

DataEng: Data Maintenance In-class Assignment

Will McIntosh

This week you will construct a data archiver that compresses, encrypts and stores pipelined data into a low-cost, high-capacity GCP Storage Bucket.

Submit: Make a copy of this document and use it to record your responses and results (**use colored highlighting when recording your responses/results**). Store a PDF copy of the document in your git repository along with your code before submitting for this week.

Develop a new python PubSub subscriber similar to the subscribers that you have created multiple times for this class. This new subscriber (archive.py) will receive data from a PubSub topic, compress the data, encrypt the data and store the resulting data into a [GCP Storage Bucket](#).

A. [MUST] Discussion Questions

When archiving data for a data pipeline we could (a) compress, (b) encrypt and/or (c) reduce the data. Here, “reducing the data” refers to the process of interpolating or aggregating detailed data, such as 5 second breadcrumbs for all buses on all trips, into coarser data. For example, we could aggregate 5-second breadcrumbs into 30-second breadcrumbs.

Under what circumstances might each of these transformations (compress, encrypt, reduce) be desirable for data archival?

Would it make sense to combine these transformations?

- **Compress** - Storage saving
 - Do you need to save storage space?
- **Encrypt** - Security
 - Is your data sensitive?
- **Reduce** - Interpretability of data (less cloudy noisy data)
 - Reducing to 30 second-breadcrumbs or removing columns is easier to read.
 - Dropping out data that could be considered infringements on data privacy.
 - Is some of the data unnecessary (hair color, eye color, pets name, etc).
- **All Three** - Compress & Encrypt
 - We'd want to reduce first, compress next, and then lastly encrypt.

B. [MUST] Create Test Pipeline

Create a new PubSub topic called “archivetest” or something similar. Create a new subscriber program (call it archiver.py) that subscribes to the topic, receives the data and (for now) discards it.

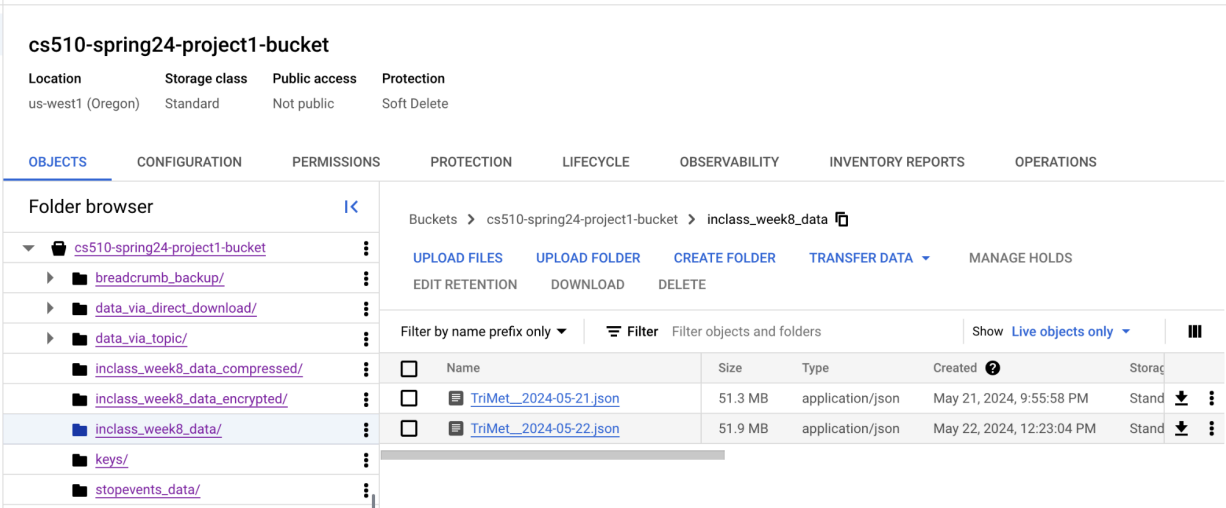
To produce test data, copy/reuse the publisher program used for your class project, and alter it to publish to the new archivetest topic. Run this test publisher manually to gather data (1 day and 100 vehicles) from busdata.cs.pdx.edu and test the archivetest topic and your archiver.py program.

