Integration of GCP BigQuery with CT-VL

GCP BigQuery [Overview]

This document describes how to configure and integrate CipherTrust Manager with GCP BigQuery. BigQuery is a fully managed enterprise data warehouse that helps you manage and analyze your data with built-in features like machine learning, geospatial analysis, and business intelligence. BigQuery's serverless architecture lets you use SQL queries to answer your organization's biggest questions with zero infrastructure management. Federated queries let you read data from external sources while streaming supports continuous data updates. BigQuery's scalable, distributed analysis engine lets you query terabytes in seconds and petabytes in minutes.

Thales provides a couple of different methods to protect sensitive data in GCP BigQuery.

Bring Your Own Encryption (BYOE)

- **Data Ingest** – with Thales Batch Data Transformation (BDT)
- **Data Access** – external remote functions for column level encrypt and decryption using Thales tokenization (CT-VL).

Bring/Hold Your Own Key (BYOK) (HYOK)

- **GCP BigQuery Customer Managed Keys-** with Thales CM CCKM BYOK and HYOK.

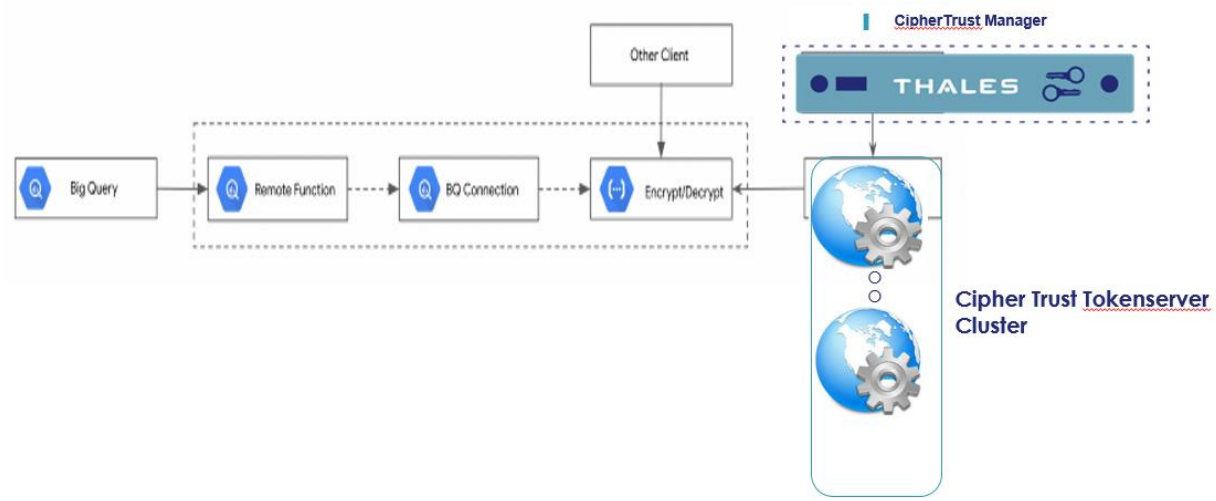
The above methods are NOT mutually exclusive. All methods can be used to build a strong defense in depth strategy to protect sensitive data in the cloud. The focus of this integration will be on Data Access protecting sensitive data in GCP BigQuery columns by using CT-VL to create User Defined Functions (UDF) for encryption and decryption of sensitive data.

The focus of this integration will be on Data Access using remote functions.

Architecture

The examples provided in this document use a capability GCP BigQuery called “Remote Function”. A BigQuery remote function lets you incorporate GoogleSQL functionality with software outside of BigQuery by providing a direct integration with Cloud Functions and Cloud Run. With BigQuery remote functions, you can deploy your functions in Cloud Functions or Cloud Run implemented with any supported language, and then invoke them from GoogleSQL queries.

A BigQuery remote function allows you to implement your function in other languages than SQL and Javascript or with the libraries or services which are not allowed in BigQuery user-defined functions. Listed below is a diagram of how this integration works.



