector": { "property1": null, "property2":
null }, "proxy": [ "string" ], "cluster_name": "string", "run_on_master_node": true, "run_on_all_nodes": true } });
const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () {
if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/distributions/create"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/distributions/create")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"name\":\"string\",\"platform\":\"windows\",\"package_type\":\"string\",\"agent_version\":\"string\",\"windows_version\":\"st
{\"property1\":null,\"property2\":null},\"proxy\":
[\"string\"],\"cluster_name\":\"string\",\"run_on_master_node\":true,\"run_on_all_nodes\":true}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "name": "string", "platform": "windows",
"package_type": "string", "agent_version": "string", "windows_version": "string", "linux_version": "string",
"macos_version": "string", "deployment_platform": "string", "default_namespace": "string", "node_selector": [
"property1": , "property2": ], "proxy": ["string"], "cluster_name": "string", "run_on_master_node": true,
"run_on_all_nodes": true ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters,
options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/distributions/create")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0)
request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session =
URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response,
error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/distributions/create", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"name\":\"string\",\"platform\":\"windows\",\"package_type\":\"string\",\"agent_version\":\"string\",\"windows_version\":\"st
{\"property1\":null,\"property2\":null},\"proxy\":
[\"string\"],\"cluster_name\":\"string\",\"run_on_master_node\":true,\"run_on_all_nodes\":true}}", CURLOPT_HTTPHEADER
=> [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/distributions/create"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"name\":\"string\",\"platform\":\"windows\",\"package_type\":\"string\",\"agent_version\":\"string\",\"windows_version\":\"st
{\"property1\":null,\"property2\":null},\"proxy\":
[\"string\"],\"cluster_name\":\"string\",\"run_on_master_node\":true,\"run_on_all_nodes\":true}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/distributions/create"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":
{\"name\":\"string\",\"platform\":\"windows\",\"package_type\":\"string\",\"agent_version\":\"string\",\"windows_version\":\"st
{\"property1\":null,\"property2\":null},\"proxy\":
[\"string\"],\"cluster_name\":\"string\",\"run_on_master_node\":true,\"run_on_all_nodes\":true}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

▶ namestring

The name of the installation package.

▶ platformstring (Enum)

The installation platform.

▶ package_typestring

A string representing the type of package to create. Each JSON object must contain *one* of the following keywords:

- `standalone`: Installation for a new agent. When using this, you must include the `platform` field with one of the following values: `windows`, `linux`, `macos`, `android`, `kubernetes`, `helm`.
- `upgrade`: Upgrade of an agent from ESM. When using this, you must include the `agent_version` field with one of the following values: `windows_version`, `linux_version`, or `macos_version`.

▶ agent_versionstring

Use `agent_version` when creating a standalone installer. The value should be the agent version number.

▶ windows_versionstring

Use `windows_version` when creating an upgrade package. The value is the relevant version number.

▶ linux_versionstring

Use `linux_version` when creating an upgrade package. The value is the relevant version number.

▶ macos_versionstring

Use `macos_version` when creating an upgrade package. The value is the relevant version number.

▶ deployment_platformstring

When the `package_type` is `kubernetes` or `helm`, use the `deployment_platform` to indicate the type of platform. Valid values include:

- `standard`
- `openshift`
- `gcos`
- `bottlerocket`
- `gke_autopilot`

▶ default_namespacestring

The default namespace

▶ node_selectorobject

The node selector in the following format: `"node_selector": {"key": "val"}'

▶ proxyarray[string]
▶ cluster_namestring

Cluster name

▶ run_on_master_nodeboolean

Whether or not to run on the master node.

▶ run_on_all_nodesboolean

Whether or not to run on all nodes.

REQUEST New Installation example

```
{ "request_data": { "name": "<installation package name>", "platform": "windows", "package_type": "standalone", "agent_version": "windows_version" } }
{ "request_data": { "name": "<installation package name>", "package_type": "upgrade", "agent_version": "windows_version" } }
{ "request_data": { "name": "PAPI Dist K8s", "description": "Created using PAPI", "endpoint_tags": [ "new-tag" ], "package_type": "kubernetes", "platform": "linux", "agent_version": "8.8.0.10594", "deployment_platform": "standard", "default_namespace": "cortex-xdr", "node_selector": { "key": "val" }, "proxy": [ "10.10.10.1:8080" ], "cluster_name": "some_name", "run_on_master_node": true, "run_on_all_nodes": false } }
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ distribution_idstring

Installation package ID.

RESPONSE

```
{ "reply": { "distribution_id": "example" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body

application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.6 | Get Violations

post /public_api/v1/device_control/get_violations

Gets a list of device control violations filtered by selected fields. You can retrieve up to 100 violations.

When filtering by multiple fields:

- Response is concatenated using AND condition (OR is not supported).
- Maximum result set size is 100.
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/device_control/get_violations'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[0]}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id_list\",\"value\":\"asc\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-
id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/device_control/get_violations", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/device_control/get_violations") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[0]}],\"search_from\":0,\"search_to\":0,\"sort\":{\"field\":\"endpoint_id_list\",\"value\":\"asc\"}}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ 0 ] } ], "search_from": 0, "search_to": 0, "sort": { "field": "endpoint_id_list", "value": "asc" } } }); const xhr =
new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/device_control/get_violations"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/device_control/get_violations")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[0]}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id_list\",\"value\":\"asc\"}}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": [0] ] ], "search_from": 0, "search_to": 0, "sort": [ "field": "endpoint_id_list", "value":
"asc" ] ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/device_control/get_violations")!
as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST"
request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask
= session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error !=
nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } })
dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/device_control/get_violations", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[0]}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id_list\",\"value\":\"asc\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/device_control/get_violations"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[0]}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id_list\",\"value\":\"asc\"}}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/device_control/get_violations"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[0]}],\"search_from\":0,\"search_to\":0,\"sort\":{\"field\":\"endpoint_id_list\",\"value\":\"asc\"}}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject

An empty object returns all results.

▶ filtersarray

Provides an array of filter fields.

▶ search_frominteger

Integer representing the starting offset within the query result set from which you want violations returned. Violations are returned as a zero-based list. Any violation indexed less than this value is not returned in the final result set and defaults to zero.

▶ search_tointeger

An integer representing the end of offset within the result set after which you do not want violations returned. Violations in the violation list that are indexed higher than this value are not returned in the final results set. Defaults to zero, which returns all alerts to the end of the list.

▶ sortobjectrequired

Identifies the sort order for the result set.

REQUEST Request all results

```
{ "request_data": {} }
{ "request_data": { "filters": [ { "field": "type", "operator": "in", "value": [ "disk drive" ] } ], "search_to": 1 }
}
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ total_countinteger

Number of total results of this filter without paging.

▶ result_countinteger

Number of alerts actually returned as a result.

▶ violationsarray

RESPONSE

```
{ "reply": { "total_count": 0, "result_count": 0, "violations": [ { "hostname": "example", "username": "example",
"ip": "example", "timestamp": 0, "violation_id": 0, "type": "example", "vendor_id": "example", "vendor": "example",
"product_id": "example", "product": "example", "serial": "example", "endpoint_id": "example" } ] } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.7 | Get Distribution status

post /public_api/v1/distributions/get_status

Check the status of the installation package.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/distributions/get_status'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"distribution_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/distributions/get_status", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/distributions/get_status") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"distribution_id\":\"string\"}}" response = http.request(request) puts
response.read_body
```

```
const data = JSON.stringify({ "request_data": { "distribution_id": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/distributions/get_status"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/distributions/get_status")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"distribution_id\":\"string\"}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["distribution_id": "string"]] as [String :
Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/distributions/get_status")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields =
headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with:
request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else
{ let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/distributions/get_status", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"distribution_id\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/distributions/get_status"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"distribution_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/distributions/get_status"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"distribution_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ distribution_idstring

The installation package ID.

REQUEST
```
{ "request_data": { "distribution_id": "example" } }
```
Responses

Successful response

Body
application/json
▼ replyobject
▶ statusstring

The status of the installation package.

RESPONSE
```
{ "reply": { "status": "example" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.8 | Get Distribution URL

post /public_api/v1/distributions/get_dist_url

Get the distribution URL for downloading the installation package.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/distributions/get_dist_url'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"distribution_id\":\"string\",\"package_type\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-
xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/distributions/get_dist_url", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/distributions/get_dist_url") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
```

```
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"distribution_id\":\"string\",\"package_type\":\"string\"}}" response =
http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "distribution_id": "string", "package_type": "string" } }); const xhr
= new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/distributions/get_dist_url"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/distributions/get_dist_url")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"distribution_id\":\"string\",\"package_type\":\"string\"}}")
.asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "distribution_id": "string", "package_type":
"string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/distributions/get_dist_url")! as
URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST"
request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask
= session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error !=
nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } })
dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/distributions/get_dist_url", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"distribution_id\":\"string\",\"package_type\":\"string\"}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/distributions/get_dist_url"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"distribution_id\":\"string\",\"package_type\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/distributions/get_dist_url"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"distribution_id\":\"string\",\"package_type\":\"string\"}}", ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

▶ distribution_idstring

Installation package ID.

▶ package_typestring

A string representing the type of installation package. Select *one* of the following valid keywords and values:

- `upgrade` Package type should match the distribution type or platform:
- `sh`: x86_64 Linux SH installer
- `rpm`: x86_64 Linux RPM installer
- `deb`: x86_64 Linux DEB installer
- `aarch64_sh`: aarch64 Linux SH installer
- `aarch64_rpm`: aarch64 Linux RPM installer
- `aarch64_deb`: aarch64 Linux DEB installer
- `pkg`: Mac
- `x86`: Windows
- `x64`: Windows

REQUEST example-1

```
{ "request_data": { "distribution_id": "<distribution ID>", "package_type": "x86" } }
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ distribution_urlstring

URL for downloading the installation package.

RESPONSE example-1
```
{ "reply": { "distribution_url": "<DOWNLOAD_URL>" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.9 | Set an Endpoint Alias

post /public_api/v1/endpoints/update_agent_name

Set or modify an Alias field for your endpoints.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
```

```
'https://api-yourfqdn/public_api/v1/endpoints/update_agent_name'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"alias\":\"string\"}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/endpoints/update_agent_name", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/update_agent_name") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"alias\":\"string\"}}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "alias": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/endpoints/update_agent_name"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/update_agent_name")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"alias\":\"string\"}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "alias": "string" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/endpoints/update_agent_name")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/update_agent_name", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"alias\":\"string\"}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/update_agent_name"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"alias\":\"string\"}}"); CURLcode ret =
curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/update_agent_name"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"alias\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
▶ filtersarray

An array of filter fields.

▶ aliasstring

The alias name you want to set or modify.

Note: If you send an empty field, the current alias name is deleted.

REQUEST example-1

```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<distribution_id>" ] }
], "alias": "<alias_name>" } }
```
Responses

Successful response

Body
application/json

true=The alias name was set or modified successfully.

▼ boolean

true=The alias name was set or modified successfully.

RESPONSE
```
false
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.10 | Assign Tags

post /public_api/v1/tags/agents/assign

Assign one or more tags to one or more endpoints.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/tags/agents/assign'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/tags/agents/assign", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/tags/agents/assign")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}" response =
http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id", "operator": "in", "value": [
"string" ] } ], "tag": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/tags/agents/assign");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/tags/agents/assign")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"tag\":\"string\"}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id",
"operator": "in", "value": ["string"] ] ], "tag": "string" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/tags/agents/assign")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/tags/agents/assign", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"tag\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type:
application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/tags/agents/assign"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}"); CURLcode ret =
curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/tags/agents/assign"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"tag\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

▶ filtersarray

An array of filter fields.

▶ tagstring

The tag you want to assign.

REQUEST example-1

```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<distribution_id>" ] }
], "tag": "<tag_name>" } }
```

Responses

Successful response

Body
application/json

true=The tag name was assigned successfully.

▼ boolean

true=The tag name was assigned successfully.

RESPONSE

```
false
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

x Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.11 | Remove Tags

post /public_api/v1/tags/agents/remove

Remove one or more tags from one or more endpoints.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/tags/agents/remove'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/tags/agents/remove", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/tags/agents/remove")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "tag": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/tags/agents/remove");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/tags/agents/remove")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "tag": "string" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/tags/agents/remove")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/tags/agents/remove", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"tag\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type:
application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/tags/agents/remove"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"tag\":\"string\"}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/tags/agents/remove"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"tag\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

A dictionary containing the following API request fields.

▶ filtersarray

Array of filter fields.

▶ tagstring

The tag you want to remove.

REQUEST example-1

```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<distribution_id>" ] }
], "tag": "<tag_name>" } }
```

Responses

Successful response

Body
application/json

true=tag name removed successfully.

▼ boolean

true=tag name removed successfully.

RESPONSE

```
false
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.
▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.12 | Get Endpoint

post /public_api/v1/endpoints/get_endpoint

Gets a list of filtered endpoints.

- The response is concatenated using AND condition (OR is not supported).
- The maximum result set size is 100.
- Offset is the zero-based number of endpoints from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}
Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/get_endpoint'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id\",\"keyword\":\"ASC\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/get_endpoint", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/get_endpoint") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id\",\"keyword\":\"ASC\"}}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
"string" } ], "search_from": 0, "search_to": 0, "sort": { "field": "endpoint_id", "keyword": "ASC" } } }); const xhr =
new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/endpoints/get_endpoint"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/get_endpoint")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id\",\"keyword\":\"ASC\"}}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": "string" ] ], "search_from": 0, "search_to": 0, "sort": [ "field": "endpoint_id",
"keyword": "ASC" ] ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: [])
let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/endpoints/get_endpoint")! as
URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST"
request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask
= session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error !=
nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } })
dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/get_endpoint", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id\",\"keyword\":\"ASC\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/get_endpoint"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id\",\"keyword\":\"ASC\"}}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/get_endpoint"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":0,\"sort\":
{\"field\":\"endpoint_id\",\"keyword\":\"ASC\"}}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject

A dictionary containing the API request fields.

An empty dictionary returns all results.

▶ filtersarray

Array of filter fields.

▶ search_frominteger

Represents the start offset within the query result set from which you want endpoints returned.

Endpoints are returned as a zero-based list. Any endpoint indexed less than this value is not returned in the final result set and defaults to zero.

▶ search_tointeger

Represents the end offset within the result set after which you do not want endpoints returned.

Endpoint in the endpoint list that is indexed higher than this value is not returned in the final results set. Defaults to 100, which returns all endpoints to the end of the list.

▶ sortobjectrequired

Identifies the sort order for the result set.

REQUEST Request filtered results

```
{ "request_data": { "search_from": 0, "search_to": 1, "sort": { "field": "endpoint_id", "keyword": "asc" }, "filters":
[ { "field": "group_name", "operator": "in", "value": [ "Test-Group-01" ] }, { "field": "endpoint_status", "operator":
"in", "value": [ "disconnected" ] }, { "field": "dist_name", "operator": "in", "value": [ "papi-test" ] }, { "field":
"scan_status", "operator": "in", "value": [ "none", "pending", "in_progress", "pending_cancellation", "aborted",
"success", "canceled", "error" ] } ] } }
{ "request_data": {} }
```

Responses

OK

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ total_countinteger

Number of total results of this filter without paging.

▶ result_countinteger

Number of endpoints actually returned as result.

▶ endpointsarray

A list of endpoints.

RESPONSE Example 1

```
{ "reply": { "total_count": 1, "result_count": 1, "endpoints": [ { "endpoint_id": "<endpoint ID>", "endpoint_name": "
<endpoint name>", "endpointTags": "<tag name>", "endpoint_type": "<endpoint type>", "endpoint_status": "CONNECTED",
"operational_status_details": [ { "title": "XDR Data Collection not running or not sent", "reason": "Linux kernel
version is not supported" }, { "title": "BTP not working", "reason": "Linux kernel version is not supported" }, {
"title": "Antimalware flow is asynchronous", "reason": "Linux kernel version is not supported" }, { "title": "Local
privilege escalation", "reason": "Linux kernel version is not supported" } ], "os_type": "AGENT_OS_WINDOWS",
"os_version": "8.0.xxx", "ip": [ "<IP address>" ], "ipv6": [], "public_ip": "<IP address>", "users": [ "XDR" ],
"domain": "WORKGROUP", "alias": "", "first_seen": 1606218761377, "last_seen": 1606218769163, "content_version": "",
"installation_package": "XDR", "active_directory": null, "install_date": 1606218762089, "endpoint_version": "
<version>", "is_isolated": "AGENT_UNISOLATED", "isolated_date": null, "group_name": [], "operational_status":
"PARTIALLY_PROTECTED", "scan_status": "SCAN_STATUS_NONE", "content_release_timestamp": 1636285746000,
```

"last_content_update_time": 1636381954285, "content_status": "up_to_date", "operating_system": "Debian 10.11", "mac_address": ["42:00:00:00:00:00"], "assigned_prevention_policy": "Linux Default", "assigned_extensions_policy": "" }] } }

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.4.13 | Initiate Forensics Triage

post /public_api/v1/triage_endpoint

Initiate forensics triage for the specified agents.

- Maximum of 10 concurrent triage actions at a time.
- Specified agents must have Forensics License enabled.
- Specified agents must be the same OS, Windows or macOS, but not a mixture of both.
- Specified configuration must have type "Online = True".

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
```

```
'https://api-yourfqdn/public_api/v1/triage_endpoint'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"agent_ids\":
[\"string\"],\"collector_uuid\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/triage_endpoint",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/triage_endpoint")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"agent_ids\":
[\"string\"],\"collector_uuid\":\"string\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "agent_ids": [ "string" ], "collector_uuid": "string" } }); const xhr
= new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/triage_endpoint"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/triage_endpoint")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"agent_ids\":[\"string\"],\"collector_uuid\":\"string\"}}")
.asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "agent_ids": ["string"], "collector_uuid":
"string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/triage_endpoint")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields =
headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with:
request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else
{ let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/triage_endpoint", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"agent_ids\":[\"string\"],\"collector_uuid\":\"string\"}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/triage_endpoint"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"agent_ids\":
[\"string\"],\"collector_uuid\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/triage_endpoint"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"agent_ids\":[\"string\"],\"collector_uuid\":\"string\"}}", ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ agent_idsarray[string]

List of agents to run forensics triage on.

▶ collector_uuidstring

UUID of the triage configuration. If none is specified, the default configuration is used for this action.

REQUEST
```
{ "request_data": { "agent_ids": [ "example" ], "collector_uuid": "example" } }
```
Responses

OK

Body
application/json
▼ replyobject
▶ group_action_idinteger

Unique ID for triage action.
▶ successful_agent_idsarray[string]

List of agent IDs that successfully received the triage action.

▶ unsuccessful_agent_idsarray

List of agent IDs that did not successfully receive the triage action.

RESPONSE Example 1

```
{ "reply": { "group_action_id": 325, "successful_agent_ids": [ "5111c5eb93944e2f97674db4f36b4211" ],
"unsuccessful_agent_ids": [] } }
```

Bad Request

Unauthorized

Payment Required

Forbidden

Internal Server Error

2.1.5 | Response Action

APIs for response actions

2.1.5.1 | Restore File

post /public_api/v1/endpoints/restore

Restore a quarantined file on a requested endpoints. When filtering by multiple fields:

- Response is concatenated using AND condition (OR is not supported).
- Maximum result set size is 100.
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/restore'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"file_hash\":\"string\",\"endpoint_id\":\"string\",\"incident_id\":0}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/restore", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/endpoints/restore")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":
{\"file_hash\":\"string\",\"endpoint_id\":\"string\",\"incident_id\":0}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "file_hash": "string", "endpoint_id": "string", "incident_id": 0 } });
const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () {
if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/endpoints/restore"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
```

```
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/restore")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"file_hash\":\"string\",\"endpoint_id\":\"string\",\"incident_id\":0}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "file_hash": "string", "endpoint_id":
"string", "incident_id": 0 ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters,
options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/endpoints/restore")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0)
request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session =
URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response,
error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/restore", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"file_hash\":\"string\",\"endpoint_id\":\"string\",\"incident_id\":0}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/restore"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"file_hash\":\"string\",\"endpoint_id\":\"string\",\"incident_id\":0}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/restore"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"file_hash\":\"string\",\"endpoint_id\":\"string\",\"incident_id\":0}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ file_hashstring

String that represents the file in hash. Hash must be a valid SHA256.

▶ endpoint_idstring

String that represents the endpoint ID. Note: if it is not specified, the request will run restore on all endpoints which relate to the quarantined file you defined.

