API QUICK START

HYCU Data Protection as a Service for Google Cloud

December 2022

HYCU®

Legal notices

Copyright notice

Trademarks

HYCU logos, names, trademarks and/or service marks and combinations thereof are the property of HYCU or its affiliates. Other product names are the property of their respective trademark or service mark holders and are hereby acknowledged.

Amazon Web Services, AWS, Amazon EC2, Amazon S3, and Amazon Cognito are trademarks of Amazon.com, Inc. or its affiliates.

GCP™, GKE™, Google Chrome™, Google Cloud™, Google Cloud Platform™, Google Cloud Storage™, and Google Compute Engine™ are trademarks of Google LLC.

Kubernetes® is the registered trademark of The Linux Foundation in the United States and/or other countries.

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Microsoft®, Microsoft Edge™, and Windows® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Mozilla and Firefox are trademarks of the Mozilla Foundation in the U.S. and other countries.

SAP HANA® is the trademark or registered trademark of SAP SE or its affiliates in Germany and in several other countries.

Disclaimer

The details and descriptions contained in this document are believed to have been accurate and up to date at the time the document was written. The information contained in this document is subject to change without notice.

HYCU provides this material "as is" and makes no warranty of any kind, expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. HYCU shall not be liable for errors and omissions contained herein. In no event shall HYCU be liable for any direct, indirect, consequential, punitive,

special or incidental damages, including, without limitation, damages for loss and profits, loss of anticipated savings, business interruption, or loss of information arising out of the use or inability to use this document, or any action taken based on the information contained herein, even if it has been advised of the possibility of such damages, whether based on warranty, contract, or any other legal theory.

The only warranties for HYCU products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty.

Notice

This document is provided in connection with HYCU products. HYCU may have copyright, patents, patent applications, trademark, or other intellectual property rights covering the subject matter of this document.

Except as expressly provided in any written license agreement from HYCU, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property on HYCU products. Use of underlying HYCU product(s) is governed by their respective Software License and Support Terms.

Important: Please read Software License and Support Terms before using the accompanying software product(s).

HYCU
www.hycu.com

Using the HYCU for Google Cloud REST API Explorer

HYCU Data Protection as a Service for Google Cloud (HYCU for Google Cloud) enables you to automate your data protection activities by using its REST-based application programming interface (API). The REST API Explorer is integrated into the HYCU for Google Cloud web user interface and allows you to quickly test and make API requests.

Prerequisites

- A service account is configured in HYCU for Google Cloud. For details on how to do this, see *HYCU for Google Cloud Help*.
- The API requests that you plan to make are authorized. For details on how to do this, see "Authorizing API requests" below.

Accessing the HYCU for Google Cloud REST API Explorer

1. Click the user session information (👤) at the upper right of the screen, and then select **REST API Explorer**. The REST API Explorer dialog box opens.
2. Depending on whether you want to access the central service REST API Explorer or the service subscription REST API Explorer, click one of the following:
   - **Central REST API Explorer**
   - **Subscription REST API Explorer**

The corresponding HYCU for Google Cloud REST API Explorer opens, allowing you to make requests after your access is authorized.

Authorizing API requests

Authorizing API requests includes acquiring and applying a bearer token.

For general information on bearer tokens for APIs of Google Cloud apps, see Google Cloud documentation.

Procedure

1. Acquire a valid OAuth 2.0 bearer token.

   > ⚠ Important Bearer tokens are valid for 60 minutes. To keep API access authorized without interruption, you must apply a new valid token before the validity of the current token expires.

   The following example includes a program in the Python programming language that you can use to acquire a new bearer token for the HYCU for Google Cloud API.

Example

To be able to run this program, replace *<PathToServiceAccountJSONFile>* with the path name of the file that stores the service account information, including its private key, in the JSON format.

```
import json
from google.oauth2 import service_account
from googleapiclient import discovery

SCOPES = ['https://www.googleapis.com/auth/cloud-platform']
SERVICE_ACCOUNT_FILE = '<PathToServiceAccountJSONFile>'

CLIENT_ID = '225038073315-sbrhk8s4hgucmhk1hnd2h2t6ofp0ff5g.apps
.googleusercontent.com'

# GENERATE SERVICE ACCOUNT ID TOKEN
credentials = service_account.Credentials.from_service_account_file(
    SERVICE_ACCOUNT_FILE, scopes=SCOPES)

with open(SERVICE_ACCOUNT_FILE) as f:
    data = json.load(f)

email = data["client_email"]


iamcredentials = discovery.build('iamcredentials', 'v1',
credentials=credentials)

print("Generating ID token for Service Account '%s'..." % email)

name = 'projects/-/serviceAccounts/' + email

generate_id_token_request = {
    'includeEmail': True,
    'audience': CLIENT_ID
}

request = iamcredentials.projects().serviceAccounts().generateIdToken
(name=name, body=generate_id_token_request)
response = request.execute()
id_token = "Bearer %s" % response["token"]

print("Token:\n%s" % id_token)
```

2. Apply the bearer token to both the Central REST API Explorer and the Subscription REST API Explorer to authorize all API requests and to take full advantage of the HYCU for Google Cloud API. To authorize the requests that you will make, do the following:
   a. Copy the bearer token that you acquired.
   b. Access the Central REST API Explorer or the Subscription REST API Explorer.
   c. In the upper-right part of the webpage, click **Authorize**.
   d. In the Available authorizations dialog box, paste the bearer token, and then click **Authorize**.