Supported Product Versions

- **CipherTrust Manager** CipherTrust Manager 2.11 and higher
- **CT-VL** CT-VL 2.6 and higher
- **GCP BigQuery**

This integration is validated using python version 3.12.

Prerequisites

Steps performed for this integration were provided by this GCP link:
https://cloud.google.com/bigquery/docs/reference/standard-sql/remote-functions

https://cloud.google.com/bigquery/docs/remote-function-tutorial#console

- Ensure that CT-VL is installed and configured. Refer to https://thalesdocs.com/ctp/con/ct-vl/latest/admin/ct-vl-qs/index.html
- Ensure that the CipherTrust Manager is installed and configured. Refer to the **CipherTrust Manager documentation** for details.
- GCP Cloud function communicates with the CT-VL using port 443. See instructions above or Appendix below for more information on ports.

Steps for Integration

- **[Installing and Configuring Thales CT-VL]**
- **[Publish python code to GCP Cloud Function]**
- **[Create and configure BigQuery connection and GCP BigQuery Remote Function]**

Installing and Configuring CT-VL

To install and configure **CT-VL visit this link.**

https://thalesdocs.com/ctp/con/ct-vl/latest/admin/ct-vl-qs/index.html

The content for this example had the following token group and token templates in CT-VL.

```
p11tokgroup = "tg1"
p11toktemplate = "tt1"
```

The screen below shows the templates, format, token groups and character sets used for this code example.

Name	Format	Token Group	Charset
prefixexample	FPE	Demo	Alphanumeric
tokentemplate	FPE	Demo	Alphanumeric
dateyyyymmdd	YYYYMMDD	Demo	All digits
Text	FPE	Demo	Alphanumeric
Numeric	FPE	Demo	All digits
Credit Card	FPE	Demo	All digits
chardigits	FPE	char	All digits
nbr	FPE	nbr	All digits
charalpha	FPE	char	Alphanumeric
fpe-all-ascii-tt	FPE	cts-tg	All printable ASCII
tt1	FPE	tg1	Alphanumeric