▶ incident_idinteger

String representing the case ID. When included in the request, the Restore File action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST example-1

```
{ "request_data": { "file_hash": "<hash value>", "incident_id": 302 } }
```
Responses

Successful response

Body
application/json
▼ replyobject
▶ action_idstring
▶ endpoints_countstring
RESPONSE example-1

```
{ "reply": { "action_id": "<action ID>", "status": 1, "endpoints_count": "673" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.2 | File Retrieval Details

post /public_api/v1/actions/file_retrieval_details

View the API required to call in order to download the file retrieved by the **Retrieve File** API request according to the action ID.

The response contains a file hash you need to download and then unzip to view:

1. Download the file.

```
curl -XPOST "https://api-{fqdn}/public_api/v1/download/<api_value>"
-H "x-xdr-auth-id:{API_KEY_ID}"
-H "Authorization:{API_KEY}"
-H 'Content-Type:application/json'
--output /tmp/file.zip
```

2. Unzip the file: `unzip /tmp/file.zip`

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/actions/file_retrieval_details'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"group_action_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/actions/file_retrieval_details", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/actions/file_retrieval_details") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"group_action_id\":\"string\"}}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "group_action_id": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/actions/file_retrieval_details"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/actions/file_retrieval_details") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{\"group_action_id\":\"string\"}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": ["group_action_id": "string"]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/actions/file_retrieval_details")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/actions/file_retrieval_details", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"group_action_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/actions/file_retrieval_details"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"group_action_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/actions/file_retrieval_details"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"group_action_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ group_action_idstring

The action ID of the Retrieve File API response.

REQUEST example-1

```
{ "request_data": { "group_action_id": "<action ID>" } }
```

Responses

Successful response

Body
application/json
▼ replyobject
▶ dataobject
RESPONSE Example 1

```
{ "reply": { "data": { "<endpoint_ID>": "https://api-{fqdn}/public_api/v1/download/<api_value>" } } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.3 | Allow List Files

post /public_api/v1/hash_exceptions/allowlist

Add files which do not exist in the allow or block lists to an allow list.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/hash_exceptions/allowlist'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"hash_list\":
[\"string\"],\"comment\":\"string\",\"incident_id\":0}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-
auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/hash_exceptions/allowlist", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/hash_exceptions/allowlist") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}" response =
http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "hash_list": [ "string" ], "comment": "string", "incident_id": 0 } });
const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () {
if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/hash_exceptions/allowlist"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/hash_exceptions/allowlist")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}")
.asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "hash_list": ["string"], "comment": "string",
"incident_id": 0 ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: [])
let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/hash_exceptions/allowlist")!
as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST"
request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask
= session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error !=
nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } })
dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/hash_exceptions/allowlist", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"hash_list\":
[\"string\"],\"comment\":\"string\",\"incident_id\":0}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/hash_exceptions/allowlist"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"hash_list\":
[\"string\"],\"comment\":\"string\",\"incident_id\":0}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/hash_exceptions/allowlist"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ hash_listarray[string]

A list of hashed files you want to add to the allow list. Hash must be a valid SH256.

▶ commentstring

Additional information regarding the action.

▶ incident_idinteger

The case ID related to the hash. When included in the request, the Allow List action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST example-1

```
{ "request_data": { "hash_list": [ "032196FB1A---DFCF69E5D553F0", "365296EB1B---FCF29E5D553E4", "365296EB1B---
FCF69E3D553E4", "365296EB1B---FCF69E5D553D4", "365296EB1B---FCF79E5D553D4" ], "comment": "test", "incident_id": 5 } }
```

Responses

Successful response

Body
application/json

true=File successfully added to the allow list.

▼ boolean

true=File successfully added to the allow list.

RESPONSE
```
false
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.4 | Get Quarantine Status

post /public_api/v1/quarantine/status

Retrieve the quarantine status for specified files.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/quarantine/status'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"files\":
[{\"endpoint_id\":\"string\",\"file_path\":\"string\",\"file_hash\":\"string\"}]}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/quarantine/status", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/quarantine/status")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"files\":
[{\"endpoint_id\":\"string\",\"file_path\":\"string\",\"file_hash\":\"string\"}]}}" response = http.request(request)
puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "files": [ { "endpoint_id": "string", "file_path": "string",
"file_hash": "string" } ] } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/quarantine/status");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/quarantine/status")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"files\":
[{\"endpoint_id\":\"string\",\"file_path\":\"string\",\"file_hash\":\"string\"}]}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["files": [ [ "endpoint_id": "string",
"file_path": "string", "file_hash": "string" ] ]]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/quarantine/status")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/quarantine/status", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"files\":
[{\"endpoint_id\":\"string\",\"file_path\":\"string\",\"file_hash\":\"string\"}]}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/quarantine/status"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"files\":
[{\"endpoint_id\":\"string\",\"file_path\":\"string\",\"file_hash\":\"string\"}]}}"); CURLcode ret =
curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/quarantine/status"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"files\":[{\"endpoint_id\":\"string\",\"file_path\":\"string\",\"file_hash\":\"string\"}]}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ filesarray

Array of endpoint IDs, filepaths, and file hash.

REQUEST example-1

```
{ "request_data": { "files": [ { "endpoint_id": "<endpoint ID>", "file_path": "C:\\<file path>\\test_x64.msi",
"file_hash": "<hash value>" } ] } }
```

Responses

Successful response

Body
application/json
▼ replyarray
[
▶ endpoint_idstring

Endpoint ID.

▶ file_pathstring

File path.
▶ file_hashstring

File hash.

▶ statusboolean

The file's status. True: The file is quarantined. False: The file is not quarantined.

]
RESPONSE example-1
```
{ "reply": [ { "endpoint_id": "<endpoint ID>", "file_path": "C:\\<file path>\\test_x64.msi", "file_hash": "<hash value>", "status": false } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.5 | Quarantine Files

post /public_api/v1/endpoints/quarantine

Quarantine file on selected endpoints. You can select up to 1000 endpoints.

Note: A success response means that the request reached the defined endpoints, however if the file was not found there, no quarantine action will take place. To ensure if the file has been quarantined, check the Cortex XDR Action Center.

When filtering by multiple fields:

- Response is concatenated using AND condition (OR is not supported).
- Maximum result set size is 1000.
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/quarantine'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"file_path\":\"string\",\"file_hash\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE",
'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/quarantine", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/quarantine") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"file_path\":\"string\",\"file_hash\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "file_path": "string", "file_hash": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/quarantine");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/quarantine")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"file_path\":\"string\",\"file_hash\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "file_path": "string", "file_hash": "string" ]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/endpoints/quarantine")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/quarantine", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"file_path\":\"string\",\"file_hash\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/quarantine"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"file_path\":\"string\",\"file_hash\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/quarantine"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
```

```
[\"string\"]}],\"file_path\":\"string\",\"file_hash\":\"string\"}}", ParameterType.RequestBody); IRestResponse
response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersarray

An array of filter fields.

▶ file_pathstring

The path of the file you want to quarantine. You must enter a proper path and not symbolic links.

▶ file_hashstring

Case ID. When included in the request, the **Quarantine File** action will appear in the **Cortex Cloud Case View Timeline** tab.

REQUEST example-1
```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID>" ] } ],
"file_path": "C:\\<file path>\\test_x64.msi", "file_hash": "<hash value>" } }
```
Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ action_idstring

ID of action to quarantine selected endpoints. Response only indicates the request was successfully sent to the endpoint. To track if the file quarantine succeeded either:

- In Cortex XDR console, navigate to **Response** > **Action Center** and search for the action ID. Make sure the **Action ID** field is selected in the table **Layout** settings.
- Send a **Get Action Status** API request.

▶ statusstring

Integer representing whether the action:

- 1: succeeded
- 0: failed

▶ endpoints_countstring

Number of endpoints included in the request.

RESPONSE example-1
```
{ "reply": { "action_id": "[ID value]", "status": "1", "endpoints_count": "673" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

2.1.5.6 | Block List Files

post /public_api/v1/hash_exceptions/blocklist

Add files which do not exist in the allow or block lists to a block list. You can view the block list in the UI at **Investigation & Response** > Response > **Action Center** > **Block List**.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/hash_exceptions/blocklist", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "hash_list": [ "string" ], "comment": "string", "incident_id": 0 } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": [ "hash_list": ["string"], "comment": "string", "incident_id": 0 ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: [])
```

```
let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/hash_exceptions/blocklist"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"hash_list\":[\"string\"],\"comment\":\"string\",\"incident_id\":0}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ hash_listarray[string]

A list of hashed files you want add to a block list. Hash must be a valid SH256.

▶ commentstring

Additional information regarding the action.

▶ incident_idinteger

The case ID related to the hash. When included in the request, the Block List action appears in the Cortex Cloud Case View Timeline tab.

REQUEST example-1

```
{ "request_data": { "hash_list": [ "032196FB1A---DFCF69E5D553F0", "365296EB1B---FCF69E7D553E4", "365296EB1B---FCF69E5D523E4", "365296EB1B---FCF69E5D553D4", "365296EB1B---FCF63E5D553D4" ], "comment": "test", "incident_id": 5 } }
```

Responses

Successful response

Body
application/json

true=File successfully added to block list.

▼ boolean

true=File successfully added to block list.

RESPONSE

```
false
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:"`{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}`"
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:"`{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}`"
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:"`{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}`"
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:"`{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}`"
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.7 | Unisolate Endpoints

post /public_api/v1/endpoints/unisolate

Reverse the isolation of one or more endpoints in single request.

Note: You can only send a request with either `endpoint_id` to unisolate one endpoint or with filters to unisolate more than one endpoint. An error is raised if you try to use both `endpoint_id` and the filters.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/unisolate'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/unisolate", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/endpoints/unisolate")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "endpoint_id": "string", "incident_id": "string" } }); const xhr = new XMLHttpRequest();
```

```
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/unisolate");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/unisolate")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "endpoint_id": "string", "incident_id": "string" ]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/endpoints/unisolate")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/unisolate", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/unisolate"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/unisolate"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse
response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersarray

An array of filter fields for unisolating a number of endpoints at once. Note: This field is only required if unisolating more than one endpoint.

▶ endpoint_idstring

The ID of the endpoint to unisolate.

Note: this field is only required if unisolating one endpoint.

▶ incident_idstring

Case ID. When included in the request, the Unisolate Endpoints action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST Unisolate one endpoint
```
{ "request_data": { "endpoint_id": "<endpoint ID>" } }
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "IN", "value": [ "<endpoint_id_1>", "
<endpoint_id_2>", "<endpoint_id_3>" ] } ] } }
```
Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.
▶ action_idstring

ID of the action to unisolate selected endpoints. Response only indicates the request was successfully sent to the endpoint. To track if the endpoint was restored either:

- In the Cortex XDR console, navigate to **Response** > **Action Center** > **Isolation** and search for the action ID. Make sure the **Action ID** field is selected in the table **Layout** settings.
- Send a **Get Action Status** API request.

▶ endpoints_countstring

Number of endpoints included in the request.

RESPONSE example-1

```
{ "reply": { "action_id": "<action ID>", "status": "1", "endpoints_count": "673" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.8 | Cancel Scan Endpoints

post /public_api/v1/endpoints/abort_scan

Cancel the scan of selected endpoints. A scan can only be aborted if the selected endpoints are in Pending or in Progress status.

When filtering by multiple fields:

- Response is concatenated using AND condition (OR is not supported).
- Offset is the zero-based number of endpoints from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/abort_scan'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"incident_id\":\"string\"}}" headers =
{ 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/endpoints/abort_scan", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/abort_scan") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"incident_id\":\"string\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "incident_id": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/abort_scan");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/abort_scan")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"incident_id\":\"string\"}}")
.asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "incident_id": "string" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/endpoints/abort_scan")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/abort_scan", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"incident_id\":\"string\"}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/abort_scan"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"incident_id\":\"string\"}}"); CURLcode
ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/abort_scan"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersobject

An array of filter fields to filter which endpoints to cancel scanning. To cancel scan of all endpoints, use the value "all".

▶ incident_idstring

Case ID. When included in the request, the Cancel Scan Endpoints action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST To cancel scan of all endpoints

```
{ "request_data": { "filters": "all" } }
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID>" ] }, {
"field": "dist_name", "operator": "in", "value": [ "WinInstaller" ] }, { "field": "group_name", "operator": "in",
"value": [ "test" ] }, { "field": "scan_status", "operator": "in", "value": [ "none", "pending", "in_progress",
"pending_cancellation", "aborted", "success" ] }, { "field": "group_name", "operator": "in", "value": [ "test" ] } ] }
}
```

Responses

Successful response

Body
application/json

▼ replyobject

JSON object containing the query result.

▶ action_idstring

ID of action to cancel scan selected endpoints. Response only indicates the request was successfully sent to the endpoint. To track if the scan succeeded either:

- In Cortex XDR console, navigate to Response > Action Center and search for the action ID. Make sure the Action ID field is selected in the table Layout settings.
- Send a Get Action Status API request.

▶ endpoints_countstring

Number of endpoints included in the request.

RESPONSE

```
{ "reply": { "action_id": "example", "endpoints_count": "example" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.9 | Scan Endpoints

post /public_api/v1/endpoints/scan

Run a scan on selected endpoints.

- Response is concatenated using AND condition (OR is not supported).
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/scan'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"filters\":\"all\",\"incident_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/endpoints/scan",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/endpoints/scan") http
= Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":
{\"filters\":\"all\",\"incident_id\":\"string\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": "all", "incident_id": "string" } }); const xhr = new
XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/endpoints/scan"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/scan")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":\"all\",\"incident_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": "all", "incident_id": "string" ]]
as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/endpoints/scan")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/scan", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":\"all\",\"incident_id\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/scan"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":\"all\",\"incident_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/scan"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"filters\":\"all\",\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersstring (Enum)

An array of filter fields. To scan all endpoints, use the value `all`.

▶ incident_idstring

Case ID. When included in the request, the Scan Endpoints action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST Scan all endpoints

```
{ "request_data": { "filters": "all" } }
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID>" ] }, { "field": "dist_name", "operator": "in", "value": [ "WinInstaller" ] }, { "field": "group_name", "operator": "in", "value": [ "test\"" ] }, { "field": "scan_status", "operator": "in", "value": [ "none", "pending", "in_progress", "pending_cancellation", "aborted", "success" ] }, { "field": "group_name", "operator": "in", "value": [ "test" ] } ] } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ action_idstring

ID of action to scan selected endpoints. Response only indicates the request was successfully sent to the endpoint. To track if the scan was successful either:

- In Cortex XDR console, navigate to Response > Action Center > All Actions and search for the action ID. Make sure the Action ID field is selected in the table Layout settings.
- Send a Get Action Status API request.

▶ endpoints_countstring

Number of endpoints included in the request.

RESPONSE

```
{ "reply": { "action_id": "example", "endpoints_count": "example" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.5.10 | Get Action Status

post /public_api/v1/actions/get_action_status

Retrieve the status of the requested actions according to the action ID.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/actions/get_action_status'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"group_action_id\":0}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/actions/get_action_status", payload,
headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/actions/get_action_status") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"group_action_id\":0}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "group_action_id": 0 } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/actions/get_action_status"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/actions/get_action_status")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"group_action_id\":0}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["group_action_id": 0]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/actions/get_action_status")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/actions/get_action_status", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"group_action_id\":0}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/actions/get_action_status"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"group_action_id\":0}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/actions/get_action_status"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"group_action_id\":0}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ group_action_idinteger

Action ID of the selected request.

REQUEST example-1

```
{ "request_data": { "group_action_id": 123456789 } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ dataobject
RESPONSE example-1

```
{ "reply": { "data": { "<agent ID>": "COMPLETED_SUCCESSFULLY" } } }
```

Bad Request. Got an invalid JSON.

Body
application/json
▼ replyobject
▶ dataobject

JSON object containing the query result.

▶ errorReasonsobject

Returns all error messages the agent returns to allow for easier analysis.

RESPONSE Example 1

```
{ "reply": { "data": { "5a763600a3e44928a94ba84b6088380f": "FAILED" }, "errorReasons": { "5a763600a3e44928a94ba84b6088380f": { "errorData": "{\"reportIds\": [\"0ca3c1ace7694fcd8b44b735e5ba6c04\"],\"errorText\":\"\"}", "terminated_by": "instance_id", "errorDescription": "Element not found.\r\n", "terminate_result": [ { "path": null, "status": 1 } ] } } } }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json
▼ replyobject
▶ dataobject

JSON object containing the query result.

▶ errorReasonsobject

Returns all error messages the agent returns to allow for easier analysis.

RESPONSE Example 1

```
{ "reply": { "data": { "5a763600a3e44928a94ba84b6088380f": "FAILED" }, "errorReasons": {
"5a763600a3e44928a94ba84b6088380f": { "errorData": "{\"reportIds\":
[\"0ca3c1ace7694fcd8b44b735e5ba6c04\"],\"errorText\":\"\"}", "terminated_by": "instance_id", "errorDescription":
"Element not found.\r\n", "terminate_result": [ { "path": null, "status": 1 } ] } } } }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json
▼ replyobject
▶ dataobject

JSON object containing the query result.

▶ errorReasonsobject

Returns all error messages the agent returns to allow for easier analysis.

RESPONSE Example 1

```
{ "reply": { "data": { "5a763600a3e44928a94ba84b6088380f": "FAILED" }, "errorReasons": {
"5a763600a3e44928a94ba84b6088380f": { "errorData": "{\"reportIds\":
[\"0ca3c1ace7694fcd8b44b735e5ba6c04\"],\"errorText\":\"\"}", "terminated_by": "instance_id", "errorDescription":
"Element not found.\r\n", "terminate_result": [ { "path": null, "status": 1 } ] } } } }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json
▼ replyobject
▶ dataobject

JSON object containing the query result.

▶ errorReasonsobject

Returns all error messages the agent returns to allow for easier analysis.

RESPONSE Example 1

```
{ "reply": { "data": { "5a763600a3e44928a94ba84b6088380f": "FAILED" }, "errorReasons": {
"5a763600a3e44928a94ba84b6088380f": { "errorData": "{\"reportIds\":
[\"0ca3c1ace7694fcd8b44b735e5ba6c04\"],\"errorText\":\"\"}", "terminated_by": "instance_id", "errorDescription":
"Element not found.\r\n", "terminate_result": [ { "path": null, "status": 1 } ] } } } }
```

Internal server error. A unified status for API communication type errors.

Body
application/json
▼ replyobject
▶ dataobject

JSON object containing the query result.

▶ errorReasonsobject

Returns all error messages the agent returns to allow for easier analysis.

RESPONSE Example 1

```
{ "reply": { "data": { "5a763600a3e44928a94ba84b6088380f": "FAILED" }, "errorReasons": {
"5a763600a3e44928a94ba84b6088380f": { "errorData": "{\"reportIds\":
[\"0ca3c1ace7694fcd8b44b735e5ba6c04\"],\"errorText\":\"\"}", "terminated_by": "instance_id", "errorDescription":
"Element not found.\r\n", "terminate_result": [ { "path": null, "status": 1 } ] } } } }
```

2.1.5.11 | Retrieve File

post /public_api/v1/endpoints/file_retrieval

Retrieve files from selected endpoints. You can retrieve up to 20 files, from no more than 10 endpoints.

- Response is concatenated using AND condition (OR is not supported).
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.
Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/file_retrieval'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"files\":{\"windows\":
[\"string\"],\"linux\":[\"string\"],\"macos\":[\"string\"]},\"incident_id\":\"string\"}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/file_retrieval", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/endpoints/file_retrieval") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"files\":{\"windows\":[\"string\"],\"linux\":[\"string\"],\"macos\":
[\"string\"]},\"incident_id\":\"string\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "files": { "windows": [ "string" ], "linux": [ "string" ], "macos": [ "string" ] }, "incident_id":
"string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange",
function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST",
"https://api-yourfqdn/public_api/v1/endpoints/file_retrieval"); xhr.setRequestHeader("Authorization",
"SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type",
"application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/file_retrieval")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"files\":{\"windows\":
[\"string\"],\"linux\":[\"string\"],\"macos\":[\"string\"]},\"incident_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "files": [ "windows": ["string"], "linux": ["string"], "macos": ["string"]
], "incident_id": "string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters,
options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/endpoints/file_retrieval")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/file_retrieval", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"files\":{\"windows\":
[\"string\"],\"linux\":[\"string\"],\"macos\":[\"string\"]},\"incident_id\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/file_retrieval"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"files\":{\"windows\":[\"string\"],\"linux\":[\"string\"],\"macos\":[\"string\"]},\"incident_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/file_retrieval"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"files\":{\"windows\":[\"string\"],\"linux\":[\"string\"],\"macos\":[\"string\"]},\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersarray

An array of filter fields.

