Integrate Vaultastic V4 Email Archiving with Node.js

Overview

Get the instructions to use the NodeJS SDK to upload data to the Vaultastic Active Store using the Vaultastic Mail Upload API.

Prerequisites

1. You should have your Vaultastic domain provisioned.
2. You should have created the vaults in the Active Store to which you want to upload the data.
3. You have got the API key from Mithi.
4. You should have NodeJS version 14.5.0 or above.
5. The following are the transitive dependencies and versions:
    - npm - 7.5.2
    - form-data - 4.0.0
    - mailparser - 3.1.0
    - uuid - 8.3.2
    - winston - 3.3.3
    - yargs - 16.2.0
    - node-fetch - 2.6.1
6. The following environment variables should be set:
    a. PRIMARY_DOMAIN: The primary domain name for filtering vaults from the mail.
    b. API_ENDPOINT_URL: ActiveStoreMailUploadAPI.vaultastic.com
    c. API_KEY: The API Key shared for uploading mail to v4.

Uploading mail

To upload mail to a vault in the Active Store of your Vaultastic domain use the script: index.js

The command-line arguments are as follows:

Required arguments

-e <EmlFilePath> where <EmlFilePath> is the path of the EML file.

-m <MailContents> where <MailContents> are the mail contents

Optional Arguments

-r <Recipients> where <Recipients> are the comma-separated list of ids on the primary domain

-s <Sender> where the sender is an id on the primary domain

Note:

1. Use either -e or -m option but not both. If both are used the -m value will be ignored.
2. Use -r and -s only if the sender or recipient is not part of the mail headers From, To, CC or BCC.

Exit Codes and Messages

Exit Code	Description	Resolution
0	Mail Successfully uploaded	N.A.
1	Unknown Exception	This is the exit code returned when there is no other specific exit code defined for the error that occurred. More Information can be found in the message/stack trace printed by nodeJS.
2	Misuse of shell builtins (UNIX reserved error)	Missing keyword or command, or permission problem (and diff return code on a failed binary file comparison)
3	Invalid number of arguments	an invalid number of arguments passed. Provide at least -i or -m.
4	Invalid Recipient	Invalid Recipient Email ID. Make sure the id passed on -r argument is a valid email id (or email ids are comma separated).
5	Failed to Upload mail to S3	Mail Failed to upload to s3 bucket. Check Error logs for the failed message.
6	Recipients not from PRIMARY_DOMAIN	At least one recipient should be from the primary domain. 1. Check if PRIMARY_DOMAIN is correct OR 2. Check any of the recipients in the mail (from, to, cc, or bcc headers) is from the PRIMARY_DOMAIN. OR 3. Provide a recipient id using the -r argument.

Limits

1. No multi-domain support. That is recipient filtering based on multiple PRIMARY_DOMAIN values is not supported.
2. Rate Limits: 10,000 Requests per second with 5000 Burst requests.

3. Max Mail Size: 50MB

Resources

- VaultasticActiveStoreMailUpload-NodeJS-SDK-1.0.2.zip

  (https://https://vaultastic.mithi.com/res/VaultasticActiveStoreMailUpload-NodeJS-SDK-1.0.2.zip)

Example Implementation

Vaultastic NodeJs SDK is developed to be used as a CLI tool as well as an external NodeJs API wrapper module for using Vaultastic mail upload APIs.

1. Vaultastic NodeJs SDK as CLI Implementation
   a. Navigate to the root directory of the project folder in a PowerShell/terminal window.
   b. Run node index.js
2. Implementation of VaultasticActiveStoreMailUpload-SDK as S3 triggered Lambda
   I. Pre-Requisite:
      a. Memory Required: 128MB (for up to 5MB Mails) - more memory required as the mail content resides in the same process
      b. Lambda Timeout: 3 mins (180 seconds)
      c. Runtime: NodeJS 14.x
      d. Handler: index.handler
      e. Environment Variables:
         i. API_ENDPOINT_URL: https://vaultasticactivestoremailuploadapi.vaultastic.com
         ii. API_KEY: <Your API KEY>
         iii. PRIMARY_DOMAIN: <your primary domain name for filtering vaults from the mail>
      f. Make sure Lambda has Internet access. (for VPC Lambda see this: https://aws.amazon.com/premiumsupport/knowledge-center/internet-access-lambda-function/)
   II. Sample Code:
      a. Using the sample code given create a js file index.js

```
const vaultastic = require('./nodejs-SDK/src/vaultastic');
const log = require('./nodejs-SDK/src/shared/logger')
const aws = require('aws-sdk');

exports.handler = async (event, context) => {
   var s3Event = await ParseJson(event);
   await uploadMail(s3Event);
  };

async function ParseJson(event){
   var s3Event = event.Records[0].s3;
   return s3Event;
}

async function uploadMail(s3Event) {
   const response = await vaultastic.vaultastic(await preCall(s3Event), "aarav@mithi.int","aarav@
mithi.int");

   if(response.err)
      return log.error('Error occured:: ', err, response.code);

   log.info('Success:: ', response.res, response.code);

}

async function preCall(s3Event){
   const s3 = new aws.S3(); // Pass in opts to S3 if necessary

   var getParams = {
      Bucket: s3Event.bucket.name, // your bucket name,
      Key: s3Event.object.key // path to the object you're looking for
   }

   var data = await s3.getObject(getParams).promise();

   // Convert Body from a Buffer to a String
   let objectData = data.Body.toString('utf-8'); // Use the encoding necessary

   return objectData;
}
```

b. Add this index.js to VaultasticActiveStoreMailUpload-NodeJS-SDK-1.0.2.zip

c. Upload zip to Lambda Function with the above parameters.

d. Upload mail to S3 and it should trigger the lambda to upload mail to Vaultastic Active Store.