**IBM Notes/Domino to IBM Cloud Object Storage Migration**

**What is the Problem/Challenge?**

We all know that IBM is planning to discontinue with Lotus Notes very soon. There are many applications which runs on Lotus Notes and we must either sunset or migrate those applications to some other platform.

For any migration project, the most critical question is – **“How you manage your existing Lotus Notes data?”**.

Existing Lotus Notes applications can have very important and confidential information which you may require to migrate to new environment for storage, reference, compliance or audit purpose. Unfortunately, there are very less options available to migrate Lotus Notes data to Cloudant or MongoDB or IBM Cloud Object Storage. There are some 3rd party tools, but it’s not secure and there is a risk exposing or compromising confidential data. We need a solution which is secure, scalable, easier to understand and implement. And for that purpose, I have developed solutions which can help team to migrate data from Lotus Notes to other databases on Cloud.

Previously I submitted 2 assets which describes solutions to migrate Lotus Notes data to Cloudant and MongoDB. Here are the links,

Lotus Notes to Cloudant – <https://w3.ibm.com/services/lighthouse/knowledge/documents/145622>

Lotus Notes to MongoDB – <https://w3.ibm.com/services/lighthouse/knowledge/documents/152242>

The challenge with these assets is the Attachments. Lotus Notes documents can store many attachments with different file types and sizes. Both Cloudant and MongoDB has certain limitation on attachment size (e.g MongoDB has 16MB file size limit). Also, it’s not a good practice to store attachments in traditional databases.

**What is the solution?**

**IBM Cloud Object Storage** is the most suitable choice to store attachments. IBM Cloud® Object Storage makes it possible to store practically limitless amounts of data, simply and cost effectively. It is commonly used for [data archiving](https://www.ibm.com/cloud/object-storage/data-archiving) and [backup](https://www.ibm.com/cloud/object-storage/backup-and-recovery); for web and mobile applications; and as scalable, persistent storage for analytics. [Flexible storage class tiers](https://www.ibm.com/cloud/object-storage/storage-class-tiers-archive) with a policy-based archive let you effectively manage costs while meeting data access needs. The integrated IBM [Aspera® high-speed data transfer](https://www.ibm.com/cloud/object-storage/aspera) option makes it easy to transfer data to and from IBM Cloud Object Storage. Query-in-place functionality allows you to run analytics directly on your data.

I have developed this solution to migrate attachments from Lotus Notes database to IBM Cloud Object Storage. This solution offers several benefits,

1. It’s easy to implement and it can be used with any IBM Notes database.
2. It’s based-on HTTP and REST, so it’s fast and secure.
3. Its written in Python which has excellent set of libraries to work with Data (Data Cleaning, Data Processing etc).
4. You can migrate any number of attachments of different file types (jpg, pdf, xlxs, docx, pptx, wmv etc).
5. There is no limitation on the attachment size.
6. Most importantly, once you are out of IBM Notes it will give you lot of options and opportunities to scale your application using modern tools. You can implement DevOps, TDD, AI and many others (Opportunities are endless).

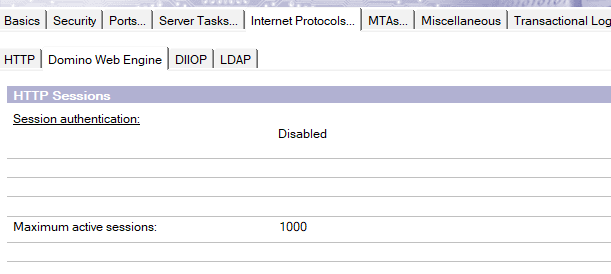
**How to implement the solution?**

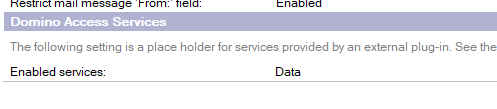
Here are the steps to implement this solution,

**Enable Domino Data Services in Domino server**

You need to enable Domino Data Service on your Domino web server so that you can access your database with Domino API. Make sure your domino server is web enabled.

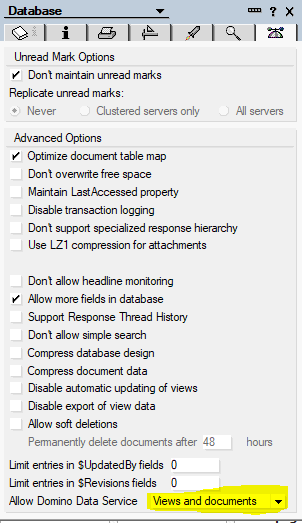
Open server in Domino Administrator. Go to “Configuration🡪Current Server Document🡪Internet Protocols🡪Domino Web Engine”. In “Domino Access Service” section, select “Data” under Enabled Services and Save.