▶ filesobject

One of the operating system types must be included.

▶ incident_idstring

Case ID. When included in the request, the Retrieve File action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST
```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "example" ] } ], "files": { "windows": [ "example" ], "linux": [ "example" ], "macos": [ "example" ] }, "incident_id": "example" } }
```
Responses

OK

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ action_idstring

ID of action to retrieve files from selected endpoints. Response only indicates the request was successfully sent to the endpoint. To track if the file was retrieved successfully either: in the Cortex XDR console, navigate to Response > Action Center > Isolation and search for the action ID. Make sure the Action ID field is selected in the table Layout settings by selecting the three vertical dots. To view the file, send a File Retrieval Details request.

▶ statusstring
▶ endpoints_countstring

Number of endpoints included in the request.

RESPONSE
```
{ "reply": { "action_id": "example", "status": "example", "endpoints_count": "example" } }
```

Bad Request

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Unauthorized access. User does not have the required license type to run this API.

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Internal server error. A unified status for API communication type errors.

2.1.5.12 | Isolate Endpoints

post /public_api/v1/endpoints/isolate

Isolate one or more endpoints in a single request. Request is limited to 1000 endpoints.

Required license: Cortex Cloud Runtime Security. In Cortex Cloud Posture Security, you need the Cortex Cloud Runtime Security add-on.
Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/endpoints/isolate'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/endpoints/isolate", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/endpoints/isolate")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "endpoint_id": "string", "incident_id": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/endpoints/isolate");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/endpoints/isolate")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "endpoint_id": "string", "incident_id": "string" ]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/endpoints/isolate")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/endpoints/isolate", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/endpoints/isolate"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/endpoints/isolate"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"endpoint_id\":\"string\",\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse
response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersarray

Array of filtered fields for isolating a number of endpoints at once. Note: Only required if isolating more than one endpoint.

▶ endpoint_idstring

Identifies the endpoint to isolate. Note: Only required if isolating one endpoint.

▶ incident_idstring

The case ID. When included in the request, the Isolate Endpoints action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST Isolate one endpoint

```
{ "request_data": { "endpoint_id": "<endpoint ID>" } }
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID 1>", "
<endpoint ID 2>", "<endpoint ID 3>" ] } ] } }
```

Responses

OK

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ action_idstring

Action ID to scan selected endpoints. The response only indicates the request was successfully sent to the endpoint. To track if the isolation succeeded either:

- In the Cortex XDR console, navigate to Response > Action Center > Isolation and search for the action ID. Make sure the Action ID field is selected in the table Layout settings by selecting the vertical ellipses.
- Send a Get Action Status request.

▶ endpoints_countstring

Number of endpoints included in the request.

RESPONSE Example 1

```
{ "reply": { "action_id": "<action ID>", "status": "1", "endpoints_count": "673" } }
```

Bad Request. Got an invalid JSON.

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, ID, or other invalid authentication parameters.

Unauthorized access. User does not have the required license type to run this API.

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Internal server error. A unified status for API communication type errors.

2.1.5.13 | Get triage presets

post /public_api/v1/get_triage_presets

Get all triage preset information including triage name, platform, description, created by, and triage type.

Required license: In Cortex Cloud Runtime Security, requires the Forensics add-on. Not supported in Cortex Cloud Posture Management.

Request headers
▼ Authorization String required

{api_key}
Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/get_triage_presets'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/get_triage_presets", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/get_triage_presets")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{}}" response = http.request(request)
puts response.read_body
const data = JSON.stringify({ "request_data": {} }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/get_triage_presets");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/get_triage_presets")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": []] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/get_triage_presets")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/get_triage_presets", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-
type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/get_triage_presets"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/get_triage_presets"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
REQUEST

```
{ "request_data": {} }
```

Responses

OK

Body
application/json
▼ replyobject
▶ triage_presetsarray

RESPONSE Example 1

{ "reply": { "triage_presets": [{ "uuid": "374ecf0457c944c39791e5a60d1a6d24", "name": "XDR Default", "os": "MACOS", "description": "Default macOS Triage configuration included with the XDR Forensics add-on", "created_by": "user@company.com", "type": "Online / Offline" }, { "uuid": "ea7a5d3ff52d41629e96fbd1d5f68535", "name": "XDR Default", "os": "WINDOWS", "description": "Default Triage configuration included with XDR Forensics add-on", "created_by": "user@company.com", "type": "Online / Offline" }] } }

2.1.6 | Script execution

APIs executing script

2.1.6.1 | Run Snippet Code Script

post /public_api/v1/scripts/run_snippet_code_script

Initiate a new endpoint script execution action using provided snippet code. Cortex XDR supports sending your request in Base64.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/run_snippet_code_script'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"timeout\":600,\"snippet_code\":\"string\",\"incident_id\":\"string\"}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/scripts/run_snippet_code_script", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/scripts/run_snippet_code_script") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"timeout\":600,\"snippet_code\":\"string\",\"incident_id\":\"string\"}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "timeout": 600, "snippet_code": "string", "incident_id": "string" } }); const xhr = new
XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/scripts/run_snippet_code_script"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/run_snippet_code_script")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"timeout\":600,\"snippet_code\":\"string\",\"incident_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "timeout": 600, "snippet_code": "string", "incident_id": "string" ]] as
[String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/scripts/run_snippet_code_script")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/scripts/run_snippet_code_script", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"timeout\":600,\"snippet_code\":\"string\",\"incident_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/scripts/run_snippet_code_script"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"timeout\":600,\"snippet_code\":\"string\",\"incident_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/run_snippet_code_script"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":[\"string\"]}],\"timeout\":600,\"snippet_code\":\"string\",\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersarray

An array of filter fields for running the script on a number of endpoints at once.

▶ timeoutinteger

The timeout in seconds for this execution. Default value is 600.

▶ snippet_codestring

Section of a script you want to initiate on an endpoint.

▶ incident_idstring

Case ID. When included in the request, the Run Snippet Code Script action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST example-1

```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID>" ] } ], "snippet_code": "print (\"7\")" } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ action_idstring

ID of the action initiated. ID will be used as a reference to track in the action center.

▶ endpoints_countinteger

Number of endpoints the action was initiated on.

RESPONSE example-1

```
{ "reply": { "action_id": "<action ID>", "endpoints_count": 21 } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.2 | Run Script

post /public_api/v1/scripts/run_script

Initiate a new endpoint script execution action using a script from the script library. The script can be run on up to 1000 endpoints.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/run_script'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"script_uid\":\"string\",\"parameters_values\":
{\"x\":\"string\",\"y\":0},\"timeout\":600,\"incident_id\":\"string\"}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/scripts/run_script", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/scripts/run_script")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"script_uid\":\"string\",\"parameters_values\":
{\"x\":\"string\",\"y\":0},\"timeout\":600,\"incident_id\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value":
[ "string" ] } ], "script_uid": "string", "parameters_values": { "x": "string", "y": 0 }, "timeout": 600,
"incident_id": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/scripts/run_script");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/run_script")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"script_uid\":\"string\",\"parameters_values\":
{\"x\":\"string\",\"y\":0},\"timeout\":600,\"incident_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id_list",
"operator": "in", "value": ["string"] ] ], "script_uid": "string", "parameters_values": [ "x": "string", "y": 0 ],
"timeout": 600, "incident_id": "string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject:
parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/scripts/run_script")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0)
request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session =
URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response,
error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/scripts/run_script", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"script_uid\":\"string\",\"parameters_values\":
{\"x\":\"string\",\"y\":0},\"timeout\":600,\"incident_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/scripts/run_script"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"script_uid\":\"string\",\"parameters_values\":
{\"x\":\"string\",\"y\":0},\"timeout\":600,\"incident_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/run_script"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id_list\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"script_uid\":\"string\",\"parameters_values\":
{\"x\":\"string\",\"y\":0},\"timeout\":600,\"incident_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse
response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ filtersarray

Array of filter fields for running the script on a number of endpoints at once.

▶ script_uidstring

GUID, unique identifier of the script, returned by the Get Scripts API per script.

▶ parameters_valuesobjectrequired

Dictionary containing the parameter name, `key`, and its value for this execution, `value`.

You can obtain these values by running Get Script Metadata API.

▶ timeoutinteger

Timeout in seconds for this execution. Default value is 600.

▶ incident_idstring

Case ID. When included in the request, the Run Script action will appear in the Cortex Cloud Case View Timeline tab.

REQUEST example-1

```
{ "request_data": { "filters": [ { "field": "endpoint_id_list", "operator": "in", "value": [ "<endpoint ID>" ] } ],
"script_uid": "<unique ID>", "parameters_values": { "x": "param input as returned in Get Script Metadata", "y": 4 } }
}
```

Responses

Successful response

Body

application/json

▼ replyobject

JSON object containing the query result.

▶ action_idinteger

ID of the action initiated. ID will be used as a reference to track in the action center.

▶ endpoints_countinteger

Number of endpoints the action was initiated on.

▶ statusinteger

Integer representing whether the action:

- 1: succeeded
- 0: failed

RESPONSE example-1

```
{ "reply": { "action_id": 22519813685366, "status": 1, "endpoints_count": 1 } }
```

Bad Request. Got an invalid JSON.

Body

application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body

application/json

The query result upon error.

▼ err_codestring

HTTP response code.
▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.3 | Get Script Metadata

post /public_api/v1/scripts/get_script_metadata

Get the full definitions of a specific script in the scripts library.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/get_script_metadata'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"script_uid\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/scripts/get_script_metadata", payload,
headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/scripts/get_script_metadata") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"script_uid\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "script_uid": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/scripts/get_script_metadata"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/get_script_metadata")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"script_uid\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["script_uid": "string"]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/scripts/get_script_metadata")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/scripts/get_script_metadata", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"script_uid\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/scripts/get_script_metadata"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"script_uid\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/get_script_metadata"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"script_uid\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ script_uidstring

Unique identifier of the script, returned by the Get Scripts API per script.

REQUEST example-1

```
{ "request_data": { "script_uid": "<unique ID>" } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ script_idstring

Script ID.

▶ namestring

Name of script.

▶ descriptionstring

Description of script.

▶ modification_dateinteger

Timestamp of when the script was last modified.

▶ created_bystring

Name of the user who created the script.

▶ is_high_riskboolean

Whether the script has a high-risk outcome.

▶ windows_supportedboolean

Whether the script can be executed on Windows OS.

▶ linux_supportedboolean

Whether the script can be executed on Linux OS.

▶ macos_supportedboolean

Whether the script can be executed on macOS.

▶ script_uidstring

GUID, global ID of the script, used toidentify the script when executing.

▶ entry_pointstring

name of the entry point selected for the script defined as `run`.

▶ script_inputarray
▶ script_output_typestring (Enum)

Type of output.

▶ script_output_dictionary_definitionsarray

When the `script_output_type` is a dictionary an array with `friendly_name`, `name`, and `type` for each output is returned. The field is empty in all other cases.

RESPONSE When entry_point is returned as run

```
{ "reply": { "script_id": "<script ID>", "name": "list_directories", "description": "List all directories under path", "modification_date": 1585074627259, "created_by": "Palo Alto Networks", "is_high_risk": false, "windows_supported": true, "linux_supported": true, "macos_supported": true, "script_uid": "<unique ID>", "entry_point": "run", "script_input": [ { "name": "path", "type": "string" }, { "friendly_name": "Number of levels", "name": "num_levels", "type": "integer" } ], "script_output_type": "dictionary", "script_output_dictionary_definitions": [ { "friendly_name": "Number Of Processes", "name": "output_2", "type": "integer" }, { "friendly_name": "Name", "name": "output_1", "type": "string" } ] } }
{ "reply": { "script_id": "<script ID>", "name": "list_directories", "description": "List all directories under path", "modification_date": 1585074627259, "created_by": "Palo Alto Networks", "is_high_risk": false, "windows_supported": true, "linux_supported": true, "macos_supported": true, "script_uid": "<unique ID>" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.4 | Get Script Execution Status

post /public_api/v1/scripts/get_script_execution_status

Retrieve the status of a script execution action.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}
Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"action_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/scripts/get_script_execution_status", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"action_id\":\"string\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "action_id": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{\"action_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": ["action_id": "string"]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"action_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"action_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/get_script_execution_status"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"action_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ action_idstring

Identifier of the action, can be found in Cortex XDR console **Response** > **Action Center** > **Action ID** field.

REQUEST example-1

```
{ "request_data": { "action_id": "<action ID>" } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ general_statusstring

General status of the action, considering the status of all the endpoints.

▶ endpoints_pendinginteger

Number of endpoints in pending status.

▶ endpoints_canceledinteger

Number of endpoints in 'canceled' status.

▶ endpoints_in_progressinteger

Number of endpoints in 'in progress' status.

▶ endpoints_timeoutinteger

Number of endpoints in 'timeout' status.

▶ endpoints_failedinteger

Number of endpoints in 'failed' status.

▶ endpoints_completed_successfullyinteger

Number of endpoints in 'completed successfully' status.

▶ endpoints_pending_abortinteger

Number of endpoints in 'pending abort' status.

▶ endpoints_abortedinteger

Number of endpoints in 'aborted' status.

▶ endpoints_expiredinteger

Number of endpoints in 'expired' status.

▶ error_messagestring

Error message regarding permissions for running APIs or stating that the action doesnâ€ t exist.

RESPONSE example-1
```
{ "reply": { "general_status": "PENDING", "endpoints_pending": 1, "endpoints_canceled": 0, "endpoints_in_progress": 0,
"endpoints_timeout": 0, "endpoints_failed": 0, "endpoints_completed_successfully": 0, "endpoints_pending_abort": 0,
"endpoints_aborted": 0, "endpoints_expired": 0 } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.5 | Get Scripts

post /public_api/v1/scripts/get_scripts

Get a list of scripts available in the scripts library.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/get_scripts'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"name\",\"operator\":\"in\",\"value\":[\"string\"]}]}}" headers = { 'Authorization': "SOME_STRING_VALUE",
'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/scripts/get_scripts", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/scripts/get_scripts")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"name\",\"operator\":\"in\",\"value\":[\"string\"]}]}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "name", "operator": "in", "value": [ "string"
] } ] } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange",
function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST",
"https://api-yourfqdn/public_api/v1/scripts/get_scripts"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/get_scripts")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":[{\"field\":\"name\",\"operator\":\"in\",\"value\":
[\"string\"]}]}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["filters": [ [ "field": "name", "operator":
"in", "value": ["string"] ] ]]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters,
options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/scripts/get_scripts")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0)
request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session =
URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/scripts/get_scripts", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"name\",\"operator\":\"in\",\"value\":[\"string\"]}]}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/scripts/get_scripts"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"name\",\"operator\":\"in\",\"value\":[\"string\"]}]}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/get_scripts"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"filters\":[{\"field\":\"name\",\"operator\":\"in\",\"value\":[\"string\"]}]}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields. An empty dictionary returns all results.

▶ filtersarray

An array of filter fields.

REQUEST Request all results

```
{ "request_data": {} }
{ "request_data": { "filters": [ { "field": "is_high_risk", "operator": "in", "value": [ "false" ] } ] } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ total_countinteger

Number of total results of this filter without paging.

▶ result_countinteger

Number of scripts returned as result.

▶ scriptsarray

An array of scripts.

RESPONSE example-1

{ "reply": { "total_count": 129, "result_count": 24, "scripts": [{ "script_id": "<script ID>", "name": "list_directories", "description": "List all directories under path", "modification_date": 1585074627259, "created_by": "Palo Alto Networks", "is_high_risk": false, "windows_supported": true, "linux_supported": true, "macos_supported": true, "script_uid": "<unique ID>" }, { "script_id": "<script ID>", "name": "test 1", "description": "test", "modification_date": 1583052236449, "created_by": "User 1", "is_high_risk": false, "windows_supported": true, "linux_supported": false, "macos_supported": false, "script_uid": "<unique ID>" }, { "script_id": "<script ID>", "name": "test 2", "description": "test 2", "modification_date": 1582709343498, "created_by": "User 2", "is_high_risk": false, "windows_supported": true, "linux_supported": true, "macos_supported": true, "script_uid": "<unique ID>" }] }
}

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.6 | Get Script Execution Results

post /public_api/v1/scripts/get_script_execution_results

Retrieve the results of a script execution action.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"action_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/scripts/get_script_execution_results",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results") http = Net::HTTP.new(url.host, url.port) http.use_ssl =
true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"action_id\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "action_id": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{\"action_id\":\"string\"}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": ["action_id": "string"]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"action_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"action_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"action_id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ action_idstring

Action ID. This can be found in the Cortex XDR console Response > Action Center > Action ID field.

REQUEST example-1

```
{ "request_data": { "action_id": "<action_id>" } }
```

Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ script_namestring

Name of the script executed.

▶ script_descriptionstring

Description of the script executed.

▶ script_parametersarray

For each input parameter used in this execution, an array of `name` and `value`.

▶ date_createdstring

Timestamp in which the action was initiated.

▶ scopestring

Number of endpoints included in this action according to the filter used to select them.
▶ error_messagestring

Error message regarding permissions for running APIs.

▶ resultsarray

For each endpoint Cortex XDR displays any returned value by the script. The number of the results and their name are dynamic per script.

RESPONSE example-1

```
{ "reply": { "script_name": "snippet script", "script_description": "", "script_parameters": [], "date_created": "2020-03-29 13:21:59", "scope": "win_10and 21 other endpoints", "error_message": "", "results": [ { "endpoint_name": "<name>", "endpoint_ip_address": [ "<IP address>" ], "endpoint_status": "LOST", "domain": "aaaa", "endpoint_id": "<endpoint ID>", "execution_status": "PENDING", "standard_output": null, "retrieved_files": 0, "failed_files": 0, "retention_date": null }, { "endpoint_name": "<name>", "endpoint_ip_address": [ "<IP address>" ], "endpoint_status": "LOST", "domain": "<domain name>", "endpoint_id": "<endpoint ID>", "execution_status": "PENDING", "standard_output": null, "retrieved_files": 0, "failed_files": 0, "retention_date": null }, { "endpoint_name": "<name>", "endpoint_ip_address": [ "<IP address>" ], "endpoint_status": "DISCONNECTED", "domain": "WORKGROUP", "endpoint_id": "<endpoint ID>", "execution_status": "PENDING", "standard_output": null, "retrieved_files": 0, "failed_files": 0, "retention_date": null } ] } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.
▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.7 | Get Script Execution Result Files

post /public_api/v1/scripts/get_script_execution_results_files

Get the files retrieved from a specific endpoint during a script execution.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results_files'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"action_id\":\"string\",\"endpoint_id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-
id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/scripts/get_script_execution_results_files", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results_files") http = Net::HTTP.new(url.host, url.port)
http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url)
request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"]
= 'application/json' request.body = "{\"request_data\":{\"action_id\":\"string\",\"endpoint_id\":\"string\"}}"
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "action_id": "string", "endpoint_id": "string" } }); const xhr = new
XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results_files"); xhr.setRequestHeader("Authorization",
"SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type",
"application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results_files") .header("Authorization", "SOME_STRING_VALUE")
.header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":
{\"action_id\":\"string\",\"endpoint_id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "action_id": "string", "endpoint_id":
"string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results_files")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/scripts/get_script_execution_results_files", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING
=> "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"action_id\":\"string\",\"endpoint_id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results_files"); struct curl_slist *headers = NULL;
headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-
auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json");
curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"action_id\":\"string\",\"endpoint_id\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/get_script_execution_results_files"); var
request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-
xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json");
request.AddParameter("application/json", "{\"request_data\":{\"action_id\":\"string\",\"endpoint_id\":\"string\"}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ action_idstring

Identifier of the action, can be found in Cortex XDR console **Response** > **Action Center** > **Action ID** field.

▶ endpoint_idstring

Endpoint ID.

REQUEST
```
{ "request_data": { "action_id": "example", "endpoint_id": "example" } }
```
Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ datastring

A signed public link to a zip file containing the retrieved files. Link expires after 10 minutes.