Example

This example shows how to retrieve a list of instances that belong to the default protection set:

a. Copy the bearer token that you acquired.
b. Access the Subscription REST API Explorer.
c. In the upper-right part of the webpage, click **Authorize**.
d. In the Available authorizations dialog box, paste the bearer token, and then click **Authorize**.
e. Make the following request:

```
GET protectionSets/default-protection-set/vms
```

Example program: assigning a policy to an instance

The following Python program is an example of how you can assign a policy to an instance in your Google Cloud project by using the HYCU for Google Cloud API.

Example

To be able to run this program, make the following replacements in the source code:

- Replace `<PathToServiceAccountJSONFile>` with the path name of the file that stores the service account information, including its private key, in the JSON format.
- Replace `<HYCUProjectName>` with the name of your Google Cloud project.
- Replace `<InstanceUniqueIdentifier>` with the UUID of the instance to which you want to assign the policy.
- Replace `<ProtectionSetName>` with the name of the HYCU for Google Cloud protection set to which your project belongs.
- Replace `<PolicyUniqueIdentifier>` with the UUID of the policy that you want to assign to the instance.

```
import json
from google.oauth2 import service_account
from googleapiclient import discovery
import http
import ssl
from json import JSONDecodeError

REGISTRY_ENDPOINT = 'endpoints.hycu.com'
SCOPES = ['https://www.googleapis.com/auth/cloud-platform']
CLIENT_ID = '225038073315-
sbrhk8s4hgucmhk1hnd2h2t6ofp0ff5g.apps.googleusercontent.com'
SERVICE_ACCOUNT_FILE = '<PathToServiceAccountJSONFile>'

HYCU_PROJECT_NAME = '<HYCUProjectName>'
#Instance unique identifier
VM_UUID = '<InstanceUniqueIdentifier>'
PROTECTION_SET_NAME = '<ProtectionSetName>'
# Policy unique identifier:
POLICY_UUID = '<PolicyUniqueIdentifier>'

def request_id_token(client_id, scopes, request_service_account):
      # Generate credentials
      credentials = service_account.Credentials.from_service_account_file
(request_service_account, scopes=scopes)
      iam_credentials = discovery.build('iamcredentials', 'v1',
credentials=credentials)
      with open(request_service_account) as f:
```

```
            data_file = json.load(f)
      email = data_file["client_email"]
      print("Generating ID token for Service Account '%s'..." % email)
      name = 'projects/-/serviceAccounts/' + email
      generate_id_token_request = {
      'includeEmail': True,
      'audience': client_id
      }

      request = iam_credentials.projects().serviceAccounts().generateIdToken
(name=name, body=generate_id_token_request)
      response_generate_id_token = request.execute()
      return "Bearer %s" % response_generate_id_token["token"]

def get_manager_url(connection, header):
      url = "/api/v1/auth/currentAuthority"

      connection.request(method="GET", url=url, body={}, headers=header)
      r = connection.getresponse()
      output = json.loads(r.read())
      return output['items'][0]['subscriptions'][0]['managerUrl'].split('//')[1]

def get_protection_set_uuid(connection, header, project_name):
      url = "/api/v1/protectionSets/{}".format(project_name)

      connection.request(method="GET", url=url, body={}, headers=header)
      r = connection.getresponse()
      output = json.loads(r.read())
      return output['items'][0]['uuid']

def get_hycu_project_uuid(connection, header, protection_set_name, vm_uuid):
      url = "/api/v2/protectionSets/{}/vms/{}/details".format(protection_set_
name, vm_uuid)

      connection.request(method="GET", url=url, body={}, headers=header)
      r = connection.getresponse()
      output = json.loads(r.read())
      return output['items'][0]['projectUuid']

def policy_assign(protection_set_uuid, hycu_project_uuid, vm_uuid, policy_uuid,
connection, header):
      # Define the body
      body = {
              "list": [
                    {
                    "hycuProjectUuid": hycu_project_uuid,
                    "backupEntityUuid": vm_uuid
                    }
             ]
      }
      json_body = json.dumps(body)
```

```
      url = "/api/v2/protectionSets/{}/policies/{}:assign".format(protection_set_
uuid, policy_uuid)

      connection.request(method="POST", url=url, body=json_body,headers=header)
      return connection.getresponse()

def print_response(response):
      print('Response status: %d' % response.status)
      temp = response.read()
      try:
           print_data = json.loads(temp)
           print(json.dumps(print_data, indent=4, sort_keys=True))
           return print_data
      except JSONDecodeError:
           print(temp)

# Establish connection to Registry
registry_endpoint_connection = http.client.HTTPSConnection(REGISTRY_ENDPOINT)
id_token = request_id_token(CLIENT_ID, SCOPES, SERVICE_ACCOUNT_FILE)
print("Token:\n%s" % id_token)
headers = {
         'Content-type'  : 'application/json',
         'Authorization' : id_token
}

# Establish connection to Manager
MANAGER_ENDPOINT = get_manager_url(registry_endpoint_connection, headers)
print("Manager URL: " + MANAGER_ENDPOINT)
manager_endpoint_connection = http.client.HTTPSConnection(MANAGER_ENDPOINT,
context=ssl._create_unverified_context())

PROTECTION_SET_UUID = get_protection_set_uuid(manager_endpoint_connection,
headers, PROTECTION_SET_NAME)
HYCU_PROJECT_UUID = get_hycu_project_uuid(manager_endpoint_connection, headers,
PROTECTION_SET_UUID, VM_UUID)

# Assign the policy
r = policy_assign(PROTECTION_SET_UUID, HYCU_PROJECT_UUID, VM_UUID, POLICY_UUID,
manager_endpoint_connection, headers)
print_response(r)
```

Provide feedback

For any suggestions and comments regarding this product or its documentation, send us an e-mail to:

info@hycu.com

We will be glad to hear from you!

www.hycu.com

HYCU®