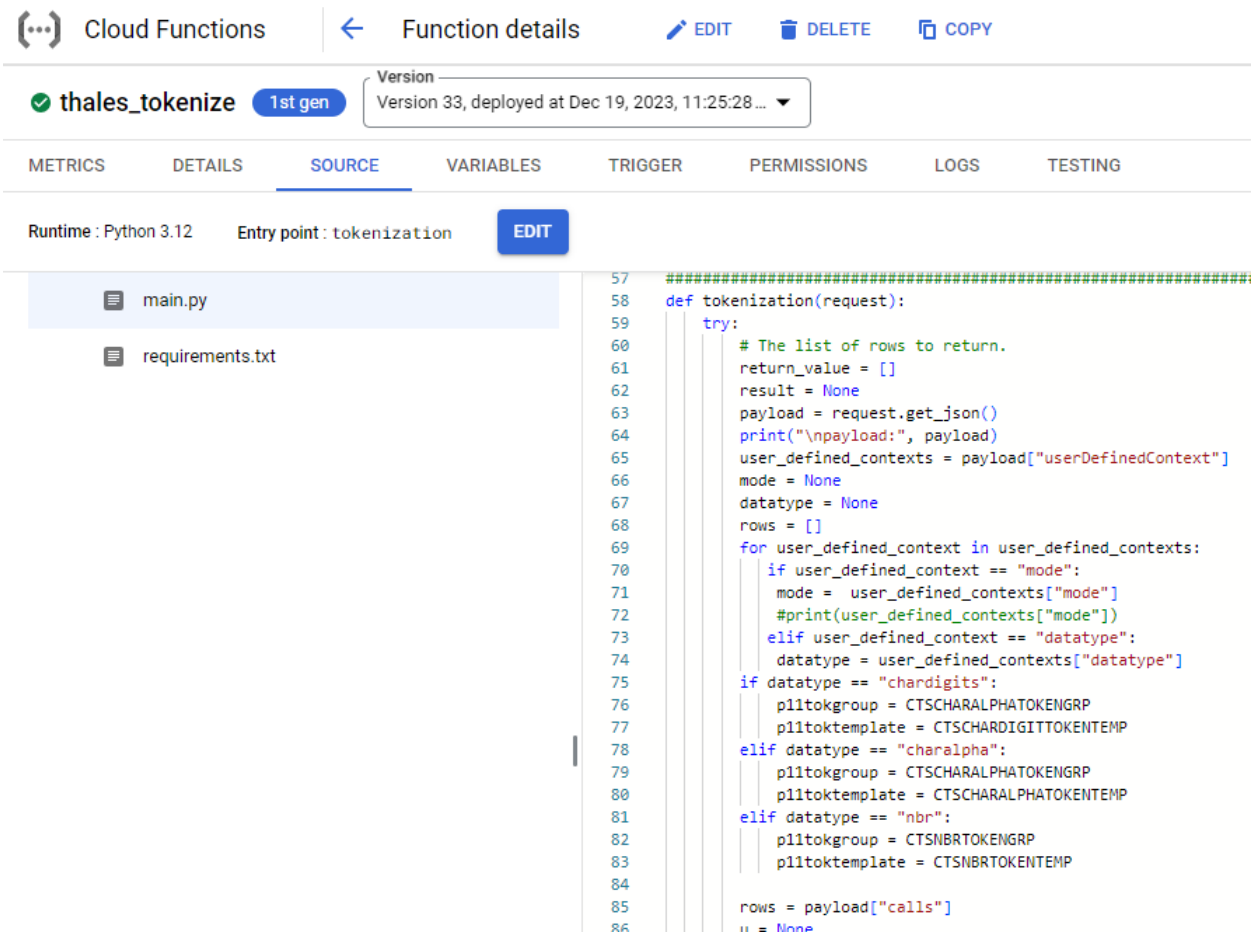
The only other setup required is to have a user created in CT-VL. This user should have the ability to tokenize/detokenize in CT-VL. This user will also be needed to be provided as an environment variable for the function. The examples have the userid/pwd as environment variable, but they could also be obtained from a secrets manager.

Publish python code to Google Cloud Function.

The example provided in git is a python tokenize example.

git clone https://github.com/ThalesGroup/CipherTrust_Application_Protection.git

Once you have obtained the python code from github you can then upload the source to GCP Cloud Function. Once you have uploaded your code in GCP Cloud Function and deployed it you should it should be like the screen below.



To work with Python 3.12 the requirements.txt should contain the following:

Function dependencies, for example:
package>=version
functions-framework
requests==2.26.0
urllib3==1.26.8

Set the Memory to 256MB. This should be able to be reduced depending on your testing.

Set the CTSUSER and CTSPWD and CTSIP to the appropriate environment values.

Be sure to change the entry point to : Entry point : tokenization

Click Deploy to deploy the function.

Once you have created the function above and if you have already configured and setup CM with the key and userid/pwd you can test the function with the test tab. You will need to provide the appropriate json to test. Here is an example:

```
{
  "requestId": "124ab1c",
  "caller":
"//bigquery.googleapis.com/projects/myproject/jobs/myproject:US.bquxjob_5b4c112c_17961fafeaf",
  "sessionUser": "test-user@test-company.com",
  "userDefinedContext": {
    "mode": "decrypt",
    "datatype": "nbr"
  },
  "calls": [
    [
      93309296
    ],
    [
      74705755
    ],
        [
      39056597430
    ],
    [
      6621883
    ]
       ,
       [
      2662402956
    ],
    [
      17506289853
    ]
  ]
}
```

Create and configure BigQuery connection and GCP BigQuery Remote Function.