RESPONSE example-1
```
{ "reply": { "DATA": "https://example-link" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.6.8 | Get Script Code

post /public_api/v1/scripts/get_script_code

Get the code of a specific script in the script library.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/scripts/get_script_code'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"script_uid\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/scripts/get_script_code", payload, headers)
res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/scripts/get_script_code") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"script_uid\":\"string\"}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "script_uid": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/scripts/get_script_code"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/scripts/get_script_code")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"script_uid\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["script_uid": "string"]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/scripts/get_script_code")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/scripts/get_script_code", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"script_uid\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/scripts/get_script_code"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"script_uid\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/scripts/get_script_code"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"script_uid\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ script_uidstring

Unique identifier of the script, returned by the Get Scripts API per script.

REQUEST example-1

```
{ "request_data": { "script_uid": "<unique ID>" } }
```

Responses

Successful response

Body
application/json
▼ replystring

JSON object containing the query result.

RESPONSE example-1

```
{ "reply": "import os\nimport sys\nimport traceback\n\n\ndef run(path, num_levels):\n\tpath =
os.path.expanduser(path)\n\tpath = os.path.expandvars(path)\n\treturn scan_directory_recursive(path,
num_levels)\n\n\ndef scan_directory_recursive(directory, level):\n\tif level == 0:\n\t\treturn []\n\n\tsubfolders =
[]\n\ttry:\n\t\twith os.scandir(directory) as entries:\n\t\t\tfor f in entries:\n\t\t\t\ttry:\n\t\t\t\t\tif
f.is_dir():\n\t\t\t\t\t\tsubfolders.append(f.path)\n\t\t\t\texcept
(OSError,PermissionError):\n\t\t\t\t\tsys.stderr.write(f\"Failed access <{f.path}>\\n\")\n\n\t\tfor folder in
list(subfolders):\n\t\t subfolders.extend(scan_directory_recursive(folder, level-1))\n\texcept
Exception:\n\t\tsys.stderr.write(f\"Failed scanning directory: <{directory}>, error:
{traceback.format_exc()}\")\n\n\treturn subfolders\n" }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.7 | Audit log

APIs for audit logs

2.1.7.1 | Get Audit Management Log

post /public_api/v1/audits/management_logs

Get audit management logs.

- Response is concatenated using AND condition (OR is not supported).
- Maximum result set size is 100.
- Offset is the zero-based number of cases from the start of the result set.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management
Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json,application/xml'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/audits/management_logs'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"email\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/audits/management_logs", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/audits/management_logs") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":
[{\"field\":\"email\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "email", "operator": "in", "value": "string" }
], "search_from": 0, "search_to": 100, "sort": { "field": "type", "keyword": "asc" } } }); const xhr = new
XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/audits/management_logs"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/audits/management_logs")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"email\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "email", "operator":
"in", "value": "string" ] ], "search_from": 0, "search_to": 100, "sort": [ "field": "type", "keyword": "asc" ] ]] as
[String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/audits/management_logs")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields =
headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with:
request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else
{ let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/audits/management_logs", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"email\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-
type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/audits/management_logs"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
```

```
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"email\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":100,\"sort\":{\"field\":\"type\",\"keyword\":\"asc\"}}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/audits/management_logs"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"filters\":[{\"field\":\"email\",\"operator\":\"in\",\"value\":\"string\"}],\"search_from\":0,\"search_to\":100,\"sort\":{\"field\":\"type\",\"keyword\":\"asc\"}}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobject

A dictionary containing the API request fields. An empty dictionary returns all results.

▶ filtersarray

Array of filter fields.

▶ search_frominteger

An integer representing the starting offset within the query result set from which you want management logs returned. Management logs are returned as a zero-based list. Any log indexed less than this value is not returned in the final result set and defaults to zero.

▶ search_tointeger

An integer representing the end offset within the result set after which you do not want management logs returned. Logs in the management log list that are indexed higher than this value are not returned in the final results set. Defaults to 100, which returns all logs to the end of the list.

▶ sortobjectrequired

Identifies the sort order for the result set. By default the sort is defined as creation-time and desc.

REQUEST Request all management logs from older to newer
```
{ "request_data": { "search_from": 0, "search_to": 100, "sort": { "field": "timestamp", "keyword": "asc" } } }
{ "request_data": {} }
{ "request_data": { "search_from": 0, "search_to": 100, "sort": { "field": "timestamp", "keyword": "asc" }, "filters": [ { "field": "type", "operator": "in", "value": [ "AUTH" ] }, { "field": "sub_type", "operator": "in", "value": [ "login" ] }, { "field": "result", "operator": "in", "value": [ "SUCCESS" ] }, { "field": "timestamp", "operator": "gte", "value": 1565074114053 } ] } }
```
Responses

Successful response

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ total_countinteger

Number of total results of this filter without paging.

▶ result_countinteger

Number of returned items.

▶ dataarray

List of audit items.

RESPONSE example-1
```
{ "reply": { "total_count": 1, "result_count": 1, "data": [ { "AUDIT_ID": 1, "AUDIT_OWNER_NAME": "User Name", "AUDIT_OWNER_EMAIL": "username@paloaltonetworks.com", "AUDIT_ASSET_JSON": "{}", "AUDIT_ASSET_NAMES": "", "AUDIT_HOSTNAME": "", "AUDIT_RESULT": "SUCCESS", "AUDIT_REASON": "", "AUDIT_DESCRIPTION": "", "AUDIT_ENTITY": "AUTH", "AUDIT_ENTITY_SUBTYPE": "Login", "AUDIT_SESSION_ID": 382303947890, "AUDIT_CASE_ID": 473829372, "AUDIT_INSERT_TIME": 1565074114053, "AUDIT_SEVERITY": "SEV_020_LOW", "AUDIT_LINK": null, "AUDIT_SOURCE_IP": "31.174.156.148", "AUDIT_USER_AGENT": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36", "AUDIT_USER_ROLES": [ "Account Admin" ], "AUDIT_ADDITIONAL_INFORMATION": { "endpoint_names": [ "WIN-fgo6762G" ], "endpoint_count": 1 }, "AUDIT_OBJECT_ID": null } ] } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```
RESPONSE
```
<root/>
```

2.1.7.2 | Get Audit Agent Report

post /public_api/v1/audits/agents_reports

Get agent event reports.

- Response is concatenated using AND condition (OR is not supported).
- Maximum result set size is 100.
- Offset is the zero-based number of cases from the start of the result set.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/audits/agents_reports'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":[\"string\"]}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
```

```
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/audits/agents_reports",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/audits/agents_reports") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"search_from\":0,\"search_to\":100,\"sort\":{\"field\":\"type\",\"keyword\":\"asc\"}}}" response =
http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "endpoint_id", "operator": "in", "value": [
"string" ] } ], "search_from": 0, "search_to": 100, "sort": { "field": "type", "keyword": "asc" } } }); const xhr =
new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/audits/agents_reports"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/audits/agents_reports")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"search_from\":0,\"search_to\":100,\"sort\":{\"field\":\"type\",\"keyword\":\"asc\"}}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ [ "field": "endpoint_id",
"operator": "in", "value": ["string"] ] ], "search_from": 0, "search_to": 100, "sort": [ "field": "type", "keyword":
"asc" ] ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/audits/agents_reports")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields =
headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with:
request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else
{ let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/audits/agents_reports", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":[\"string\"]}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-
type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/audits/agents_reports"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":[\"string\"]}],\"search_from\":0,\"search_to\":100,\"sort\":
{\"field\":\"type\",\"keyword\":\"asc\"}}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/audits/agents_reports"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"endpoint_id\",\"operator\":\"in\",\"value\":
[\"string\"]}],\"search_from\":0,\"search_to\":100,\"sort\":{\"field\":\"type\",\"keyword\":\"asc\"}}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

A dictionary containing the API request fields. An empty dictionary returns all results.

▶ filtersarray

An array of filter fields.

▶ search_frominteger

An integer representing the starting offset within the query result set from which you want agent reports returned. Reports are returned as a zero-based list. Any report indexed less than this value is not returned in the final result set and defaults to zero.

▶ search_tointeger

An integer representing the end offset within the result set after which you do not want agent reports returned. Reports in the agent report list that are indexed higher than this value are not returned in the final results set. Defaults to 100, which returns all reports to the end of the list.

▶ sortobjectrequired

Identifies the sort order for the result set.

REQUEST Request all results

```
{ "request_data": {} }
{ "request_data": { "filters": [ { "field": "trapsversion", "operator": "in", "value": [ "<version value>", "<version value>" ] }, { "field": "timestamp", "operator": "gte", "value": 0 }, { "field": "domain", "operator": "in", "value": [ "WORKGROUP" ] } ], "sort": { "field": "timestamp", "keyword": "asc" } } }
```

Responses

OK

Body
application/json

▼ replyobject

JSON object containing the query result.

▶ total_countinteger

Number of total results of this filter without paging.

▶ result_countinteger

Number of returned items.

▶ dataarray

List of audit items.

RESPONSE

```
{ "reply": { "total_count": 0, "result_count": 0, "data": [ { "TIMESTAMP": 0.1, "RECEIVEDTIME": 0.1, "ENDPOINTID": "example", "ENDPOINTNAME": "example", "DOMAIN": "example", "TRAPSVERSION": "example", "CATEGORY": "example", "TYPE": "example", "SUBTYPE": "example", "RESULT": "example", "REASON": "example", "DESCRIPTION": "example" } ] } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, ID, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8 | System management

APIs for system management

2.1.8.1 | System Health Check

get /public_api/v1/healthcheck

Perform a health check of your Cortex Cloud environment.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'GET'
-H 'Accept: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/healthcheck'
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE" } conn.request("GET", "/public_api/v1/healthcheck",
headers=headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/healthcheck") http =
Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Get.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
response = http.request(request) puts response.read_body
const data = null; const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("GET", "https://api-yourfqdn/public_api/v1/healthcheck");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.send(data);
HttpResponse<String> response = Unirest.get("https://api-yourfqdn/public_api/v1/healthcheck") .header("Authorization",
"SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE" ] let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/healthcheck")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "GET" request.allHTTPHeaderFields =
headers let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/healthcheck",
CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30,
CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "GET", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "GET"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/healthcheck"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/healthcheck"); var request = new
RestRequest(Method.GET); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); IRestResponse response = client.Execute(request);
```

Responses

OK

Body
application/json
▼ statusstring

The condition of your Cortex environment.

RESPONSE

```
{ "status": "example" }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8.2 | Get Tenant Info

post /public_api/v1/system/get_tenant_info

Get your tenant license information.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/system/get_tenant_info'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/system/get_tenant_info", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/system/get_tenant_info") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": {} }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/system/get_tenant_info");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/system/get_tenant_info")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": []] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/system/get_tenant_info")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/system/get_tenant_info", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/system/get_tenant_info"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/system/get_tenant_info"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

REQUEST

```
{ "request_data": {} }
```

Responses

OK

Body

application/json

▼ replyobject

List of available licenses, number of devices, and purchased add-ons on your tenant.

- ▶ xsiam_premium_expirationinteger
- ▶ purchased_xsiam_premiumobject
- ▶ pro_per_endpoint_expirationinteger
- ▶ purchased_pro_per_endpointobject
- ▶ data_enabled_pro_per_endpointinteger
- ▶ prevent_expirationinteger
- ▶ purchased_preventinteger
- ▶ installed_preventinteger
- ▶ pro_gb_expirationinteger
- ▶ purchased_pro_gbobject
- ▶ installed_pro_gbinteger
- ▶ compute_unit_expirationinteger
- ▶ purchased_compute_unitinteger
- ▶ host_insights_expirationinteger
- ▶ enabled_host_insightsinteger
- ▶ purchased_host_insightsinteger

▶ forensics_expirationinteger
▶ enabled_forensicsinteger
▶ pro_cloud_expirationinteger
▶ purchased_pro_cloudobject
▶ installed_pro_cloudinteger
▶ data_enabled_pro_cloudinteger
▶ identity_threat_expirationinteger
▶ xth_expirationinteger
▶ purchased_xthinteger
▶ threat_intelligence_management_expirationinteger
▶ purchased_threat_intelligence_managementinteger
▶ attack_surface_management_expirationinteger
▶ purchased_attack_surface_managementinteger
▶ xsiam_ep_hot_expirationstring
▶ purchased_xsiam_ep_hotinteger
▶ xsiam_ep_cold_expirationstring
▶ purchased_xsiam_ep_coldinteger
▶ xsiam_gb_hot_expirationstring
▶ purchased_xsiam_gb_hotinteger
▶ xsiam_gb_cold_expirationstring
▶ purchased_xsiam_gb_coldinteger
RESPONSE Example 1

{ "reply": { "xsiam_premium_expiration": 12478046378, "purchased_xsiam_premium": { "users": 100, "gb": 100, "agents": 500 }, "pro_per_endpoint_expiration": 12478046378, "purchased_pro_per_endpoint": { "agents": 200 }, "data_enabled_pro_per_endpoint": 26, "prevent_expiration": 0, "purchased_prevent": 0, "installed_prevent": 27, "pro_tb_expiration": 12478046378, "purchased_pro_gb": { "gb": 1 }, "installed_pro_tb": 0, "compute_unit_expiration": 0, "purchased_compute_unit": 16, "host_insights_expiration": 12478046378, "enabled_host_insights": 26, "purchased_host_insights": 400, "forensics_expiration": 0, "enabled_forensics": 12 } }
{ "reply": { "pro_cloud_expiration": 0, "purchased_pro_cloud": 0, "installed_pro_cloud": 4, "data_enabled_pro_cloud": 4, "pro_per_endpoint_expiration": "Nov 7th 2025 07:59:59", "purchased_pro_per_endpoint": { "agents": 200 }, "data_enabled_pro_per_endpoint": 4, "prevent_expiration": 0, "purchased_prevent": 0, "installed_prevent": 4, "pro_gb_expiration": "Nov 8th 2027 07:59:59", "purchased_pro_gb": { "gb": 33 }, "identity_threat_expiration": "Aug 13th 2023 11:03:41", "compute_unit_expiration": "Nov 7th 2025 07:59:59", "purchased_compute_unit": 50, "host_insights_expiration": "Nov 8th 2027 07:59:59", "enabled_host_insights": 8, "purchased_host_insights": 250, "forensics_expiration": 0, "enabled_forensics": 4, "xsiam_ep_hot_expiration": "Jan 2nd 2024 23:59:59", "purchased_xsiam_ep_hot": 2 } }

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal Server Error

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8.3 | Get Users

post /public_api/v1/rbac/get_users

Retrieve a list of the current users in your environment.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/rbac/get_users'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/rbac/get_users", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-
8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/rbac/get_users") http
= Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/rbac/get_users");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/rbac/get_users")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = [] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/rbac/get_users")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/rbac/get_users", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type:
application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/rbac/get_users"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/rbac/get_users"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
Object
application/json
REQUEST

```
{}
```

Responses

OK

Body
application/json
▼ replyarray

JSON object containing the query result.

[
▶ user_emailstring

Email address of the user.

▶ user_first_namestring

First name of the user.

▶ user_last_namestring

Last name of the user.

▶ role_namestring

Role name associated with the user.

▶ last_logged_ininteger

Timestamp of when the user last logged in.

▶ user_typestring

Type of user.

▶ groupsarray

Name of user groups associated with the user, if applicable.

▶ scopearray

Name of scope associated with the user, if applicable.

]
RESPONSE Example 1

```
{ "reply": [ { "user_email": "user1@acme.com", "user_first_name": "<first name>", "user_last_name": "<last name>",
"role_name": "Account Admin", "last_logged_in": 1640024700241, "user_type": "CSP", "groups": [], "scope": [] }, {
"user_email": "user2@acme.com", "user_first_name": "<first name>", "user_last_name": "<last name>", "role_name":
"Account Admin", "last_logged_in": null, "user_type": "CSP", "groups": [], "scope": [] }, { "user_email":
"user3@acme.com", "user_first_name": "<first name>", "user_last_name": "<last name>", "role_name": "Investigator",
"last_logged_in": null, "user_type": "CSP", "groups": [], "scope": [] } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body

application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8.4 | Get Roles

post /public_api/v1/rbac/get_roles

Retrieve information about one or more roles created in your environment.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/rbac/get_roles'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"role_names\":
[\"string\"]}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-
type': "application/json" } conn.request("POST", "/public_api/v1/rbac/get_roles", payload, headers) res =
conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/rbac/get_roles") http
= Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"role_names\":[\"string\"]}}"
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "role_names": [ "string" ] } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/rbac/get_roles");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/rbac/get_roles")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"role_names\":[\"string\"]}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["role_names": ["string"]]] as [String : Any]
let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/rbac/get_roles")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```
```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/rbac/get_roles", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"role_names\":[\"string\"]}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
```
```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/rbac/get_roles"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"role_names\":
[\"string\"]}}"); CURLcode ret = curl_easy_perform(hnd);
```
```
var client = new RestClient("https://api-yourfqdn/public_api/v1/rbac/get_roles"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"role_names\":[\"string\"]}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```
Body parameters
application/json
▼ request_dataobject

A dictionary containing the API request fields.

▶ role_namesarray[string]

List of one or more role names in your environment for which you want detailed information.

REQUEST Example 1

```
{ "request_data": { "role_names": [ "Role1", "Role2" ] } }
```
Responses

OK

Body
application/json
▼ replyarray

JSON object containing the query result.

[
▶ pretty_namestring

Name of the role as it appears in the Management Console.

▶ permissionsarray[string]

List of permissions associated with this role.

▶ insert_timeinteger

Timestamp of when the Role was created.

▶ update_timeinteger

Timestamp of when the Role was last updated.

▶ created_bystring

Email of the user who created the Role.

▶ descriptionstring

Description of the Role, if available.

▶ groupsarray[string]

Group names associated with the Role.

▶ usersarray[string]

Email addresses of users associated with the Role.

]
RESPONSE Example 1

{ "reply": [[{ "pretty_name": "Role1", "permissions": ["Reports", "Playbooks", "Datasets Access Control", "Dashboards", "Scripts"], "insert_time": 1658315576844, "update_time": null, "created_by": "user1@acme.com", "description": "", "groups": ["group1", "group2"], "users": [] }], [{ "pretty_name": "Role2", "permissions": ["Dashboards", "Datasets Access Control"], "insert_time": 1661435660656, "update_time": null, "created_by": "user1@acme.com", "description": "", "groups": [], "users": [] }]] }

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8.5 | Get User Groups

post /public_api/v1/rbac/get_user_group

Retrieve a list of the current user emails associated with one or more user groups in your environment.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
```

```
'https://api-yourfqdn/public_api/v1/rbac/get_user_group'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"group_names\":
[\"string\"]}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-
type': "application/json" } conn.request("POST", "/public_api/v1/rbac/get_user_group", payload, headers) res =
conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/rbac/get_user_group")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"group_names\":[\"string\"]}}"
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "group_names": [ "string" ] } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/rbac/get_user_group");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/rbac/get_user_group")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"group_names\":[\"string\"]}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["group_names": ["string"]]] as [String : Any]
let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/rbac/get_user_group")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/rbac/get_user_group", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"group_names\":[\"string\"]}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/rbac/get_user_group"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"group_names\":
[\"string\"]}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/rbac/get_user_group"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"group_names\":[\"string\"]}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject

A dictionary containing the API request fields.

▶ group_namesarray[string]

List of one or more user group names for which you want the associated users.

REQUEST Example 1

```
{ "request_data": { "group_names": [ "Group1", "Group2" ] } }
```

Responses

OK

Body
application/json
▼ replyarray

JSON object containing the query result.

[
▶ group_namestring

Name of the User Group.
▶ descriptionobject

Description of the User Group, if available.

▶ pretty_namestring

Name of the User Group as it appears in the Management Console.

▶ insert_timeinteger

Timestamp of when the User Group was created.

▶ update_timeinteger

Timestamp of when the User Group was last updated.

▶ user_emailarray[string]

List of email addresses belonging to the users associated with the User Group.

▶ sourcestring

Type of User Group.