As always, you can/should test your code with smaller data sets first. Try it with just one bus or one trip, and then when everything is working, run it with 100 vehicles.

C. [MUST] Store Data to GCP Storage Bucket

Modify archiver.py to store all received data to a [GCP Storage Bucket](#). You will need to create and configure a Storage Bucket for this purpose. We recommend using the Nearline Storage class for this assignment though you are free to choose any of the offered classes of service. Be sure to remove the bucket at the end of the week to reduce GCP credit usage.

How much bucket space (in KiBs) does it take to store 1 day of breadcrumbs for 100 vehicles?



The screenshot shows the Google Cloud Storage console for the bucket 'cs510-spring24-project1-bucket'. The bucket is located in 'us-west1 (Oregon)', has a 'Standard' storage class, 'Not public' access, and 'Soft Delete' protection. The 'OBJECTS' tab is active, showing a folder browser on the left and a list of objects on the right. The folder browser shows a hierarchy starting with 'cs510-spring24-project1-bucket', followed by folders like 'breadcrumb_backup/', 'data_via_direct_download/', 'data_via_topic/', 'inclass_week8_data_compressed/', 'inclass_week8_data_encrypted/', 'inclass_week8_data/' (selected), 'keys/', and 'stopevents_data/'. The object list on the right shows two JSON files: 'TriMet_2024-05-21.json' (51.3 MB) and 'TriMet_2024-05-22.json' (51.9 MB), both of type 'application/json' and created on May 21, 2024, and May 22, 2024, respectively.

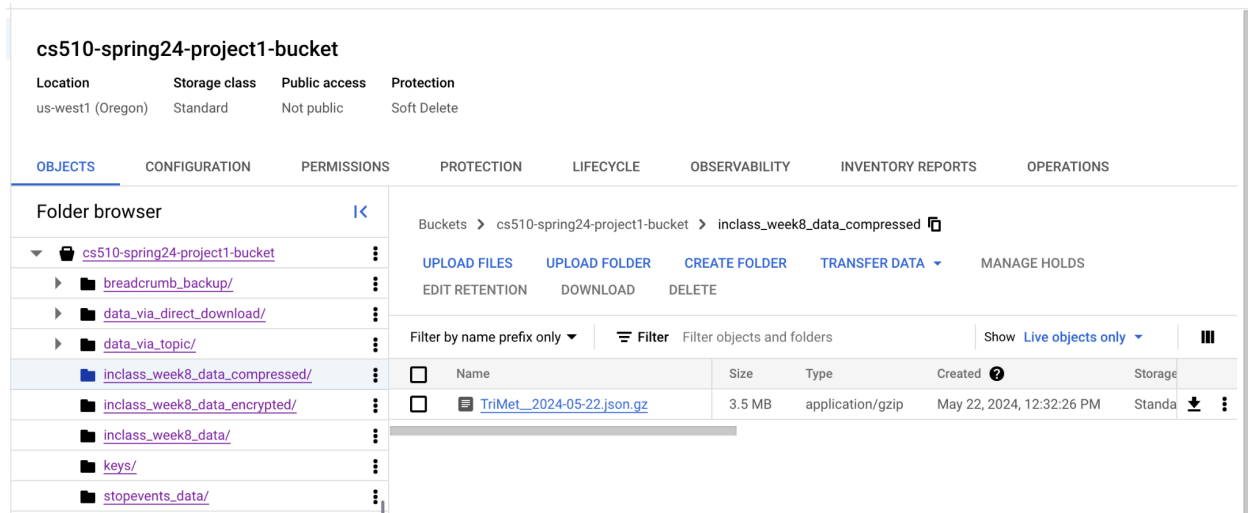
Name	Size	Type	Created	Storage
TriMet_2024-05-21.json	51.3 MB	application/json	May 21, 2024, 9:55:58 PM	Standard
TriMet_2024-05-22.json	51.9 MB	application/json	May 22, 2024, 12:23:04 PM	Standard

- Converting 51.3MB to KiBs is 50683.59 kibibytes (focusing on the 5/22 json file).

