I. Introduction

1. Overview

[10/14/2020 12:07 PM CST]
AAM Auto Backup elegantly solves every data backup and restoration problem and issue of Asian-American Man: AAM Auto Backup is designed to satisfy all of Asian-American Man's data backup and restoration needs in very easy-to-use, hassle-free ways.

[10/14/2020 7:38 AM CST]
AAM Auto Backup is a CLI Python program that fully automates backing up data to a local or network-attached data storage and AWS S3 Glacier Deep Archive.

AAM Auto Backup also automates restoring data from AWS S3 Glacier Deep Archive: Executing a single AAM Auto Backup command is all that is required for restoring data from AWS S3 Glacier Deep Archive, for emergency data restoration.

[10/9/2020 11:19 AM CST] email notification of backup report via AWS SES.

Currently, data backup is supported only on Windows: AAM Auto Backup uses the Windows Shadow Copy Service to back up all the user-specified directories and their contents, even the files in use.

Using Task Scheduler on Windows, AAM Auto Backup can be executed automatically everyday, in the regular backup mode, for performing data replication to a local or network-attached data storage (likely a USB hard drive), and full and incremental backup to AWS S3 Glacier Deep Archive.

AAM Auto Backup can be also executed manually for on-demand incremental backup to a local or network-attached data storage, to AWS S3 Glacier Deep Archive, or both.

2. AAM Auto Backup Inputs and Outputs

[10/7/2020 3:58 PM CST]

2.1. Inputs

AAM Auto Backup requires a number of command arguments for performing a data backup operation.

In particular, for performing a backup operation, AAM Auto Backup requires an AAM Auto Backup instruction plain text file, the full path of which is provided as an argument in the AAM Auto Backup backup command.

AAM Auto Backup also requires a number of command arguments for performing a data restoration operation.

2.2. Final Outputs

After each successful data backup operation, AAM Auto Backup produces the updated backup in the user-specified local or network-attached data storage, the full or incremental backup in AWS S3 Glacier Deep Archive, or both.

After each successful data restoration operation, AAM Auto Backup produces a copy of the restored data in the user-specified local ~~or network-attached~~ data storage.

[10/14/2020 10:18 AM CST]
AAM Auto Backup creates and saves a log file after each operation.

[10/2/2020 5:59 AM CST]
AAM Auto Backup sends data backup and restoration report emails via AWS SES. (As such, it requires an AWS SES email-sending access.)

2.2.1. AWS S3 backup object name formats

2.2.1.1. [12/17/2020 8:54 PM CST]

[12/17/2020 9:09 PM CST]
update the software design and code. the archive file name format is incorrect in the current design: colon cannot be present (currently, 7z.exe excution error because of this wrong archive file name format)!. [12/17/2020 7:57 PM CST] ([12/17/2020 9:09 PM CST] make sure to get this right, and document everything.)
first, decide which character to used instead of colon. [12/17/2020 8:50 PM CST]
[12/17/2020 9:09 PM CST] maybe use underscore instead of colon? I need to resolve the issue of "backup source unique name" having one or more of the demarcation character. actually, I think it can have the demarcation character(s): regular expression can handle that.
FINAL DECISION [12/18/2020 3:58 PM CST] use underscore instead of colon.

[10/30/2020 12:22 PM CST] [12/18/2020 3:58 PM CST] RESTRICTIONS
a unique backup-source location name cannot contain underscore.

<u>full-backup AWS S3 backup object name format</u>

(a unique backup-source location name)_YYYYMMDD_HHMMSS.SSSSSS_0_0.0_no_(backup source unique name).zip.###

YYYYMMDD is full-backup date.
HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
0_0.0 indicates this is full backup, not incremental backup.
'no' indicates that this is not an upload-completion indicator AWS S3 object.