]
RESPONSE Example 1

```
{ "reply": [ { "group_name": "Group1", "description": null, "pretty_name": "Investigator", "insert_time": 1661170832341, "update_time": 1661171650679, "user_email": [ "user1@acme.com", "user2@pacme.com", "user3@acme.com", "user4@acme.com", "user5@acme.com" ], "source": "Custom" }, { "group_name": "Group2", "description": null, "pretty_name": "Instance Administrator", "insert_time": 1660830450590, "update_time": 1661171631589, "user_email": [ "user1@acme.com", "user2@acme.com", "user3@acme.com", "user4@acme.com" ], "source": "Custom" } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

2.1.8.6 | Set a User Role

post /public_api/v1/rbac/set_user_role

Add or remove one or more users from a role.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
```

```
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/rbac/set_user_role'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"user_emails\":
[\"string\"],\"role_name\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/rbac/set_user_role",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/rbac/set_user_role")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"user_emails\":
[\"string\"],\"role_name\":\"string\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "user_emails": [ "string" ], "role_name": "string" } }); const xhr =
new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/rbac/set_user_role"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/rbac/set_user_role")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"user_emails\":[\"string\"],\"role_name\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "user_emails": ["string"], "role_name":
"string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/rbac/set_user_role")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields =
headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with:
request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else
{ let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/rbac/set_user_role", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"user_emails\":[\"string\"],\"role_name\":\"string\"}}", CURLOPT_HTTPHEADER
=> [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/rbac/set_user_role"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"user_emails\":
[\"string\"],\"role_name\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/rbac/set_user_role"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"user_emails\":[\"string\"],\"role_name\":\"string\"}}", ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

A dictionary containing the API request fields.

▶ user_emailsarray[string]

List of one or more user emails of users you want to add to or remove from a role.

▶ role_namestring

Name of the role you want to add a user to. Send an empty field to remove the user.

REQUEST Example 1

```
{ "request_data": { "user_emails": [ "user1@acme.com", "user2@acme.com" ], "role_name": "Role1" } }
```

Responses

OK

Body

application/json
▼ replyobject

JSON object containing the query result.

▶ update_countstring

Number of updated users.

RESPONSE Example 1
```
{ "reply": { "update_count": "2" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Internal Server Error

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

2.1.8.7 | Get Risk Score

post /public_api/v1/get_risk_score

Retrieve the risk score of a specific user or endpoint in your environment, along with the reason for the score.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/get_risk_score'
```

```
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"id\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/get_risk_score", payload, headers) res =
conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/get_risk_score") http
= Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"id\":\"string\"}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "id": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/get_risk_score");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/get_risk_score")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"id\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["id": "string"]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/get_risk_score")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/get_risk_score", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"id\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/get_risk_score"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"id\":\"string\"}}");
CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/get_risk_score"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"id\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired

A dictionary containing the API request fields.

▶ idstring

Unique ID of a specific user or endpoint.

- User ID should be in the following format: `netBIOS/samAccount`
- Endpoint ID is the Cortex Agent ID.

You can only request one ID at a time.

REQUEST Example 1

```
{ "request_data": { "id": "<user or endpoint ID>" } }
```

Responses

OK

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ typestring

Form of identification element.

▶ idstring

Identification value of the type field.

▶ scoreinteger

The score assigned to the type.

▶ norm_risk_scoreinteger

Normalization of the risk score.

▶ risk_levelstring (Enum)

The risk level.

▶ reasonsarray

Details describing when and which case name affected the score.

▶ emailstring

Email address.

RESPONSE Example 1

```
{ "reply": { "type": "user", "id": "user2_9fa235", "score": 100, "norm_risk_score": 800, "risk_level": "HIGH", "reasons": [ { "date created": "2023-01-31", "description": "'Encoded VBScript executed' generated by XDR BIOC detected on host acme3 involving user user2_9fa235", "severity": "SEV_050_CRITICAL", "status": "STATUS_010_NEW", "points": 95 } ], "email": "user@company.com" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`

▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8.8 | Get Risky Users

post /public_api/v1/get_risky_users

Retrieve a list of users with the highest risk score in your environment along with the reason affecting each score.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/get_risky_users'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/get_risky_users", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-
8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/get_risky_users")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{}" response = http.request(request) puts
response.read_body
```

```
const data = JSON.stringify({}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/get_risky_users");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/get_risky_users")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = [] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/get_risky_users")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/get_risky_users", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type:
application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/get_risky_users"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/get_risky_users"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

Object

application/json

REQUEST

`{}`
Responses

OK

Body
application/json
▼ replyarray

JSON object containing the query result.

[
▶ typestring

Form of identification element.

▶ idstring

Identification value of the type field.

▶ scoreinteger

The score assigned to the user.

▶ norm_risk_scoreinteger

Normalization of the risk score.

▶ risk_levelstring (Enum)

The risk level.

▶ reasonsarray

Details describing when and which case name affected the user score.

▶ emailstring

Email address.

]
RESPONSE Example 1

```
{ "reply": [ { "type": "user", "id": "acme1_9fa235", "score": 100, "norm_risk_score": 800, "risk_level": "HIGH",
"reasons": [ { "date created": "2023-01-31", "description": "'Encoded VBScript executed' generated by XDR BIOC
detected on host acme_agent-6a7b involving user acme1_9fa235", "severity": "SEV_050_CRITICAL", "status":
"STATUS_010_NEW", "points": 95 } ], "email": "user@company.com" }, { "type": "user", "id": "acme2_d40b59", "score":
100, "norm_risk_score": 800, "risk_level": "HIGH", "reasons": [ { "date created": "2023-01-26", "description":
"'Kernel Privilege Escalation' along with 4 other alerts generated by XDR Analytics BIOC, XDR Analytics and XDR Agent
detected on 3 hosts involving 3 users", "severity": "SEV_040_HIGH", "status": "STATUS_010_NEW", "points": 95 } ] }, {
"type": "user", "id": "acme3_3a509d", "score": 15, "norm_risk_score": 100, "risk_level": "LOW", "reasons": [ { "date
created": "2023-02-13", "description": "'First successful SSO connection from a country in organization' generated by
XDR Analytics BIOC involving user acme3", "severity": "SEV_020_LOW", "status": "STATUS_010_NEW", "points": 15 } ],
"email": "user@company.com" }, { "type": "user", "id": "acme4_207a4e", "score": 0, "norm_risk_score": 0, "risk_level":
"LOW", "reasons": [], "email": "user@company.com" } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.8.9 | Get Risky Hosts

post /public_api/v1/get_risky_hosts

Retrieve a list of endpoints with the highest risk score in your environment along with the reason for each score.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/get_risky_hosts'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/get_risky_hosts", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-
8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/get_risky_hosts")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/get_risky_hosts");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/get_risky_hosts")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = [] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/get_risky_hosts")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/get_risky_hosts", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type:
application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl);
curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/get_risky_hosts"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/get_risky_hosts"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
Object
application/json
REQUEST

```
{}
```

Responses

OK

Body
application/json
▼ replyarray

JSON object containing the query result.

[
▶ typestring

Form of identification element.

▶ idstring

Identification value of the type field.

▶ scoreinteger

The score assigned to the endpoint.

▶ norm_risk_scoreinteger

Normalization of the risk score.

▶ risk_levelstring (Enum)

The risk level.

▶ reasonsarray

Details describing when and which case name affected the endpoint score.

]
RESPONSE Example 1

```
{ "reply": [ { "type": "host", "id": "host1", "score": 100, "norm_risk_score": 800, "risk_level": "HIGH", "reasons": [
{ "date created": "2023-01-26", "description": "'Kernel Privilege Escalation' along with 4 other alerts generated by
XDR Analytics BIOC, XDR Analytics and XDR Agent detected on 3 hosts involving 3 users", "severity": "SEV_040_HIGH",
"status": "STATUS_010_NEW", "points": 95 } ] } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.
Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.9 | API keys

APIs for managing API keys

2.1.9.1 | Get existing API keys

post /public_api/v1/api_keys/get_api_keys

Get a list of API keys filtered by expiration date, role, or ID.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ authorization String required

api_key

Example:
`DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm`
▼ x-xdr-auth-id String required

api_key_id

Example: `2841`
▼ x-xdr-timestamp String required

timestamp in milliseconds

Example: `xXdrTimestamp_example`
▼ x-xdr-nonce String required

64 byte random string

Example: `xXdrNonce_example`
▼ x-child-tenant-id String required

child tenant ID

Example: `xChildTenantId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
-H 'x-xdr-auth-id: 2841' -H 'x-xdr-timestamp: xXdrTimestamp_example' -H 'x-xdr-nonce: xXdrNonce_example' -H 'x-child-
tenant-id: xChildTenantId_example'
```

```
'https://api-yourfqdn/public_api/v1/api_keys/get_api_keys'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"expiration\",\"operator\":\"gte\",\"value\":0}]}}" headers = { 'authorization':
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
'x-xdr-auth-id': "2841", 'x-xdr-timestamp': "SOME_STRING_VALUE", 'x-xdr-nonce': "SOME_STRING_VALUE", 'x-child-tenant-
id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/api_keys/get_api_keys", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/api_keys/get_api_keys") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["authorization"] =
'DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
request["x-xdr-auth-id"] = '2841' request["x-xdr-timestamp"] = 'SOME_STRING_VALUE' request["x-xdr-nonce"] =
'SOME_STRING_VALUE' request["x-child-tenant-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"expiration\",\"operator\":\"gte\",\"value\":0}]}}"
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "expiration", "operator": "gte", "value": 0 }
] } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange",
function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST",
"https://api-yourfqdn/public_api/v1/api_keys/get_api_keys"); xhr.setRequestHeader("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
xhr.setRequestHeader("x-xdr-auth-id", "2841"); xhr.setRequestHeader("x-xdr-timestamp", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-nonce", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-child-tenant-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/api_keys/get_api_keys")
.header("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
.header("x-xdr-auth-id", "2841") .header("x-xdr-timestamp", "SOME_STRING_VALUE") .header("x-xdr-nonce",
"SOME_STRING_VALUE") .header("x-child-tenant-id", "SOME_STRING_VALUE") .header("content-type", "application/json")
.body("{\"request_data\":{\"filters\":[{\"field\":\"expiration\",\"operator\":\"gte\",\"value\":0}]}}") .asString();
import Foundation let headers = [ "authorization":
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
"x-xdr-auth-id": "2841", "x-xdr-timestamp": "SOME_STRING_VALUE", "x-xdr-nonce": "SOME_STRING_VALUE", "x-child-tenant-
id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": ["filters": [ [
"field": "expiration", "operator": "gte", "value": 0 ] ]]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/api_keys/get_api_keys")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/api_keys/get_api_keys", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"expiration\",\"operator\":\"gte\",\"value\":0}]}}", CURLOPT_HTTPHEADER => [ "authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
"content-type: application/json", "x-child-tenant-id: SOME_STRING_VALUE", "x-xdr-auth-id: 2841", "x-xdr-nonce:
SOME_STRING_VALUE", "x-xdr-timestamp: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/api_keys/get_api_keys"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
headers = curl_slist_append(headers, "x-xdr-auth-id: 2841"); headers = curl_slist_append(headers, "x-xdr-timestamp:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-nonce: SOME_STRING_VALUE"); headers =
curl_slist_append(headers, "x-child-tenant-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-
type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd,
CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"expiration\",\"operator\":\"gte\",\"value\":0}]}}");
CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/api_keys/get_api_keys"); var request = new
RestRequest(Method.POST); request.AddHeader("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
request.AddHeader("x-xdr-auth-id", "2841"); request.AddHeader("x-xdr-timestamp", "SOME_STRING_VALUE");
request.AddHeader("x-xdr-nonce", "SOME_STRING_VALUE"); request.AddHeader("x-child-tenant-id", "SOME_STRING_VALUE");
request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":
{\"filters\":[{\"field\":\"expiration\",\"operator\":\"gte\",\"value\":0}]}}", ParameterType.RequestBody);
IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

▶ filtersarray

An array of filter fields.

REQUEST Get API keys and filter by expiration

```
{ "request_data": { "filters": [ { "field": "expiration", "operator": "gte", "value": 1721149909250 } ] } }
{ "request_data": { "filters": [ { "field": "roles", "operator": "contains", "value": [ "Public API Action" ] } ] } }
{ "request_data": { "filters": [ { "field": "id", "operator": "in", "value": [ 85 ] } ] } }
```

Responses

OK

Body

application/json

▼ replyobject

▶ dataarray

▶ filter_countinteger

▶ total_countinteger

Note: The `total_count` value contains all API Keys, including ones that have expired.

RESPONSE Example 1

```
{ "reply": { "DATA": [ { "id": 25, "creation_time": 1709544947832, "created_by": "N/A", "user_name": "Public API -
21", "roles": [ "default" ], "security_level": "standard", "comment": null, "expiration": 1710149747184 }, { "id": 24,
"creation_time": 1709544787367, "created_by": "N/A", "user_name": "Public API - 21", "roles": [ "default" ],
"security_level": "standard", "comment": null, "expiration": 1710149586852 }, { "id": 23, "creation_time":
1709544227320, "created_by": "N/A", "user_name": "Public API - 21", "roles": [ "default" ], "security_level":
"standard", "comment": null, "expiration": 1710149026834 } ], "FILTER_COUNT": 3, "TOTAL_COUNT": 70 } }
```

2.1.9.2 | Generate an API key

post /public_api/v1/api_keys/generate

Generate a new API key and define the roles assigned to it and whether the security level is standard or advanced.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers

▼ authorization String required

api_key

Example:
`DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm`

▼ x-xdr-auth-id String required

api_key_id

Example: `2841`

▼ x-xdr-timestamp String required

timestamp in milliseconds

Example: `xXdrTimestamp_example`

▼ x-xdr-nonce String required

64 byte random string

Example: `xXdrNonce_example`

▼ x-child-tenant-id String required

child tenant ID

Example: `xChildTenantId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
-H 'x-xdr-auth-id: 2841' -H 'x-xdr-timestamp: xXdrTimestamp_example' -H 'x-xdr-nonce: xXdrNonce_example' -H 'x-child-
tenant-id: xChildTenantId_example'
'https://api-yourfqdn/public_api/v1/api_keys/generate'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"roles\":
[\"string\"],\"security_level\":\"standard\",\"expiration\":0,\"comment\":\"string\"}}" headers = { 'authorization':
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
'x-xdr-auth-id': "2841", 'x-xdr-timestamp': "SOME_STRING_VALUE", 'x-xdr-nonce': "SOME_STRING_VALUE", 'x-child-tenant-
id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/api_keys/generate", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/api_keys/generate")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["authorization"] =
'DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
request["x-xdr-auth-id"] = '2841' request["x-xdr-timestamp"] = 'SOME_STRING_VALUE' request["x-xdr-nonce"] =
'SOME_STRING_VALUE' request["x-child-tenant-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"roles\":
[\"string\"],\"security_level\":\"standard\",\"expiration\":0,\"comment\":\"string\"}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "roles": [ "string" ], "security_level": "standard", "expiration": 0,
"comment": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/api_keys/generate");
xhr.setRequestHeader("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
xhr.setRequestHeader("x-xdr-auth-id", "2841"); xhr.setRequestHeader("x-xdr-timestamp", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-nonce", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-child-tenant-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/api_keys/generate")
.header("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
.header("x-xdr-auth-id", "2841") .header("x-xdr-timestamp", "SOME_STRING_VALUE") .header("x-xdr-nonce",
"SOME_STRING_VALUE") .header("x-child-tenant-id", "SOME_STRING_VALUE") .header("content-type", "application/json")
.body("{\"request_data\":{\"roles\":
[\"string\"],\"security_level\":\"standard\",\"expiration\":0,\"comment\":\"string\"}}") .asString();
import Foundation let headers = [ "authorization":
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
"x-xdr-auth-id": "2841", "x-xdr-timestamp": "SOME_STRING_VALUE", "x-xdr-nonce": "SOME_STRING_VALUE", "x-child-tenant-
id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": [ "roles":
["string"], "security_level": "standard", "expiration": 0, "comment": "string" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/api_keys/generate")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/api_keys/generate", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"roles\":
[\"string\"],\"security_level\":\"standard\",\"expiration\":0,\"comment\":\"string\"}}", CURLOPT_HTTPHEADER => [
"authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
"content-type: application/json", "x-child-tenant-id: SOME_STRING_VALUE", "x-xdr-auth-id: 2841", "x-xdr-nonce:
SOME_STRING_VALUE", "x-xdr-timestamp: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/api_keys/generate"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaY4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
```

```
headers = curl_slist_append(headers, "x-xdr-auth-id: 2841"); headers = curl_slist_append(headers, "x-xdr-timestamp: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-nonce: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-child-tenant-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"roles\":
[\"string\"],\"security_level\":\"standard\",\"expiration\":0,\"comment\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/api_keys/generate"); var request = new RestRequest(Method.POST); request.AddHeader("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
request.AddHeader("x-xdr-auth-id", "2841"); request.AddHeader("x-xdr-timestamp", "SOME_STRING_VALUE");
request.AddHeader("x-xdr-nonce", "SOME_STRING_VALUE"); request.AddHeader("x-child-tenant-id", "SOME_STRING_VALUE");
request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":
{\"roles\":[\"string\"],\"security_level\":\"standard\",\"expiration\":0,\"comment\":\"string\"}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

▶ rolesarray[string]

A list of roles to be assigned to the API key.

▶ security_levelobject (Enum)

The security level of the API key. API keys with advanced security are hashed with a nonce and timestamp, which is useful for proprietary scripts and are intended to prevent replay attacks. Standard security API keys can be used as-is and are suitable for curl.

▶ expirationinteger

Integer in timestamp epoch milliseconds. Default value is one week from the time of the API call. Maximum expiration date is six months from the time of the API call.

▶ commentstring

REQUEST Generate a standard API key

```
{ "request_data": { "roles": [ "Public API Action" ], "security_level": "standard", "comment": "Api key for John" } }
{ "request_data": { "roles": [ "Public API Action" ], "security_level": "standard", "expiration": 1725802080000,
"comment": "API key for Joe" } }
```

Responses

OK

Body

application/json

▼ replyobject

▶ idinteger

▶ keystring

RESPONSE Standard API key

```
{ "reply": { "id": 4267, "key": "oZFaXmXCLCdoWIXTb9ITSqUvl4OBLsUQgCeuZc5FdwMpmpY5QBn8OUePZiLMb" } }
```

2.1.9.3 | Delete API keys

post /public_api/v1/api_keys/delete

Delete API keys by ID.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers

▼ authorization String required

api_key

Example:
`DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm`

▼ x-xdr-auth-id String required

api_key_id

Example: `2841`

▼ x-xdr-timestamp String required

timestamp in milliseconds

Example: `xXdrTimestamp_example`

▼ x-xdr-nonce String required

64 byte random string

Example: `xXdrNonce_example`

▼ x-child-tenant-id String required

child tenant ID

Example: `xChildTenantId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
-H 'x-xdr-auth-id: 2841' -H 'x-xdr-timestamp: xXdrTimestamp_example' -H 'x-xdr-nonce: xXdrNonce_example' -H 'x-child-
tenant-id: xChildTenantId_example'
'https://api-yourfqdn/public_api/v1/api_keys/delete'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"id\",\"operator\":\"in\",\"value\":[0]}]}}" headers = { 'authorization':
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
'x-xdr-auth-id': "2841", 'x-xdr-timestamp': "SOME_STRING_VALUE", 'x-xdr-nonce': "SOME_STRING_VALUE", 'x-child-tenant-
id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/api_keys/delete",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/api_keys/delete")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["authorization"] =
'DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
request["x-xdr-auth-id"] = '2841' request["x-xdr-timestamp"] = 'SOME_STRING_VALUE' request["x-xdr-nonce"] =
'SOME_STRING_VALUE' request["x-child-tenant-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"id\",\"operator\":\"in\",\"value\":[0]}]}}" response =
http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "id", "operator": "in", "value": [ 0 ] } ] }
}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function ()
{ if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/api_keys/delete"); xhr.setRequestHeader("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
xhr.setRequestHeader("x-xdr-auth-id", "2841"); xhr.setRequestHeader("x-xdr-timestamp", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-nonce", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-child-tenant-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/api_keys/delete")
.header("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
.header("x-xdr-auth-id", "2841") .header("x-xdr-timestamp", "SOME_STRING_VALUE") .header("x-xdr-nonce",
"SOME_STRING_VALUE") .header("x-child-tenant-id", "SOME_STRING_VALUE") .header("content-type", "application/json")
.body("{\"request_data\":{\"filters\":[{\"field\":\"id\",\"operator\":\"in\",\"value\":[0]}]}}") .asString();
```

```
import Foundation let headers = [ "authorization":
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
"x-xdr-auth-id": "2841", "x-xdr-timestamp": "SOME_STRING_VALUE", "x-xdr-nonce": "SOME_STRING_VALUE", "x-child-tenant-
id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": ["filters": [ [
"field": "id", "operator": "in", "value": [0] ] ]]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/api_keys/delete")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/api_keys/delete", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":[{\"field\":\"id\",\"operator\":\"in\",\"value\":[0]}]}}",
CURLOPT_HTTPHEADER => [ "authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
"content-type: application/json", "x-child-tenant-id: SOME_STRING_VALUE", "x-xdr-auth-id: 2841", "x-xdr-nonce:
SOME_STRING_VALUE", "x-xdr-timestamp: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/api_keys/delete"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "authorization:
DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7Gm
headers = curl_slist_append(headers, "x-xdr-auth-id: 2841"); headers = curl_slist_append(headers, "x-xdr-timestamp:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-nonce: SOME_STRING_VALUE"); headers =
curl_slist_append(headers, "x-child-tenant-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-
type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd,
CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":[{\"field\":\"id\",\"operator\":\"in\",\"value\":[0]}]}}");
CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/api_keys/delete"); var request = new
RestRequest(Method.POST); request.AddHeader("authorization",
"DCdIeow0xm73EwnxjPza1tdHTfZQv2eH7bTKlTPkgBHLj8aSjFzjgTE9bQUK1DidlWLrnYRhaYQ4PCIyNrNJbMUC6DOWi8ANIn1JWpMTE2neGvoDIRsKUbj6pJ1z7G
request.AddHeader("x-xdr-auth-id", "2841"); request.AddHeader("x-xdr-timestamp", "SOME_STRING_VALUE");
request.AddHeader("x-xdr-nonce", "SOME_STRING_VALUE"); request.AddHeader("x-child-tenant-id", "SOME_STRING_VALUE");
request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":
{\"filters\":[{\"field\":\"id\",\"operator\":\"in\",\"value\":[0]}]}}", ParameterType.RequestBody); IRestResponse
response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
▶ filtersarray

An array of filter fields.

REQUEST Example 1