Here are some links that provide details.

https://cloud.google.com/bigquery/docs/remote-function-tutorial#console

https://cloud.google.com/bigquery/docs/remote-functions

As noted above the steps are:

1. Create the GCP Cloud Function. (should already be done from above)
2. Create GCP BigQuery Connection.
3. Create remote function object in GCP BigQuery

Examples:

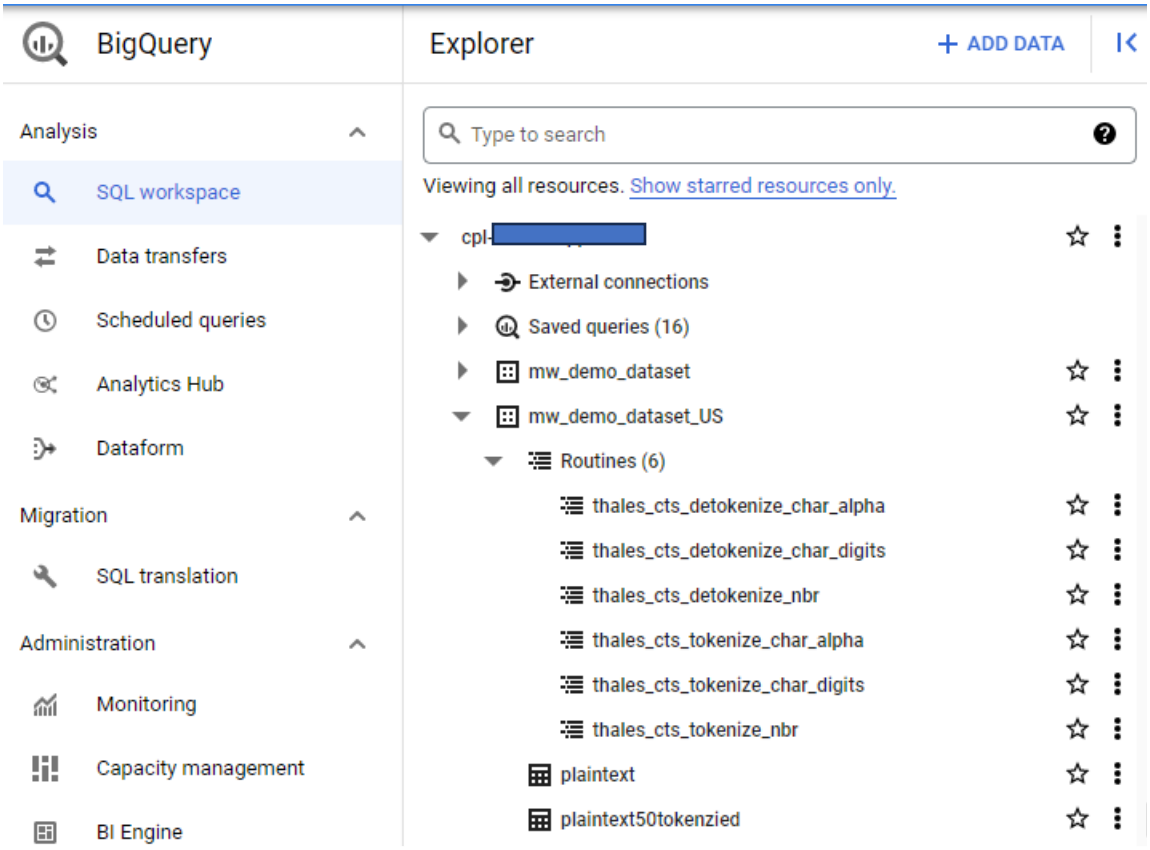
bq mk --connection --display_name='warnerscorner' --connection_type=CLOUD_RESOURCE --project_id=yourprojectid --location=US mw-remote-add-conn

