Practical Worksheet 3

Version: 1.0 Date: 12/04/2018 Author: David Glance

## Learning Objectives

1. Learn how to create and configure S3 buckets and read and write objects to them
2. Learn how to use operations on DynamoDB: Create table, put items, get items
3. Start an application is your own personal Cloud Storage

## Technologies Covered

Ubuntu

AWS

AWS S3

AWS DynamoDB

Python/Boto scripts

VirtualBox

**Note**: Do this from your VirtualBox VM – if you do it from any other platform (Windows, Mac – you will need to resolve any potential issues yourself)

## Background

The aim of this lab is to write a program that will:

[1] Scan a directory and upload all of the files found in the directory to an S3 bucket, preserving the path information

[2] Store information about each file uploaded to S3 in a DynamoDB

[3] Restore the directory on a local drive using the files in S3 and the information in DynamoDB

## Program

## [Step 1] Preparation

Download the python code cloudstorage.py

Create a directory *rootdir*

Create a file in *rootdir* called *rootfile.txt* and put some content in it “1\n2\n3\n4\n5\n”

Create a second directory in *rootdir* called subdir and create another file *subfile.txt* with the same content as rootfile.txt

## [Step 2] Save to S3

Edit cloudstorage.py to take one argument: -i, --initialise=True – this will use boto to create a bucket on S3 that is identified by <student number>-cloudstorage

Insert boto commands to save each file that is found as the program traverses the directory starting at the root directory *rootdir*.

**NOTE** the easiest way to upload files is to use the command:

s3.meta.client.upload\_file()

## [Step 3] Restore from S3

Create a new program called restorefromcloud.py that reads the S3 bucket and writes the contents of the bucket within the appropriate directories. You should have a copy of the files and the directories you started with.

## [Step 4] Write information about files to DynamoDB

Create a table on DynamoDB with the key userId

The attributes for the table will be:

CloudFiles = {

'userId',

'fileName',

'path',

'lastUpdated',

'owner',

'permissions'

}

)

For every file that is stored in S3, get the information to put in the DynamoDB item and write it to the table. You will have to find functions in Python to get details like time lastUpdated, owner and permissions. All of this information can be stored as strings.

## [Step 5] Optional

Add the functionality to apply changes to permissions and ownership when the directory and files are restored.

Check timestamps on files and only upload if the file has been updated.

## Submission and Quiz

Submit the python code files you wrote

## Respond to the Quiz

[1] Is the statement: passing a file name with multiple parts separated by a ‘/’ to the upload\_file command will create separate buckets for each part of the path:

[A] True

[B] False

[C] Neither: You can’t use the upload\_file with a filename that has multiple parts

[2] Is the satement: To create a table in DynamoDB you need to specify all attributes of the items you are going to store in the table

[A] True

[B] False