```
{ "request_data": { "filters": [ { "field": "id", "operator": "in", "value": [ 121, 134 ] } ] } }
```

Responses

OK

Body
application/json
▼ replyobject
▶ update_countinteger

The number of API keys deleted.

RESPONSE Example 1

```
{ "reply": { "update_count": 2 } }
```

2.1.10 | Attack surface management

APIs for attack surface management

2.1.10.1 | Get External Service

post /public_api/v1/assets/get_external_service

Get service details according to the service ID. You can send up to 20 IDs.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
```

```
'https://api-yourfqdn/public_api/v1/assets/get_external_service'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"service_id_list\":[\"string\"]}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/assets/get_external_service", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/assets/get_external_service") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"service_id_list\":[\"string\"]}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "service_id_list": [ "string" ] } }); const xhr = new
XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/assets/get_external_service"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/assets/get_external_service")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"service_id_list\":[\"string\"]}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["service_id_list": ["string"]]] as [String :
Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/assets/get_external_service")! as URL,
cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields =
headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with:
request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else
{ let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/assets/get_external_service", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"service_id_list\":[\"string\"]}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/assets/get_external_service"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"service_id_list\":
[\"string\"]}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/assets/get_external_service"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"service_id_list\":[\"string\"]}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject

A dictionary containing the API request fields.

▶ service_id_listarray[string]

Represents the service ID you want to get details for.

REQUEST
```
{ "request_data": { "service_id_list": [ "example" ] } }
```
Responses

OK

Body
application/json
▼ replyobject

JSON object containing the query result.

▶ detailsarray

Service details according to the service ID.

RESPONSE
{ "reply": { "details": [{ "service_id": "example", "service_name": "example", "service_type": "example", "ip_address": ["example"], "domain": [{}], "externally_detected_providers": ["example"], "is_active": "example", "first_observed": 0, "last_observed": 0, "port": 0, "protocol": "example", "active_classifications": ["example"], "inactive_classifications": [{}], "discovery_type": "example", "business_units": ["example"], "externally_inferred_vulnerability_score": {}, "externally_inferred_cves": [{}], "details": { "serviceKey": "example", "serviceKeyType": "example", "businessUnits": [{ "name": "example" }], "providerDetails": [{ "name": "example", "firstObserved": 0, "lastObserved": 0 }], "certificates": [{ "certificate": { "issuer": "example", "issuerAlternativeNames": "example", "issuerCountry": "example", "issuerEmail": {}, "issuerLocality": "example", "issuerName": "example", "issuerOrg": "example", "formattedIssuerOrg": "example", "issuerOrgUnit": "example", "issuerState": "example", "publicKey": "example", "publicKeyAlgorithm": "example", "publicKeyRsaExponent": 0, "signatureAlgorithm": "example", "subject": "example", "subjectAlternativeNames": "example", "subjectCountry": "example", "subjectEmail": "example", "subjectLocality": "example", "subjectName": "example", "subjectOrg": "example", "subjectOrgUnit": "example", "subjectState": "example", "serialNumber": "example", "validNotBefore": 0, "validNotAfter": 0, "version": "example", "publicKeyBits": 0, "publicKeyModulus": "example", "publicKeySpki": "example", "sha1Fingerprint": "example", "sha256Fingerprint": "example", "md5Fingerprint": "example" }, "activityStatus": "example", "lastObserved": 0, "firstObserved": 0 }], "domains": [{}], "ips": [{ "ip": "example", "protocol": "example", "provider": "example", "geolocation": { "latitude": 0, "longitude": 0, "countryCode": "example", "city": "example", "regionCode": "example", "timeZone": {} }, "activityStatus": "example", "lastObserved": 0, "firstObserved": 0 }], "classifications": [{ "name": "example", "activityStatus": "example", "values": [{ "jsonValue": "example", "firstObserved": 0, "lastObserved": 0 }], "firstObserved": 0, "lastObserved": 0 }], "tlsVersions": [{ "tlsVersion": "example", "cipherSuite": "example", "firstObserved": 0, "lastObserved": 0, "activityStatus": "example" }], "inferredCvesObserved": [{}], "enrichedObservationSource": "example", "ip_ranges": {} } }] } }

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.10.2 | Get All Services

post /public_api/v1/assets/get_external_services

Get a complete or filtered list of all your external services.

The maximum result limit is 500.

Request headers
▼ authorization String required

api-key

Example: `authorization_example`
▼ x-xdr-auth-id String required

api-key-id

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/assets/get_external_services'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
{\"field\":\"active_classifications\",\"operator\":\"contains\",\"value\":\"string\"},\"vulnerability_test_results\":true,\"sea
{\"keyword\":\"string\",\"field\":\"string\"}}}" headers = { 'authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/assets/get_external_services", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/assets/get_external_services") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":
{\"field\":\"active_classifications\",\"operator\":\"contains\",\"value\":\"string\"},\"vulnerability_test_results\":true,\"sea
{\"keyword\":\"string\",\"field\":\"string\"}}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": { "field": "active_classifications", "operator":
"contains", "value": "string" }, "vulnerability_test_results": true, "search_from": 0, "search_to": 5000, "sort": {
"keyword": "string", "field": "string" } } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/assets/get_external_services"); xhr.setRequestHeader("authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/assets/get_external_services")
.header("authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
{\"field\":\"active_classifications\",\"operator\":\"contains\",\"value\":\"string\"},\"vulnerability_test_results\":true,\"sea
{\"keyword\":\"string\",\"field\":\"string\"}}}") .asString();
import Foundation let headers = [ "authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "filters": [ "field":
"active_classifications", "operator": "contains", "value": "string" ], "vulnerability_test_results": true,
"search_from": 0, "search_to": 5000, "sort": [ "keyword": "string", "field": "string" ] ]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/assets/get_external_services")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/assets/get_external_services", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
{\"field\":\"active_classifications\",\"operator\":\"contains\",\"value\":\"string\"},\"vulnerability_test_results\":true,\"sea
{\"keyword\":\"string\",\"field\":\"string\"}}}", CURLOPT_HTTPHEADER => [ "authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/assets/get_external_services"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
{\"field\":\"active_classifications\",\"operator\":\"contains\",\"value\":\"string\"},\"vulnerability_test_results\":true,\"sea
{\"keyword\":\"string\",\"field\":\"string\"}}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/assets/get_external_services"); var request = new
RestRequest(Method.POST); request.AddHeader("authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":
{\"field\":\"active_classifications\",\"operator\":\"contains\",\"value\":\"string\"},\"vulnerability_test_results\":true,\"sea
{\"keyword\":\"string\",\"field\":\"string\"}}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
▶ filtersobject

An array of filter fields.

▶ vulnerability_test_resultsboolean (Enum)

Use this field with the value `true` to get vulnerability test results for the last 14 days for each service. Using this field will slow down the endpoint.

▶ search_frominteger

An integer representing the start offset index of results.

▶ search_tointeger

An integer representing the start offset index of results. Use this field to specify the number of results on a page when using page token pagination.

▶ sortobject

Identifies the sort order for the result set.

REQUEST Example with page token

```
{ "request_data": { "filters": [ { "field": "string", "operator": "string", "value": "string" } ], "use_page_token":
true } }
{ "request_data": { "filters": [ { "field": "discovery_type", "operator": "in", "value": [ "colocated_on_ip",
"directly_discovery" ] }, { "field": "service_name", "operator": "contains", "value": "apache" } ], "search_from": 0,
"search_to": 500 } }
```

Responses

OK

Body
application/json
▼ replyobject
▶ total_countinteger
▶ result_countinteger
▶ external_servicesarray
RESPONSE Example 1

```
{ "reply": { "total_count": 0, "result_count": 0, "external_services": [ { "service_id": "string", "service_name":
"string", "service_type": "string", "ip_address": [ "string" ], "domain": [ "string" ],
"externally_detected_providers": [ "string" ], "is_active": "string", "first_observed": 0, "last_observed": 0, "port":
0, "protocol": "string", "active_classifications": [ "string" ], "inactive_classifications": [ "string" ],
"discovery_type": "string", "business_units": [ "string" ], "externally_inferred_vulnerability_score": "null",
"externally_inferred_cves": [ {} ], "tls_versions": [ {} ], "inferred_cves_observed": [ {} ],
"cloud_management_status": null } ] } }
{ "reply": { "total_count": 2, "result_count": 2, "external_services": [ { "service_id": "<service_id>",
"service_name": "Server", "service_type": "AServer", "ip_address": [ "<ip_address>" ], "domain": [],
"externally_detected_providers": [ "On Prem" ], "is_active": "Active", "first_observed": 1647152340000,
"last_observed": 1649499420000, "port": 8009, "protocol": "TCP", "active_classifications": [ "AServer" ],
"inactive_classifications": [], "discovery_type": "DirectlyDiscovered", "business_units": [ [ { "creation_time":
1684197662636, "family": "business_units", "family_alias": "BU", "id": "BU:<id>", "name": "Business Unit 1",
"parent_id": null, "update_time": 1684197662636 } ] ], "externally_inferred_vulnerability_score": "null",
"externally_inferred_cves": [], "tls_versions": [], "inferred_cves_observed": [] }, { "service_id": "<service_id>",
```

"service_name": "HTTP Server", "service_type": "HttpServer", "ip_address": "ip_address", "domain": ["email.test.org"], "externally_detected_providers": ["Demo"], "is_active": "Inactive", "first_observed": 1647087420000, "last_observed": 1647087420000, "port": 80, "protocol": "TCP", "active_classifications": [], "inactive_classifications": ["HttpServer", "NginxWebServer", "ServerSoftware"], "discovery_type": "ColocatedOnIp", "business_units": ["Test - Import-Export"], "externally_inferred_vulnerability_score": null, "externally_inferred_cves": [], "cloud_management_status": null }] } }

Bad Request. Got invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.10.3 | Get vulnerability tests

post /public_api/v1/assets/get_vulnerability_tests

Get a complete or filtered list of vulnerability tests. Results include details about each test, including the number of services confirmed vulnerable.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/assets/get_vulnerability_tests'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"filters\":
[{\"field\":\"name\",\"operator\":\"string\",\"value\":\"string\"}]}}" headers = { 'Authorization':
"SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/assets/get_vulnerability_tests", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/assets/get_vulnerability_tests") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"filters\":[{\"field\":\"name\",\"operator\":\"string\",\"value\":\"string\"}]}}"
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "filters": [ { "field": "name", "operator": "string", "value":
"string" } ] } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/assets/get_vulnerability_tests"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/assets/get_vulnerability_tests")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"filters\":
[{\"field\":\"name\",\"operator\":\"string\",\"value\":\"string\"}]}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["filters": [ [ "field": "name", "operator":
"string", "value": "string" ] ]]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters,
options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/assets/get_vulnerability_tests")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/assets/get_vulnerability_tests", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"filters\":
[{\"field\":\"name\",\"operator\":\"string\",\"value\":\"string\"}]}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/assets/get_vulnerability_tests"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"filters\":
[{\"field\":\"name\",\"operator\":\"string\",\"value\":\"string\"}]}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/assets/get_vulnerability_tests"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"filters\":[{\"field\":\"name\",\"operator\":\"string\",\"value\":\"string\"}]}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

▶ filtersarray

An array of filter fields.

REQUEST Example

```
{ "request_data": { "filters": [ { "field": "name", "operator": "contains", "value": "apache" } ] } }
```

Responses

OK

Body

application/json

▼ replyobject

▶ total_countinteger

▶ result_countinteger

▶ vulnerability_testsarray

RESPONSE Example

{ "reply": { "total_count": 1, "result_count": 1, "vulnerability_tests": [{ "id": "69527826-e1c5-42d8-b8d8-2c2005b75cbe", "name": "Apache Solr DataImportHandler Code Injection Vulnerability", "vulnerability_ids": ["CVE-2019-0193"], "description": "Apache Solr, a popular open-source search platform built on Apache Lucene, is affected by a remote code execution vulnerability. Solr's DataImportHandler (DIH), an optional module widely used to import data from databases and other sources, allows the entire DIH configuration to come from a request's \"dataConfig\" parameter. The debug mode of the DIH admin screen uses this feature for convenient debugging and development of a DIH configuration. However, since a DIH configuration can contain scripts, this parameter poses a security risk. The affected products can potentially be exposed to the public internet, making them vulnerable to exploitation.\n", "status": "DISABLED", "vendor_names": ["apache"], "affected_software": [{ "NAME": "cpe:2.3:a:apache:solr:*:*:*:*:*:*:*:*", "VERSION_START_INCLUDING": null, "VERSION_START_EXCLUDING": null, "VERSION_END_INCLUDING": null, "VERSION_END_EXCLUDING": "8.2.0", "VENDOR": "apache", "PRODUCT": "solr", "VERSION": "*" }], "severity_score": 7.2, "cwe_ids": ["CWE-94"], "epss_score": 0.9605, "references": ["https://issues.apache.org/jira/browse/SOLR-13669"], "remediation_guidance": "Exploiting this vulnerability may lead to remote code execution, which could compromise the security and integrity of the affected system. To address this issue, follow these steps:\n\n1. Upgrade to Apache Solr 8.2.0 or later, which is secure by default.\n2. Alternatively, edit the solrconfig.xml to configure all DataImportHandler usages with an \"invariants\" section listing the \"dataConfig\" parameter set to an empty string.\n3. Ensure your network settings are configured so that only trusted traffic communicates with Solr, especially to the DIH request handler. This is a best practice for all Solr installations.\n\nBy implementing these fixes and mitigations, you can protect your Apache Solr installation from the remote code execution vulnerability described in CVE-2019-0193.\n", "first_published": 1699326060000, "created": 1711058940000, "count_vulnerable_services": null }] } }

2.1.10.4 | Bulk Update Vulnerability Tests

post /public_api/v1/assets/bulk_update_vulnerability_tests

Enable or disable vulnerability tests.

To view vulnerability test results, use the Get All Services or Get Service Details endpoints.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"test_names\":[\"string\"],\"status\":\"Enabled\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/assets/bulk_update_vulnerability_tests", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"test_names\":[\"string\"],\"status\":\"Enabled\"}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "test_names": [ "string" ], "status": "Enabled" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{\"test_names\":[\"string\"],\"status\":\"Enabled\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": [ "test_names": ["string"], "status": "Enabled" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } })
dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"test_names\":[\"string\"],\"status\":\"Enabled\"}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests"); struct curl_slist *headers = NULL;
headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-
auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json");
curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"test_names\":[\"string\"],\"status\":\"Enabled\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/assets/bulk_update_vulnerability_tests"); var request
= new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-
auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json");
request.AddParameter("application/json", "{\"request_data\":{\"test_names\":[\"string\"],\"status\":\"Enabled\"}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

▶ test_namesarray[string]

Names of tests, for example ["test1", "test2", "test3"]

▶ statusstring (Enum)

REQUEST Example 1

```
{ "request_data": { "test_names": [ "Apache Solr DataImportHandler Code Injection Vulnerability" ], "status":
"Enabled" } }
```

Responses

OK

2.1.11 | XQL user datasets

APIs for managing XQL user datasets.

2.1.11.1 | Define an XQL user dataset

post /public_api/v1/dataset/define_dataset

Define an XQL user dataset based on an existing BigQuery table created by the user.

Note: BigQuery table must be an existing table under public_access_user.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

These APIs are only applicable from within the XSIAM Notebook environment.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/dataset/define_dataset'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"table_name\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/dataset/define_dataset", payload, headers)
res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/dataset/define_dataset") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"table_name\":\"string\"}}" response = http.request(request) puts
response.read_body
```

```
const data = JSON.stringify({ "request_data": { "table_name": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/dataset/define_dataset");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/dataset/define_dataset")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"table_name\":\"string\"}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["table_name": "string"]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/dataset/define_dataset")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/dataset/define_dataset", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"table_name\":\"string\"}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/dataset/define_dataset"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"table_name\":\"string\"}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/dataset/define_dataset"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"table_name\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired
▶ table_namestring

An existing BigQuery table name that was created by the user.

REQUEST
```
{ "request_data": { "table_name": "example" } }
```
Responses

OK

2.1.11.2 | Get created XQL user datasets

post /public_api/v1/dataset/get_created_datasets

Retrieve a list of all XQL user datasets created using the Cortex SDK.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

These APIs are only applicable from within the XSIAM Notebook environment.

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/dataset/get_created_datasets'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/dataset/get_created_datasets", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/dataset/get_created_datasets") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({}); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/dataset/get_created_datasets"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/dataset/get_created_datasets") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = [] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/dataset/get_created_datasets")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/dataset/get_created_datasets", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/dataset/get_created_datasets"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/dataset/get_created_datasets"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
Object
application/json
REQUEST

```
{}
```

Responses

OK

Body
application/json
▼ datasetsarray[string]

A list of created datasets.

RESPONSE

```
{ "datasets": [ "example" ] }
```

2.1.11.3 | Delete an XQL user dataset

post /public_api/v1/dataset/delete_dataset

Delete an XQL user dataset that was created by the Cortex SDK.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

These APIs are only applicable from within the XSIAM Notebook environment.

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/dataset/delete_dataset'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"dataset_name\":\"string\",\"delete_underlying_bq_table\":false}}" headers = { 'Authorization': "SOME_STRING_VALUE",
'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/dataset/delete_dataset", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/dataset/delete_dataset") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"dataset_name\":\"string\",\"delete_underlying_bq_table\":false}}" response =
http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "dataset_name": "string", "delete_underlying_bq_table": false } });
const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () {
if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/dataset/delete_dataset"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/dataset/delete_dataset")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"dataset_name\":\"string\",\"delete_underlying_bq_table\":false}}")
.asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "dataset_name": "string",
"delete_underlying_bq_table": false ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject:
parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-
yourfqdn/public_api/v1/dataset/delete_dataset")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0)
request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session =
URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response,
error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/dataset/delete_dataset", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"dataset_name\":\"string\",\"delete_underlying_bq_table\":false}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/dataset/delete_dataset"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
```

```
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"dataset_name\":\"string\",\"delete_underlying_bq_table\":false}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/dataset/delete_dataset"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{\"dataset_name\":\"string\",\"delete_underlying_bq_table\":false}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ dataset_namestring

The dataset name to be deleted.

▶ delete_underlying_bq_tableboolean

Define whether or not to delete the BigQuery table related to the dataset.

REQUEST
```
{ "request_data": { "dataset_name": "example", "delete_underlying_bq_table": false } }
```
Responses

OK

Body
application/json
RESPONSE
```
{}
```

2.1.12 | Dataset Management

APIs for managing datasets

2.1.12.1 | Add Dataset

post /public_api/v1/xql/add_dataset

Add a dataset of type `lookup` with the specified name and schema.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/add_dataset'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"dataset_name\":\"string\",\"dataset_type\":\"string\",\"dataset_schema\":{\"property1\":\"text\",\"property2\":\"text\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/xql/add_dataset", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/xql/add_dataset") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":
```

```
{\"dataset_name\":\"string\",\"dataset_type\":\"string\",\"dataset_schema\":
{\"property1\":\"text\",\"property2\":\"text\"}}}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "dataset_name": "string", "dataset_type": "string", "dataset_schema":
{ "property1": "text", "property2": "text" } } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/add_dataset");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/add_dataset")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"dataset_name\":\"string\",\"dataset_type\":\"string\",\"dataset_schema\":
{\"property1\":\"text\",\"property2\":\"text\"}}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "dataset_name": "string", "dataset_type":
"string", "dataset_schema": [ "property1": "text", "property2": "text" ] ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/add_dataset")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/add_dataset", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"dataset_name\":\"string\",\"dataset_type\":\"string\",\"dataset_schema\":
{\"property1\":\"text\",\"property2\":\"text\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/add_dataset"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"dataset_name\":\"string\",\"dataset_type\":\"string\",\"dataset_schema\":
{\"property1\":\"text\",\"property2\":\"text\"}}}"); CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/add_dataset"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"dataset_name\":\"string\",\"dataset_type\":\"string\",\"dataset_schema\":
{\"property1\":\"text\",\"property2\":\"text\"}}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ dataset_namestring

The designated name of the dataset.

▶ dataset_typestring

Dataset type. Currently only `lookup` is supported.

▶ dataset_schemaobject

The schema of the dataset in a comma-separated list of JSON pairs where the key is the field name and the value is the field type.

REQUEST Example 1