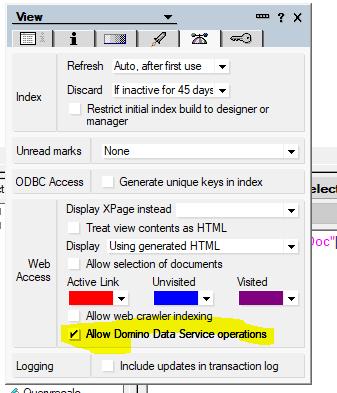


**Enable Domino Data Services for views and database**

Go to Database properties and in the last tab, select “Views and documents” under Allow Domino Data Service,



Open the view which you need to migrate in Designer and in second last tab of View Properties, check “Allow Domino Data Service operations” under Web Access,



**Prepare View and CSV**

Next step is to prepare the data you need to migrate. We will access Document using Domino API and we will target each document based on its Document Unique ID (UNID). To get UNID, make sure to create a column in the view (preferably 1st column) with this formula,

@Text(@DocumentUniqueID)

This will populate document unique ID of every document in the view in that column. Once it’s done, open this view in Notes client and Export it in a CSV file using File🡪Export option and save the file.

**Migrate attachments to IBM Cloud Object Storage (Python Script)**

import ibm\_boto3  
from ibm\_botocore.client import Config, ClientError  
import requests  
import pprint  
import base64  
import os  
from cos\_config.resource\_configuration\_v1 import ResourceConfigurationV1  
  
*# Cloud Object Storage credentials*credentials = {  
 "apikey": "<API-KEY>",  
 "endpoints": "<ENDPOINT>",  
 "iam\_apikey\_description": "<API-KEY-DESCRIPTION>",  
 "iam\_apikey\_name": "<API-KEY-NAME>",  
 "iam\_role\_crn": "<ROLE-CRN>",  
 "iam\_serviceid\_crn": "<SERVICE-ID-CRN>",  
 "resource\_instance\_id": "<INSTANCE-ID>"  
}  
*# Create COS Client*cos = ibm\_boto3.client(service\_name='s3',  
 ibm\_api\_key\_id=credentials['apikey'],  
 ibm\_service\_instance\_id=credentials['resource\_instance\_id'],  
 ibm\_auth\_endpoint="<AUTH-ENDPOINT>",  
 config=Config(signature\_version='oauth'),  
 endpoint\_url="<ENDPOINT-URL>")  
  
*# Get list of all the buckets [Optional]*res = cos.list\_buckets()  
print(res['Buckets'])  
  
domino\_url = "https://<SERVER>/<DATABASEPATH>/api/data/documents/unid/<DOC\_UNID>"  
resp = requests.get(domino\_url, verify=False)  
*# JSON response of Notes document*resp\_json = resp.json()  
*# In this case, "Body" field has attachments. So we are getting the content of Body field.  
# If there are more fields with attachments, you have to include those fields as well.*body\_content = resp\_json['Body'].get('content')  
for item in body\_content:  
 *# Check if content is base64* if item.get('contentTransferEncoding') == "base64":  
 *# Get attachment name* start\_string = item.get('contentDisposition').find("=")  
 attachmentName = item.get('contentDisposition')[start\_string + 1:].replace('"','')  
 print(attachmentName)  
 *# Get base64 attachment data* b64Data = item.get("data")  
 *# Encode to utf-8* b64Data\_bytes = b64Data.encode('utf-8')  
 *# Decode base64 data and store it in a local file in your computer* with open(attachmentName, 'wb') as f:  
 decoded\_file\_data = base64.decodebytes(b64Data\_bytes)  
 f.write(decoded\_file\_data)  
 *# Read attachment file as binary* with open(attachmentName,'rb') as f\_cos:  
 *# Upload to IBM Cloud Onject Storage* cos.upload\_fileobj(f\_cos, Bucket='swlc-cloud-object-storage-cos-migration-test', Key=attachmentName)  
 *# Remove attachment file from your local computer* os.remove(attachmentName)  
 print(f'Attachment {attachmentName} is successfully migrated to IBM Cloud Object Storage')