This is the sql to create the Google Remote functions.

```
CREATE or replace FUNCTION `yourproject-uk-
04.mw_demo_dataset_US.thales_cts_tokenize_char_alpha`(x String)
RETURNS String
REMOTE WITH CONNECTION `yourproject-uk-04.us.mw-remote-add-conn`
OPTIONS (
endpoint = 'https://us-central1-yourproject-uk-
04.cloudfunctions.net/thales_tokenize',
user_defined_context = [("mode", "tokenize"),("datatype", "charalpha")]
);
CREATE or replace FUNCTION `yourproject-uk-
04.mw_demo_dataset_US.thales_cts_detokenize_char_alpha`(x String)
RETURNS String
REMOTE WITH CONNECTION `yourproject-uk-04.us.mw-remote-add-conn`
OPTIONS (
endpoint = 'https://us-central1-yourproject-uk-
04.cloudfunctions.net/thales_tokenize',
user_defined_context = [("mode", "detokenize"),("datatype", "charalpha")]
);
CREATE or replace FUNCTION `yourproject-uk-
04.mw_demo_dataset_US.thales_cts_tokenize_char_digits`(x String)
RETURNS String
REMOTE WITH CONNECTION `yourproject-uk-04.us.mw-remote-add-conn`
OPTIONS (
endpoint = 'https://us-central1-yourproject-uk-
04.cloudfunctions.net/thales_tokenize',
user_defined_context = [("mode", "tokenize"),("datatype", "chardigits")]
);
CREATE or replace FUNCTION `yourproject-uk-
04.mw_demo_dataset_US.thales_cts_detokenize_char_digits`(x String)
RETURNS String
REMOTE WITH CONNECTION `yourproject-uk-04.us.mw-remote-add-conn`
OPTIONS (
endpoint = 'https://us-central1-yourproject-uk-
04.cloudfunctions.net/thales_tokenize',
user_defined_context = [("mode", "detokenize"),("datatype", "chardigits")]
);
CREATE or replace FUNCTION `yourproject-uk-
04.mw_demo_dataset_US.thales_cts_tokenize_nbr`(x INT64)
RETURNS String
REMOTE WITH CONNECTION `yourproject-uk-04.us.mw-remote-add-conn`
OPTIONS (
endpoint = 'https://us-central1-yourproject-uk-
04.cloudfunctions.net/thales_tokenize',
user_defined_context = [("mode", "tokenize"),("datatype", "nbr")]
);
CREATE or replace FUNCTION `yourproject-uk-
04.mw_demo_dataset_US.thales_cts_detokenize_nbr`(x INT64)
RETURNS String
REMOTE WITH CONNECTION `yourproject-uk-04.us.mw-remote-add-conn`
OPTIONS (
```

```
endpoint = 'https://us-central1-yourproject-uk-
04.cloudfunctions.net/thales_tokenize',
user_defined_context = [("mode", "detokenize"),("datatype", "nbr")]
)
```

When all the above steps are performed you should see your UDF's in GCP BigQuery under Routines in the UI.



Here is a sample query using one of the UDF's.

Sample Results:

```
select CREDITCARD, `yourproject.mw_demo_dataset_US.thales_cts_tokenize_nbr`
(CREDITCARD) as tokenizedcc from `yourproject.mw_demo_dataset_US.plaintext` where
RECORDNBR < 5
```

Row	CREDITCARD	tokenizedcc
1	5169539515301201	1697402460405557
2	5548473258777835	9566869570210111
3	5564791250929487	2704926296460879
4	5225629041834452	7483419815481511

Advanced Topics.

Google also provides the ability to setup a secure perimeter with VPC Service Controls. See link below for more information.

https://cloud.google.com/bigquery/docs/remote-functions#using_vpc_service_controls

CT-VL Ports

Inbound Ports to CTS

Port	Direction	Purpose
TCP 22	SSH → CTS and CM	Command line administration access on CTS and CM appliances
TCP 443	Application → CTS Admin → CTS GUI Console Admin → CM GUI Console	Access to the administration web GUI as well as application access to the REST API.
TCP 5432	CTS ↔ CTS	PostgreSQL Bi-Directional Replication (BDR) port for database replication across nodes.

Outbound Ports from CTS

Port	Direction	Purpose
UDP 123	CTS ↔ NTP Server	Network time synchronization
TCP 389	CTS → LDAP Directory	Optional CTS read-only access to LDAP/AD for authentication
UDP 514	CTS → Log Server / SIEM	Remote logging via Syslog
TCP 636	CTS → LDAP Directory	Optional CTS read-only access to LDAP over SSL
TCP 9000	CTS → CM	Communication to the NAE interface of the Ciphertrust Manager