```
{ "request_data": { "dataset_name": "users", "dataset_schema": { "uid": "text", "username": "text", "zipcode":
"number", "salary": "number", "is_admin": "bool", "birthday": "datetime" }, "dataset_type": "lookup" } }
```

Responses

OK

Body
application/json
▼ dataset_namestring

Name of the dataset added.

RESPONSE
```
{ "dataset_name": "example" }
```

2.1.12.2 | Delete a dataset

post /public_api/v2/xql/delete_dataset

Delete a dataset with the specified name. The following dataset types can be deleted: Lookup, Raw, User, Snapshot, and Correlation. You can only delete a dataset with dependencies by setting `force` to TRUE.

Note: The System dataset and other protected datasets cannot be deleted.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v2/xql/delete_dataset'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"dataset_name\":\"string\",\"force\":true}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id':
"SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v2/xql/delete_dataset",
payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v2/xql/delete_dataset")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":
{\"dataset_name\":\"string\",\"force\":true}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "dataset_name": "string", "force": true } }); const xhr = new
XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if
(this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v2/xql/delete_dataset"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v2/xql/delete_dataset")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"dataset_name\":\"string\",\"force\":true}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "dataset_name": "string", "force": true ]] as
[String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request =
NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v2/xql/delete_dataset")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v2/xql/delete_dataset", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS
=> 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{\"dataset_name\":\"string\",\"force\":true}}", CURLOPT_HTTPHEADER => [
"Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]);
$response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; }
else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v2/xql/delete_dataset"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
```

```
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"dataset_name\":\"string\",\"force\":true}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v2/xql/delete_dataset"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"dataset_name\":\"string\",\"force\":true}}", ParameterType.RequestBody); IRestResponse response
= client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

▶ dataset_namestring

The name of the dataset to be deleted.

▶ forceboolean

Warning: Setting this to `True` forces deletion even when there are dependencies.

REQUEST Example 1

```
{ "request_data": { "dataset_name": "users", "force": "yes" } }
```

Responses

OK

2.1.12.3 | Get all datasets

post /public_api/v1/xql/get_datasets

Retrieve a list of all the datasets and their properties.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers

▼ Authorization String required

{api_key}

Example: `authorization_example`

▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`

CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/get_datasets'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/xql/get_datasets", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/xql/get_datasets")
http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request =
Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE'
request["content-type"] = 'application/json' request.body = "{\"request_data\":{}}" response = http.request(request)
puts response.read_body
const data = JSON.stringify({ "request_data": {} }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/get_datasets");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/get_datasets")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": []] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/get_datasets")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval:
10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let
session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data,
response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse
print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/get_datasets", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS =>
10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST",
CURLOPT_POSTFIELDS => "{\"request_data\":{}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-
type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/get_datasets"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{}}"); CURLcode ret =
curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/get_datasets"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
REQUEST
`{ "request_data": {} }`
Responses

OK

Body
application/json
▼ replyarray
[
▶ dataset_namestring

Dataset name.

▶ typestring

Dataset type. Can be one of the following: `System`, `Lookup`, `Raw`, `User`, `Snapshot`, `Correlation`, `System Audit`.

▶ log_update_typestring

Log update type. Can be one of the following: `Logs` (event logs are updated continuously), `State` (the current state is updated periodically).

▶ last_updatedinteger

Integer in timestamp epoch milliseconds. When the data in the dataset was last updated.

▶ total_days_storedinteger

Number of dats the data is stored in the tenant, which is comprised of `hot_range` + `cold_range`.

▶ hot_rangeobject

The time period of the hot storage from the start date to the end date.

▶ cold_rangeobject

The time period of the cold storage from the start date to the end date.

▶ total_size_storedinteger

Actual size of the data (in bytes) that is stored in the tenant. This number is dependent on the events stored in the hot storage. For the xdr_data dataset, where the first 31 days of storage are included with your license, the first 31 days are not included in the `total_size_stored` number.

▶ average_daily_sizeinteger

Average daily amount stored (in bytes) in the tenant. This number is dependent on the events stored in the hot storage.

▶ total_eventsinteger

Number of total events/logs that are stored in the tenant. This number is dependent on the events stored in the hot storage.

▶ average_event_sizeinteger

Average size (in bytes) of a single event in the dataset (`total_size_stored` divided by the `total_events`). This number is dependent on the events stored in the hot storage.

▶ ttlinteger

Time to live. Defines when lookup entries expire and are removed automatically from the lookup dataset.

▶ default_query_targetboolean

whether the dataset is configured to use as your default query target in XQL Search, so when you write your queries you do not need to define a dataset. Can be one of the following: `True`, `False`.

]
RESPONSE Example 1

```
{ "reply": [ { "Dataset Name": "xdr_data", "Type": "SYSTEM", "Log Update Type": "LOGS", "Last Updated": null, "Total Days Stored": null, "Hot Range": { "from": 1715299200000, "to": 1716595200000 }, "Cold Range": {}, "Total Size Stored": null, "Average Daily Size": null, "Total Events": null, "Average Event Size": null, "TTL": null, "Default Query Target": "FALSE" }, { "Dataset Name": "host_inventory", "Type": "SYSTEM", "Log Update Type": "LOGS", "Last Updated": null, "Total Days Stored": null, "Hot Range": {}, "Cold Range": {}, "Total Size Stored": null, "Average Daily Size": null, "Total Events": null, "Average Event Size": null, "TTL": null, "Default Query Target": "FALSE" }, { "Dataset Name": "host_users_to_groups", "Type": "SYSTEM", "Log Update Type": "LOGS", "Last Updated": null, "Total Days Stored": null, "Hot Range": {}, "Cold Range": {}, "Total Size Stored": null, "Average Daily Size": null, "Total Events": null, "Average Event Size": null, "TTL": null, "Default Query Target": "FALSE" } ] }
```

2.1.13 | Lookup Datasets

APIs for lookup datasets

2.1.13.1 | Add or update data in a lookup dataset

post /public_api/v1/xql/lookups/add_data

Add or update data in a lookup dataset.

When updating data, any field not specified in the `data` field, but specified on at least one of the rows, will be set to `None`.

The `/public_api/xql/lookups/add_data/` endpoint does not support concurrent edits. Sending concurrent calls to this endpoint can cause data to be unintentionally overwritten or deleted. To allow sufficient time for each API call to complete its operation before initiating another one, assume that 1000 entries can be added per API every 10 seconds.

**Note: **

- The maximum size of a lookup dataset is 50 MB. Attempting to exceed this limit will fail.
- Requests time out after three minutes.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/lookups/add_data'
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"dataset_name\":\"string\",\"key_fields\":[\"string\"],\"data\":
{\"property1\":\"string\",\"property2\":\"string\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-
id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST",
"/public_api/v1/xql/lookups/add_data", payload, headers) res = conn.getresponse() data = res.read()
print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/xql/lookups/add_data") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"dataset_name\":\"string\",\"key_fields\":[\"string\"],\"data\":
{\"property1\":\"string\",\"property2\":\"string\"}}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "dataset_name": "string", "key_fields": [ "string" ], "data": {
"property1": "string", "property2": "string" } } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/lookups/add_data");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/lookups/add_data")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"dataset_name\":\"string\",\"key_fields\":[\"string\"],\"data\":
{\"property1\":\"string\",\"property2\":\"string\"}}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "dataset_name": "string", "key_fields":
["string"], "data": [ "property1": "string", "property2": "string" ] ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/lookups/add_data")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/lookups/add_data", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"dataset_name\":\"string\",\"key_fields\":[\"string\"],\"data\":
{\"property1\":\"string\",\"property2\":\"string\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/lookups/add_data"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"dataset_name\":\"string\",\"key_fields\":[\"string\"],\"data\":
{\"property1\":\"string\",\"property2\":\"string\"}}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/lookups/add_data"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"dataset_name\":\"string\",\"key_fields\":[\"string\"],\"data\":
{\"property1\":\"string\",\"property2\":\"string\"}}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```
Body parameters

application/json

▼ request_dataobjectrequired

▶ dataset_namestring

Unique dataset name

▶ key_fieldsarray[string]

The fields used to identify existing records. If there is not an exact match to the key_fields specified, a new row is created. When you specify `key_fields`, these fields are mandatory in data entries. When `key_fields` are not specified, existing data entries are not updated, and new entries are added with the specified data.

▶ dataobject

Key-value pairs of data entries.

REQUEST Example 1
```
{ "request_data": { "dataset_name": "users", "key_fields": [ "uid", "username" ], "data": [ { "uid": "123abc", "username": "john", "zipcode": 58672, "salary": 5.1, "is_admin": false, "birthday": "31-05-1982T10:22:45Z" }, { "uid": "124abc", "username": "jane", "zipcode": 58642, "salary": 5000000, "is_admin": true, "birthday": "31-03-1982T10:22:45Z" } ] } }
```
Responses

OK

Body
application/json
▼ addedinteger
▼ updatedinteger
▼ skippedinteger
RESPONSE
```
{ "added": 0, "updated": 0, "skipped": 0 }
```

2.1.13.2 | Remove data from a lookup dataset

post /public_api/v1/xql/lookups/remove_data

Remove data from a dataset based on the specified parameters. If any one of the filter sets are not found, the API does not delete any data.

The `/public_api/xql/lookups/remove_data/` endpoint does not support concurrent edits. Sending concurrent calls to this endpoint can cause data to be unintentionally overwritten or deleted. To allow sufficient time for each API call to complete its operation before initiating another one, assume that 1000 entries can be added per API every 10 seconds.

Note:

- All lookup entries matching any of the filter blocks are deleted. To match a filter block, a lookup entry must match all the specified fields as if there were an `AND` operator between them.
- Requests time out after three minutes.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/lookups/remove_data'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{\"dataset_name\":\"string\",\"filters\":{\"property1\":\"string\",\"property2\":\"string\"}}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" } conn.request("POST", "/public_api/v1/xql/lookups/remove_data", payload, headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/xql/lookups/remove_data") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":{\"dataset_name\":\"string\",\"filters\":{\"property1\":\"string\",\"property2\":\"string\"}}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "dataset_name": "string", "filters": { "property1": "string", "property2": "string" } } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/lookups/remove_data");
```

```
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/lookups/remove_data")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"dataset_name\":\"string\",\"filters\":
{\"property1\":\"string\",\"property2\":\"string\"}}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "dataset_name": "string", "filters": [
"property1": "string", "property2": "string" ] ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/lookups/remove_data")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/lookups/remove_data", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"dataset_name\":\"string\",\"filters\":
{\"property1\":\"string\",\"property2\":\"string\"}}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE",
"content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err =
curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/lookups/remove_data"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"dataset_name\":\"string\",\"filters\":{\"property1\":\"string\",\"property2\":\"string\"}}}"); CURLcode ret =
curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/lookups/remove_data"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"dataset_name\":\"string\",\"filters\":{\"property1\":\"string\",\"property2\":\"string\"}}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ dataset_namestring

The name of the dataset to delete.

▶ filtersobject

Key-value pairs of fields to query in datasets. A lookup entry must match all the specified fields as if there were an `AND` operator between them. You can use one or more fields, up to the number of fields in the schema.

REQUEST Example 1

```
{ "request_data": { "dataset_name": "users", "filters": [ { "uid": "123", "username": "john" }, { "uid": "124",
"zipcode": 58672 } ] } }
```

Responses

OK

Body
application/json
▼ deletedinteger

Number of entries deleted successfully.

RESPONSE

```
{ "deleted": 0 }
```

2.1.13.3 | Get data from a lookup dataset

post /public_api/v1/xql/lookups/get_data

Get data from a lookup dataset according to the specified filter fields. All lookup entries matching any of the filter blocks are returned. To match a filter block, a lookup entry must match all the specified fields as if there were an `AND` operator between them. If no filters are specified, return all lookup entries.

Note:

- The maximum number of entries returned is 10,000.
- Requests time out after three minutes.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management
Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/xql/lookups/get_data'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"dataset_name\":\"string\",\"filters\":[{\"property1\":\"string\",\"property2\":\"string\"}],\"limit\":0}}" headers
= { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/xql/lookups/get_data", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/xql/lookups/get_data") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":{\"dataset_name\":\"string\",\"filters\":
[{\"property1\":\"string\",\"property2\":\"string\"}],\"limit\":0}}" response = http.request(request) puts
response.read_body
const data = JSON.stringify({ "request_data": { "dataset_name": "string", "filters": [ { "property1": "string",
"property2": "string" } ], "limit": 0 } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/xql/lookups/get_data");
xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/xql/lookups/get_data")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"dataset_name\":\"string\",\"filters\":
[{\"property1\":\"string\",\"property2\":\"string\"}],\"limit\":0}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "dataset_name": "string", "filters": [ [
"property1": "string", "property2": "string" ] ], "limit": 0 ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/xql/lookups/get_data")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/xql/lookups/get_data", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"dataset_name\":\"string\",\"filters\":
[{\"property1\":\"string\",\"property2\":\"string\"}],\"limit\":0}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/xql/lookups/get_data"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
```

```
{\"dataset_name\":\"string\",\"filters\":[{\"property1\":\"string\",\"property2\":\"string\"}],\"limit\":0}}");
CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/xql/lookups/get_data"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"dataset_name\":\"string\",\"filters\":
[{\"property1\":\"string\",\"property2\":\"string\"}],\"limit\":0}}", ParameterType.RequestBody); IRestResponse
response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobjectrequired
▶ dataset_namestring

Name of the dataset to query.

▶ filtersarray

Key-value pairs of fields to query in a dataset. A lookup entry must match all the specified fields as if there were an `AND` operator between them. You can use one or more fields, up to the number of fields in the schema.

▶ limitinteger

The maximum number of results to return. If this is not specified, return all lookup entries that match the filter criteria.

REQUEST Example 1

```
{ "request_data": { "dataset_name": "users", "filters": [ { "uid": "123", "username": "john" }, { "department": "dev",
"zipcode": "58674" } ], "limit": 20 } }
```
Responses

OK

Body
application/json
▼ dataobject
▶ Additional propertiesstring
▼ filter_countinteger

Number of entries that match the filter.

▼ total_countinteger

Total number of entries.

RESPONSE Example 1

```
{ "reply": { "data": [ { "uid": "uid5", "salary": 5.1, "zipcode": 70005, "birthday": 386418165000, "is_admin": true,
"username": "username5", "_insert_time": 1718807765000, "_update_time": 1718807765000, "_collector_name": "Console",
"_collector_type": "Console" }, { "uid": "uid6", "salary": 6.1, "zipcode": 70006, "birthday": 386418165000,
"is_admin": true, "username": "username6", "_insert_time": 1718807765000, "_update_time": 1718807765000,
"_collector_name": "Console", "_collector_type": "Console" } ], "filter count": 2, "total count": 10 } }
```

2.1.14 | Authentication settings

APIs for authentication settings, such as IdP and SSO

2.1.14.1 | Create authentication settings for IdP SSO or metadata URL

post /public_api/v1/authentication-settings/create

Create authentication settings for IdP SSO or metadata URL. You must include either the `metadata_url` field or all of the following fields: `idp_sso_url`, `idp_issuer`, and `idp_certificate`.

You must have Instance Administrator permissions to run this endpoint.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}
Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/authentication-settings/create'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"domain\":\"string\",\"mappings\":
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type':
"application/json" } conn.request("POST", "/public_api/v1/authentication-settings/create", payload, headers) res =
conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/authentication-
settings/create") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode =
OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE'
request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "
{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"domain\":\"string\",\"mappings\":
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "name": "string", "default_role": "string", "is_account_role": false,
"domain": "string", "mappings": { "email": "string", "firstname": "string", "lastname": "string", "group_name":
"string" }, "advanced_settings": { "relay_state": "string", "idp_single_logout_url": "string",
"service_provider_public_cert": "string", "service_provider_private_key": "string", "authn_context_enabled": false,
"force_authn": false }, "idp_sso_url": "string", "idp_certificate": "string", "idp_issuer": "string", "metadata_url":
"string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange",
function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST",
"https://api-yourfqdn/public_api/v1/authentication-settings/create"); xhr.setRequestHeader("Authorization",
"SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type",
"application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/authentication-settings/create")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"domain\":\"string\",\"mappings\":
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
.asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "name": "string", "default_role": "string",
"is_account_role": false, "domain": "string", "mappings": [ "email": "string", "firstname": "string", "lastname":
"string", "group_name": "string" ], "advanced_settings": [ "relay_state": "string", "idp_single_logout_url": "string",
"service_provider_public_cert": "string", "service_provider_private_key": "string", "authn_context_enabled": false,
"force_authn": false ], "idp_sso_url": "string", "idp_certificate": "string", "idp_issuer": "string", "metadata_url":
"string" ]] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let
request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/authentication-settings/create")!
as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST"
request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask
= session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error !=
nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } })
dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/authentication-settings/create", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"domain\":\"string\",\"mappings\":
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/authentication-settings/create"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"domain\":\"string\",\"mappings\":
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/authentication-settings/create"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"domain\":\"string\",\"mappings\":
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobjectrequired
▶ namestring

The name of the SSO integration.

▶ default_rolestring

The default role automatically assigned to every user who authenticates to Cortex using SAML. This is an inherited role and is not the same as a direct role assigned to the user.

If a role with the same name exists on both Cortex Gateway and the tenant, the role will mapped to the role from the tenant. If you want to use specifically the role from Cortex Gateway, use the `is_account_role` parameter set to `true`.

▶ is_account_roleboolean

Whether the role was created in Cortex Gateway or in the tenant. When the value is `true`, the role was created in Cortex Gateway.

▶ domainstring

When configuring the first SSO, this parameter should be included as empty because it is the default SSO and has a fixed, read-only value. For additional SSOs, specify this IdP with an email domain (user@). When logging in, users are redirected to the IdP associated with their email domain or to the default IdP if no association exists.

▶ mappingsobjectrequired

These IdP attribute mappings are dependent on your organizationâ€ s IdP.

▶ advanced_settingsobject

The advanced settings are optional to configure and some are specific for a particular IdP.

▶ idp_sso_urlstring

The login URL of your IdP and should be copied from your SAML integration configuration on the IdP. For example:

- Okta: https://cortex-test.okta.com/app/cortex-test/eacbt6b2jj08CasdUQ7sdf15d7/sso/SAML
- Microsoft Azure: https://login.microsoftonline.com/6a5a9780-96a4-41ef-bf45-0535d8a70025/saml2

▶ idp_certificatestring

The Idp's public X.509 digital certificate in PEM format for verification, which is copied from your organization's IdP.

▶ idp_issuerstring

The unique identifier of the IdP issuing SAML assertions, which is copied from your organization's IdP.

▶ metadata_urlstring

The metadata URL provides information about hte IdP's capabilities, endpoints, keys, and more. For example:

- Okta: https://cortex-test.okta.com/app/exkbuuzw77Bh04V6M6b8/sso/saml/metadata
- Microsoft Azure: https://login.microsoftonline.com/6a5a9780-96a4-41ef-bf45-0535d8a70025/saml2/metadata

REQUEST Example 1

```
{ "request_data": { "name": "IdP configuration", "default_role": "Analyst", "domain": "my-test-domain.com",
"mappings": { "email": "user@company.com", "firstname": "John", "lastname": "Smith", "group_name": "analysts" },
"idp_sso_url": "https://cortex-test.okta.com/app/cortex-test/xxxxxxx/sso/SAML", "idp_certificate":
"MY_CERTIFICATE_FROM_OKTA", "idp_issuer": "https://cortex-test.okta.com/idp", "advanced_settings": {},
"is_account_role": true } }
```
Responses

OK

Body
application/json
▼ replyboolean
RESPONSE Example 1
```
{ "reply": true }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.14.2 | Update authentication settings

post /public_api/v1/authentication-settings/update

Update existing authentication settings. To update the default domain, include empty value for both `current_domain_value` and `new_domain_value`.