D. [SHOULD] Compress

Modify archiver.py to compress the data before it stores the data to the storage bucket. Use [zlib compression](#) which is provided by default by python. How large is the archived data compared to the original?

How much bucket space (in KiBs) does it take to store the compressed data?



cs510-spring24-project1-bucket

Location	Storage class	Public access	Protection
us-west1 (Oregon)	Standard	Not public	Soft Delete

OBJECTS CONFIGURATION PERMISSIONS PROTECTION LIFECYCLE OBSERVABILITY INVENTORY REPORTS OPERATIONS

Folder browser

Buckets > cs510-spring24-project1-bucket > inclass_week8_data_compressed

UPLOAD FILES UPLOAD FOLDER CREATE FOLDER TRANSFER DATA MANAGE HOLDS

EDIT RETENTION DOWNLOAD DELETE

Filter by name prefix only Filter Filter objects and folders Show Live objects only

Name	Size	Type	Created	Storage
TriMet_2024-05-22.json.gz	3.5 MB	application/gzip	May 22, 2024, 12:32:26 PM	Standa

- Converting 3.5MB to KiBs is 3417.97 kibibytes (focusing on the 5/22 json file).

E. [SHOULD] Encrypt

Modify archiver.py to encrypt the data prior to writing it to the Storage Bucket. Your archive.py program should encrypt after compressing the data. Use RSA encryption as described here: [link](#). There is no need to manage your private encryption keys securely for this assignment, and you may keep your private key in a file or within your python code.

Be sure to test your archiver by decrypting and decompressing the data stored in the Storage Bucket. We suggest that you create a separate python program for this purpose.

How much bucket space (in KiBs) does it take to store the encrypted, compressed data?

```

▶ # Example usage
filename = 'TriMet__2024-05-22.json.enc'
data = download_and_process_file(filename)
print(data)

```

Downloaded data length: 54409037
IV length: 16
Encrypted AES key length: 256
Encrypted message length: 54408765
Decryption failed: Decryption failed

```

ValueError                                Traceback (most recent call last)
<ipython-input-18-d446446a03c3> in <cell line: 3>()
      1 # Example usage
      2 filename = 'TriMet__2024-05-22.json.enc'
----> 3 data = download_and_process_file(filename)
      4 print(data)

```

1 frames

```

<ipython-input-17-8df1b0815027> in decrypt_data(encrypted_data)
      42     print(f"Encrypted message length: {len(encrypted_message)}")
      43
----> 44     aes_key = private_key.decrypt(
      45         encrypted_aes_key,
      46         padding.OAEP(

```

ValueError: Decryption failed

- I couldn't figure it out! I wrote some code that encrypts the files in the **inclass_week8_receiver.py** using encryption keys created by my **inclass_week8_encryptionkeycreator.py** code saved in my bucket but using the same keys in my **week8_decryption_test.ipynb** Google Colab notebook, I couldn't figure out how to decrypt the files and display them as dataframes to compare them with the non-encrypted files. It's a bummer it didn't work out!

F. [ASPIRE] Add Archiving to your class project

Add an archiver to your class project's pipeline(s). To receive extra credit, mention your archiver when submitting the next part of your project. You should only need one archiver for the entire project, so coordinate with your teammates if you choose to take this step. For the class project, we recommend storing to a Google Storage Bucket and compressing. Encryption is OK too but not necessary.