## **Streaming De-Identification of a FhirStore**

### **Purpose**

This document provides step-by-step instructions for programmatically de-identifying any FhirStore and posting the de-identified results to a fixed endpoint FhirStore. Specifically a FhirStore will be de-identified whenever there is an update, and the elements that have a timestamp of being modified in the past 10 minutes will be de-identified and updated.

### **Requirements**

### Permissions: Access to google cloud storage, pubsub messaging, compute engine, and fhirstore

Credentials to programmatically use the GCP

After an import is done to a fhirStore, this function must be run:

Text

Description automatically generated

### **Procedure**

#### **Step 1: Create a PubSub Topic In GCP and Create a Subscription to the Topic**

1. Create a PubSub Topic with some name
   1. Under Pub/Sub Topics in GCP
2. Create a PubSub Subscription to that Topic
   1. Under Pub/Sub Subscriptions in GCP

#### **Step 2: Install and Import the Necessary Libraries**

pip install google-cloud-pubsub

pip install google-auth google-auth-oauthlib google-auth-httplib2

pip install google-cloud-dlp

pip install google-api-python-client

pip install google-cloud-storage

pip install datetime

imports for libraries used

Text

Description automatically generated

#### **Step 4: Load the Credentials and Initialize the PubSub Client**

1. If the credentials are in the environment variables no need to load them
2. The path to the subscription is the subscription you made to the PubSub topic in Step 1

Text

Description automatically generated

#### **Step 5: Create a Python Function that take a parameter and will be used for De-Identification**

1. Parameter name is: datastore (sent with script that did the import call to fhirStore)
   1. Datastore sent has the following format:
      1. This will allow for a dynamic source for a fhirStore



#### **Step 6: Format the Datastore Parameter Passed in, Set the Project Id and Location, Find and Set De-Identification Template, Load the De-Identification Client**

1. Find the de-identification client template id to use on the GCP DLP Configuration page

Text

Description automatically generated

**Step 7:** **Create a File Called Resources that will Have a List of All the Resources In a FhirStore**

1. Each entry in the file should have a name on its own line being delimited by an “enter” after each resource name

**Step 8: Load all the Entries from the Fhir Resources File**

1. Loop through all the entries and add it to a list

Text

Description automatically generated

**Step 7: Create a loop that will de-identify each resource if there are entries to de-identify.**

1. Loops through all the resources in the file resources that was created
   1. Sets the resource path to get 50 entries from that resource page in the fhirStore
      1. Note: later reset the resource path to get the next page of data if data exists there
   2. Get the current datetime and subtract a time delta (example 10 minutes)
      1. De-Identify only resources that were modified after this time delta
      2. Format this time delta to be used to query the fhirStore
   3. Set a counter that can be used to create unique file names when posting to google cloud storage bucket

Text

Description automatically generated

1. Load the data in the resource path and check if there’s data that needs de-identifying

Text

Description automatically generated

1. Loop through each entry and have a parameter to scan for those elements in the data

Text

Description automatically generated

1. Convert the data into a usable format, de-identify it with the parameters previously created

Text

Description automatically generated

1. Format the data so that it can be imported into a google cloud storage and then a fhirStore

Text

Description automatically generated

1. Create a storage client to post the data to in a cloud storage bucket. Use the counter to have a unique name for each entry. Need a sleep because of the API request requirements.
   1. Assumes a cloud storage exists with this name already

Text

Description automatically generated

1. Once all the entries on a page of a resource have been processed change the path to the next page of the same resource and check if there’s data there to process. Otherwise move on to the next resource to process and continue until all of them are done.

Text

Description automatically generated

1. Once all the data have been posted to the google cloud storage, import the data to a hardcoded fhirStore end point.
   1. Assumes the destination fhirStore already exists
   2. Have the contentStructure be: RESOURCE

Text

Description automatically generated

**Step 8: Create a function that will be used to listen for a PubSub message on the Subscription and Handle the Message**

1. The function prints a message that was received from the PubSub.
2. Calls the de-identification() function to run when a pubsub message is received
   1. Converts the data sent from PubSub to utf-8 because sent in byte string

Graphical user interface, text

Description automatically generated

**Step 9: Create a Loop that Keeps Listening for a PubSub Message from the Subscription Endpoint Set Earlier**

1. Have a method to handle the message once received (callback function built in previous step)
2. Has a way to exit through user interruption
3. Displays an error message when an error occurs trying to process data

**Text

Description automatically generated**

**Step 10: Create a Google Compute Engine so that the Script can Run in the Background and Start De-Identifying Once a Message has Been Received**

1. Go to Google Compute Engine and Create a e2-small instance

The Specifications of the Instance are as follows:

**Table

Description automatically generated**

**Table

Description automatically generated**

**Graphical user interface, application

Description automatically generated** **Graphical user interface, text, application, email

Description automatically generated**

Graphical user interface, text, application

Description automatically generated

1. Update the Instance with
   1. sudo apt-get update && sudo apt-get upgrade -y
2. Install Python Pip on the Server
   1. sudo apt install python3-pip
   2. check if it was installed correctly with pip3 –version
   3. Install all the libraries on this machine using **Step 3**
   4. sudo apt-get install tmux
      1. used to run a session in the background even when terminal session is closed
3. Can set variable for google application credential with
   1. GOOGLE\_APPLICATION\_CREDENTIALS=involuted-woods-360619-59d8e0b17cdb.json
      1. Test with echo $GOOGLE\_APPLICATION\_CREDENTIALS
         1. Should equal path to credential file
4. Upload all the files used for De-Identification into the instance
   1. Resources.txt
   2. De-identification script built
   3. Secret key file
5. tmux new -s my-session -d # to create a background session
6. tmux ls # lists all the sessions
7. tmux attach -t my-session # go to that session
8. Ctrl-b then click d to go back to the terminal
9. Execute step 7 to go to the session and run the script with:
   1. python3 script\_name.py
   2. The script should be running and you can exit the terminal with step 9
   3. To test if its working can force run the pub sub scheduler set up earlier
      1. Go back to the session and see if the message was printed and de-identification started
      2. Due to the way it was created the task takes a long time to de-identify a large dataset
10. Don’t stop the instance when done otherwise it won’t continue running in the background

**Conclusion:**

Created a script that runs in the background of a Google Compute Engine that is able to de-identify entries in a fhirStore that were modified in the last 10 minutes and post the results to another hard coded fhirStore (endpoint).