You must have Instance Administrator permissions to run this endpoint.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST [Bash+curl]

```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/authentication-settings/update'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"current_domain_value\":\"string\",\"new_domain_val
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type':
"application/json" } conn.request("POST", "/public_api/v1/authentication-settings/update", payload, headers) res =
conn.getresponse() data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/authentication-
settings/update") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode =
OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE'
request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "
{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"current_domain_value\":\"string\",\"new_domain_val
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": { "name": "string", "default_role": "string", "is_account_role": false,
"current_domain_value": "string", "new_domain_value": "string", "mappings": { "email": "string", "firstname":
"string", "lastname": "string", "group_name": "string" }, "advanced_settings": { "relay_state": "string",
"idp_single_logout_url": "string", "service_provider_public_cert": "string", "service_provider_private_key": "string",
"authn_context_enabled": false, "force_authn": false }, "idp_sso_url": "string", "idp_certificate": "string",
"idp_issuer": "string", "metadata_url": "string" } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/authentication-
settings/update"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/authentication-settings/update")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"current_domain_value\":\"string\",\"new_domain_val
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
.asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "name": "string", "default_role": "string",
"is_account_role": false, "current_domain_value": "string", "new_domain_value": "string", "mappings": [ "email":
"string", "firstname": "string", "lastname": "string", "group_name": "string" ], "advanced_settings": [ "relay_state":
"string", "idp_single_logout_url": "string", "service_provider_public_cert": "string", "service_provider_private_key":
"string", "authn_context_enabled": false, "force_authn": false ], "idp_sso_url": "string", "idp_certificate":
"string", "idp_issuer": "string", "metadata_url": "string" ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/authentication-settings/update")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/authentication-settings/update", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"current_domain_value\":\"string\",\"new_domain_val
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/authentication-settings/update"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"current_domain_value\":\"string\",\"new_domain_val
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/authentication-settings/update"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":
{\"name\":\"string\",\"default_role\":\"string\",\"is_account_role\":false,\"current_domain_value\":\"string\",\"new_domain_val
{\"email\":\"string\",\"firstname\":\"string\",\"lastname\":\"string\",\"group_name\":\"string\"},\"advanced_settings\":
{\"relay_state\":\"string\",\"idp_single_logout_url\":\"string\",\"service_provider_public_cert\":\"string\",\"service_provider
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobjectrequired

▶ namestring

The name of the SSO integration.

▶ default_rolestring

The default role automatically assigned to every user who authenticates to Cortex using SAML. This is an inherited role and is not the same as a direct role assigned to the user.

If a role with the same name exists on both Cortex Gateway and the tenant, the role will mapped to the role from the tenant. If you want to use specifically the role from Cortex Gateway, use the `is_account_role` parameter set to `true`.

▶ is_account_roleboolean

Whether the role was created in Cortex Gateway or in the tenant. When the value is `true`, the role was created in Cortex Gateway.

▶ current_domain_valuestring

The domain whose authentication settings you want to update.

▶ new_domain_valuestring

If you want to update the domain value, include a new unique domain.

▶ mappingsobjectrequired

These IdP attribute mappings are dependent on your organization's IdP.

▶ advanced_settingsobject

The advanced settings are optional to configure and some are specific for a particular IdP.

▶ idp_sso_urlstring

The URL of your IdP's SSO, which is a fixed, read-only value based on your tenant's URL. If you are using this parameter, you must also specify: `idp_certificate` and `idp_issuer`.

▶ idp_certificatestring

The Idp's public X.509 digital certificate in PEM format for verification, which is copied from your organization's IdP.

▶ idp_issuerstring

The unique identifier of the IdP issuing SAML assertions, which is copied from your organization's IdP.

▶ metadata_urlstring

Specify your IdP SSO URL, which is a fixed, read-only value based on your tenant's URL.

REQUEST Example 1

```
{ "request_data": { "name": "IDP configuration", "default_role": "Analyst", "current_domain_value": "my-test-
domain.com", "new_domain_value": "my-test-domain.org", "mappings": { "email": "user@company.com", "firstname": "John",
"lastname": "Smith", "group_name": "analysts" }, "idp_sso_url": "https://cortex-test.okta.com/app/cortex-
test/xxxxxxx/sso/SAML", "idp_certificate": "========MY_UPDATED_TEST_CERTIFICATE_FROM_OKTA======", "idp_issuer":
"https://cortex-test.okta.com/idp", "advanced_settings": {}, "is_account_role": true } }
```

Responses

OK

Body

application/json
▼ replyboolean
RESPONSE
```
{ "reply": false }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.14.3 | Delete authentication settings by domain

post /public_api/v1/authentication-settings/delete

Delete all authentication settings for the specified domain.

**Note: ** The first configuration on the tenant is the default configuration and cannot be deleted.

You must have Instance Administrator permissions to run this endpoint.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/authentication-settings/delete'
```

```
-d ''
```

```
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"domain\":\"string\"}}" headers = { 'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE",
'content-type': "application/json" } conn.request("POST", "/public_api/v1/authentication-settings/delete", payload,
headers) res = conn.getresponse() data = res.read() print(data.decode("utf-8"))
```

```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/authentication-
settings/delete") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode =
OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE'
request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "
{\"request_data\":{\"domain\":\"string\"}}" response = http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "domain": "string" } }); const xhr = new XMLHttpRequest();
xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE)
{ console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/authentication-
settings/delete"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/authentication-settings/delete")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":{\"domain\":\"string\"}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": ["domain": "string"]] as [String : Any] let
postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url:
NSURL(string: "https://api-yourfqdn/public_api/v1/authentication-settings/delete")! as URL, cachePolicy:
.useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers
request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as
URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let
httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/authentication-settings/delete", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{\"domain\":\"string\"}}", CURLOPT_HTTPHEADER => [ "Authorization:
SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response =
curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo
$response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/authentication-settings/delete"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{\"domain\":\"string\"}}");
CURLcode ret = curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/authentication-settings/delete"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":{\"domain\":\"string\"}}", ParameterType.RequestBody); IRestResponse response =
client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
▶ domainstring

The domain whose authentication settings you want to delete.

REQUEST Example 1

```
{ "request_data": { "domain": "my-test-domain.org" } }
```

Responses

OK

Body
application/json
▼ replyboolean
RESPONSE Example 1

```
{ "reply": true }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.14.4 | Get authentication settings for all configured domains

post /public_api/v1/authentication-settings/get/settings

Get all the authentication settings for every configured domain in the tenant.

You must have Instance Administrator permissions to run this endpoint.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/authentication-settings/get/settings'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/authentication-settings/get/settings", payload, headers) res = conn.getresponse()
data = res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/authentication-
settings/get/settings") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode =
OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE'
request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "
{\"request_data\":{}}" response = http.request(request) puts response.read_body
const data = JSON.stringify({ "request_data": {} }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/authentication-
settings/get/settings"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/authentication-settings/get/settings") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{}}") .asString();
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": []] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/authentication-settings/get/settings")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/authentication-settings/get/settings", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/authentication-settings/get/settings"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{}}"); CURLcode ret = curl_easy_perform(hnd);
var client = new RestClient("https://api-yourfqdn/public_api/v1/authentication-settings/get/settings"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters
application/json
▼ request_dataobject
REQUEST

```
{ "request_data": {} }
```

Responses

OK

Body
application/json
▼ replyarray
[
▶ tenant_idstring
▶ namestring
▶ domainstring
▶ idp_enabledboolean
▶ default_roleobject
▶ is_account_roleobject
▶ idp_certificatestring
▶ idp_issuerstring
▶ idp_sso_urlstring
▶ metadata_urlstring
▶ mappingsobject
▶ advanced_settingsobject
▶ sp_entity_idstring
▶ sp_logout_urlstring
▶ sp_urlstring
]
RESPONSE Example 1

```
{ "reply": [ { "tenant_id": "9949042437653", "name": "SSO Integration", "domain": "", "idp_enabled": true, "default_role": null, "is_account_role": null, "idp_certificate": "certificate", "idp_issuer": "http://test.com", "idp_sso_url": "http://test.com/", "metadata_url": "", "mappings": { "email": "user@company.com", "firstname": "John", "group_name": "Users", "lastname": "Smith" }, "advanced_settings": { "authn_context_enabled": false, "force_authn": null, "idp_single_logout_url": "", "relay_state": "", "service_provider_private_key": "", "service_provider_public_cert": "" }, "sp_entity_id": "https://tenant.cortex.us.paloaltonetworks.com",
"sp_logout_url": "https://tenant.cortex.us.paloaltonetworks.com/idp/logout", "sp_url":
"https://tenant.cortex.us.paloaltonetworks.com/idp/saml" } ] }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.14.5 | Get IdP metadata

post /public_api/v1/authentication-settings/get/metadata

Get the metadata for all IdPs.

You must have **Instance Administrator** permissions to run this endpoint.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":{}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
```

```
conn.request("POST", "/public_api/v1/authentication-settings/get/metadata", payload, headers) res = conn.getresponse()
data = res.read() print(data.decode("utf-8"))
```
```
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] = 'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json' request.body = "{\"request_data\":{}}" response = http.request(request) puts response.read_body
```
```
const data = JSON.stringify({ "request_data": {} }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true; xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) { console.log(this.responseText); } }); xhr.open("POST", "https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE"); xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json"); xhr.send(data);
```
```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata") .header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type", "application/json") .body("{\"request_data\":{}}") .asString();
```
```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE", "content-type": "application/json" ] let parameters = ["request_data": []] as [String : Any] let postData = JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string: "https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata")! as URL, cachePolicy: .useProtocolCachePolicy, timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse = response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```
```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "", CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST => "POST", CURLOPT_POSTFIELDS => "{\"request_data\":{}}", CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id: SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo "cURL Error #:" . $err; } else { echo $response; }
```
```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL, "https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata"); struct curl_slist *headers = NULL; headers = curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd, CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":{}}"); CURLcode ret = curl_easy_perform(hnd);
```
```
var client = new RestClient("https://api-yourfqdn/public_api/v1/authentication-settings/get/metadata"); var request = new RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json", "{\"request_data\":{}}", ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```
Body parameters
application/json
▼ request_dataobject
REQUEST
```
{ "request_data": {} }
```
Responses

OK

Body
application/json
▼ replyobject
▶ sp_entity_idstring
▶ sp_logout_urlstring
▶ sp_urlstring
▶ tenant_idstring
RESPONSE Example 1
```
{ "reply": { "sp_entity_id": "https://tenant.cortex.us.paloaltonetworks.com", "sp_logout_url": "https://tenant.cortex.us.paloaltonetworks.com/idp/logout", "sp_url": "https://tenant.cortex.us.paloaltonetworks.com/idp/saml", "tenant_id": "9949042437653" } }
```

Bad Request. Got an invalid JSON.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. An issue occurred during authentication. This can indicate an incorrect key, id, or other invalid authentication parameters.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Unauthorized access. User does not have the required license type to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Forbidden access. The provided API Key does not have the required RBAC permissions to run this API.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.
RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

Internal server error. A unified status for API communication type errors.

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE
```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input
'|alter2'\"}", "err_extra": "example" }
```

2.1.15 | Syslog servers

APIs for managing syslog servers

2.1.15.1 | Create a syslog integration

post /public_api/v1/integrations/syslog/create

Create a new syslog integration.

You must have **View/Edit Alert Notification** permissions to run this endpoint.

Required license: Cortex Cloud Runtime Security or Cortex Cloud Posture Management

Request headers
▼ Authorization String required

{api_key}

Example: `authorization_example`
▼ x-xdr-auth-id String required

{api_key_id}

Example: `xXdrAuthId_example`
CLIENT REQUEST Bash+curl
```
curl -X 'POST'
-H 'Accept: application/json'
-H 'Content-Type: application/json'
-H 'Authorization: authorization_example' -H 'x-xdr-auth-id: xXdrAuthId_example'
'https://api-yourfqdn/public_api/v1/integrations/syslog/create'
-d ''
import http.client conn = http.client.HTTPSConnection("api-yourfqdn") payload = "{\"request_data\":
{\"name\":\"string\",\"address\":\"string\",\"port\":0,\"protocol\":\"TCP\",\"facility\":\"string\",\"security_info\":
{\"certificate_name\":\"string\",\"ignore_cert_errors\":true,\"certificate_content\":\"string\"}}}" headers = {
'Authorization': "SOME_STRING_VALUE", 'x-xdr-auth-id': "SOME_STRING_VALUE", 'content-type': "application/json" }
conn.request("POST", "/public_api/v1/integrations/syslog/create", payload, headers) res = conn.getresponse() data =
res.read() print(data.decode("utf-8"))
require 'uri' require 'net/http' require 'openssl' url = URI("https://api-
yourfqdn/public_api/v1/integrations/syslog/create") http = Net::HTTP.new(url.host, url.port) http.use_ssl = true
```

```
http.verify_mode = OpenSSL::SSL::VERIFY_NONE request = Net::HTTP::Post.new(url) request["Authorization"] =
'SOME_STRING_VALUE' request["x-xdr-auth-id"] = 'SOME_STRING_VALUE' request["content-type"] = 'application/json'
request.body = "{\"request_data\":
{\"name\":\"string\",\"address\":\"string\",\"port\":0,\"protocol\":\"TCP\",\"facility\":\"string\",\"security_info\":
{\"certificate_name\":\"string\",\"ignore_cert_errors\":true,\"certificate_content\":\"string\"}}}" response =
http.request(request) puts response.read_body
```

```
const data = JSON.stringify({ "request_data": { "name": "string", "address": "string", "port": 0, "protocol": "TCP",
"facility": "string", "security_info": { "certificate_name": "string", "ignore_cert_errors": true,
"certificate_content": "string" } } }); const xhr = new XMLHttpRequest(); xhr.withCredentials = true;
xhr.addEventListener("readystatechange", function () { if (this.readyState === this.DONE) {
console.log(this.responseText); } }); xhr.open("POST", "https://api-
yourfqdn/public_api/v1/integrations/syslog/create"); xhr.setRequestHeader("Authorization", "SOME_STRING_VALUE");
xhr.setRequestHeader("x-xdr-auth-id", "SOME_STRING_VALUE"); xhr.setRequestHeader("content-type", "application/json");
xhr.send(data);
```

```
HttpResponse<String> response = Unirest.post("https://api-yourfqdn/public_api/v1/integrations/syslog/create")
.header("Authorization", "SOME_STRING_VALUE") .header("x-xdr-auth-id", "SOME_STRING_VALUE") .header("content-type",
"application/json") .body("{\"request_data\":
{\"name\":\"string\",\"address\":\"string\",\"port\":0,\"protocol\":\"TCP\",\"facility\":\"string\",\"security_info\":
{\"certificate_name\":\"string\",\"ignore_cert_errors\":true,\"certificate_content\":\"string\"}}}") .asString();
```

```
import Foundation let headers = [ "Authorization": "SOME_STRING_VALUE", "x-xdr-auth-id": "SOME_STRING_VALUE",
"content-type": "application/json" ] let parameters = ["request_data": [ "name": "string", "address": "string",
"port": 0, "protocol": "TCP", "facility": "string", "security_info": [ "certificate_name": "string",
"ignore_cert_errors": true, "certificate_content": "string" ] ]] as [String : Any] let postData =
JSONSerialization.data(withJSONObject: parameters, options: []) let request = NSMutableURLRequest(url: NSURL(string:
"https://api-yourfqdn/public_api/v1/integrations/syslog/create")! as URL, cachePolicy: .useProtocolCachePolicy,
timeoutInterval: 10.0) request.httpMethod = "POST" request.allHTTPHeaderFields = headers request.httpBody = postData
as Data let session = URLSession.shared let dataTask = session.dataTask(with: request as URLRequest,
completionHandler: { (data, response, error) -> Void in if (error != nil) { print(error) } else { let httpResponse =
response as? HTTPURLResponse print(httpResponse) } }) dataTask.resume()
```

```
<?php $curl = curl_init(); curl_setopt_array($curl, [ CURLOPT_URL => "https://api-
yourfqdn/public_api/v1/integrations/syslog/create", CURLOPT_RETURNTRANSFER => true, CURLOPT_ENCODING => "",
CURLOPT_MAXREDIRS => 10, CURLOPT_TIMEOUT => 30, CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, CURLOPT_CUSTOMREQUEST
=> "POST", CURLOPT_POSTFIELDS => "{\"request_data\":
{\"name\":\"string\",\"address\":\"string\",\"port\":0,\"protocol\":\"TCP\",\"facility\":\"string\",\"security_info\":
{\"certificate_name\":\"string\",\"ignore_cert_errors\":true,\"certificate_content\":\"string\"}}}",
CURLOPT_HTTPHEADER => [ "Authorization: SOME_STRING_VALUE", "content-type: application/json", "x-xdr-auth-id:
SOME_STRING_VALUE" ], ]); $response = curl_exec($curl); $err = curl_error($curl); curl_close($curl); if ($err) { echo
"cURL Error #:" . $err; } else { echo $response; }
```

```
CURL *hnd = curl_easy_init(); curl_easy_setopt(hnd, CURLOPT_CUSTOMREQUEST, "POST"); curl_easy_setopt(hnd, CURLOPT_URL,
"https://api-yourfqdn/public_api/v1/integrations/syslog/create"); struct curl_slist *headers = NULL; headers =
curl_slist_append(headers, "Authorization: SOME_STRING_VALUE"); headers = curl_slist_append(headers, "x-xdr-auth-id:
SOME_STRING_VALUE"); headers = curl_slist_append(headers, "content-type: application/json"); curl_easy_setopt(hnd,
CURLOPT_HTTPHEADER, headers); curl_easy_setopt(hnd, CURLOPT_POSTFIELDS, "{\"request_data\":
{\"name\":\"string\",\"address\":\"string\",\"port\":0,\"protocol\":\"TCP\",\"facility\":\"string\",\"security_info\":
{\"certificate_name\":\"string\",\"ignore_cert_errors\":true,\"certificate_content\":\"string\"}}}"); CURLcode ret =
curl_easy_perform(hnd);
```

```
var client = new RestClient("https://api-yourfqdn/public_api/v1/integrations/syslog/create"); var request = new
RestRequest(Method.POST); request.AddHeader("Authorization", "SOME_STRING_VALUE"); request.AddHeader("x-xdr-auth-id",
"SOME_STRING_VALUE"); request.AddHeader("content-type", "application/json"); request.AddParameter("application/json",
"{\"request_data\":
{\"name\":\"string\",\"address\":\"string\",\"port\":0,\"protocol\":\"TCP\",\"facility\":\"string\",\"security_info\":
{\"certificate_name\":\"string\",\"ignore_cert_errors\":true,\"certificate_content\":\"string\"}}}",
ParameterType.RequestBody); IRestResponse response = client.Execute(request);
```

Body parameters

application/json

▼ request_dataobject

A dictionary containing the API request fields.

▶ namestring

Unique name for the syslog server integration.

▶ addressstring

IP address or fully qualified domain name (FQDN) of the syslog server.

▶ portinteger

The port number on which the syslog server listens for messages.

▶ protocolstring (Enum)

Select a method of communication:

- TCP: No validation is made on the connection with the syslog server. However, if an error occurred with the domain used to make the connection, the Test connection will fail.
- UDP: No error checking, error correction, or acknowledgment. No validation is done for the connection or when sending data.
- TLS: Cortex validates the syslog server certificate and uses the certificate signature and public key to encrypt the data sent over the connection.

▶ facilitystring

Choose one of the syslog standard values. The value maps to how your syslog server uses the facility field to manage messages. For details on the facility field, see RFC 5424.

▶ security_infoobject

The `security_info` parameters are necessary only when `protocol` is `TLS`.

REQUEST Example 1

```
{ "request_data": { "address": "xdr-splunk-qa.traps.company.com", "facility": "FAC_USER", "name": "Syslog_PAPI_Test_7H55R76T", "port": 5006, "protocol": "TCP", "security_info": "None" } }
{ "request_data": { "address": "xdr-splunk-qa.traps.paloaltonetworks.com", "facility": "FAC_USER", "name": "Syslog_PAPI_Test_2QYH3VGS", "port": 5002, "protocol": "TLS", "security_info": { "ignore_cert_errors": false } } }
```

Responses

OK

Body
application/json
▼ syslog_integration_idinteger
▼ namestring
RESPONSE Example 1

```
{ "syslog_integration_id": 630, "name": "Test PAPI" }
```

Bad Request

Body
application/json

The query result upon error.

▼ err_codestring

HTTP response code.

▼ err_msgstring

Error message.

Example:`"{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}"`
▼ err_extrastring

Additional information describing the error.

RESPONSE

```
{ "err_code": "example", "err_msg": "{\"line\": 1, \"column\": 19, \"message\": \"no viable alternative at input '|alter2'\"}", "err_extra": "example" }
```

Unauthorized

Body
application/json

The query result upon error.

▼ Details