<u>full-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name)_YYYYMMDD_HHMMSS.SSSSSS_0_0.0_yes_(backup source unique name)

YYYYMMDD is full-backup date.
HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
0_0.0 indicates this is full backup, not incremental backup.
'yes' indicates that this is an upload-completion indicator AWS S3 object.

<u>incremental-backup AWS S3 backup object name format</u>
(a unique backup-source location name)_YYYYMMDD_HHMMSS.SSSSSS_YYYYMMDD_HHMMSS.SSSSSS_no_(backup source unique name).zip.###

first YYYYMMDD is full-backup date.
first HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
second YYYYMMDD is incremental-backup date.
second HHMMSS.SSSSSS is incremental-backup time. HH is in 24 hour time format.
'no' indicates that this is not an upload-completion indicator AWS S3 object.

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name)_YYYYMMDD_HHMMSS.SSSSSS_YYYYMMDD_HHMMSS.SSSSSS_yes_(backup source unique name)

first YYYYMMDD is full-backup date.
first HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
second YYYYMMDD is incremental-backup date.
second HHMMSS.SSSSSS is incremental-backup time. HH is in 24 hour time format.
'yes' indicates that this is an upload-completion indicator AWS S3 object.

2.2.1.2. DEPRECATED [12/17/2020 8:54 PM CST]

[10/30/2020 1:10 PM CST]

[10/22/2020 7:48 PM CST] the following AWS S3 backup object name formats enable easy backup AWS S3 bucket querying.

[10/30/2020 11:54 AM CST] wait a minute. I need to first resolve the issue of a backup source unique name in the backup AWS S3 object name containing all or part of "incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)". I think preventing including "incremental-backup " in code is the best way to go. (not needed now, for my own use. needed later in . actually, a simple solution is making the backup AWS S3 object name uniform. I'll do that now.
https://stackoverflow.com/questions/1976007/what-characters-are-forbidden-in-windows-and-linux-directory-names

search "AWS S3 object name unallowed characters" online.
Amazon S3 Bucket Naming Requirements
The bucket name can be between 3 and 63 characters long, and can contain only lower-case characters, numbers, periods, and dashes.
Each label in the bucket name must start with a lowercase letter or number.
More items...
Amazon S3 Bucket Naming Requirements - AWS CloudTrail
docs.aws.amazon.com › awscloudtrail › latest › userguide

search "AWS S3 object name length limit" online.
https://stackoverflow.com/questions/6870824/what-is-the-maximum-length-of-a-filename-in-s3
As follows from the Amazon documentation,
These names are the object keys. The name for a key is a sequence of Unicode characters whose UTF-8 encoding is at most 1024 bytes long.
The max filename length is 1024 characters. (NOTE [10/30/2020 12:23 PM CST] so it can contain any Windows directory path [e.g. shadow-mounted directory path])

[10/30/2020 1:07 PM CST] do I want to support restoring data backup from multiple backup-source locations, on the same computer? for now, probably no. but it's not that difficult to do, so why not do it now? I'll probably do it. (FINAL DECISION [10/30/2020 1:18 PM CST] I'll do it.)

[10/30/2020 12:22 PM CST] RESTRICTIONS
a unique backup-source location name cannot contain colon.

<u>full-backup AWS S3 backup object name format</u>
(a unique backup-source location name):YYYYMMDD:HHMMSS.SSSSSS:0:0.0:no:(backup source unique name).zip.###

YYYYMMDD is full-backup date.
HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.

0:0.0 indicates this is full backup, not incremental backup.
'no' indicates that this is not an upload-completion indicator AWS S3 object.

<u>full-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name):YYYYMMDD:HHMMSS.SSSSSS:0:0.0:yes:(backup source unique name)

YYYYMMDD is full-backup date.
HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
0:0.0 indicates this is full backup, not incremental backup.
'yes' indicates that this is an upload-completion indicator AWS S3 object.

<u>incremental-backup AWS S3 backup object name format</u>
(a unique backup-source location name):YYYYMMDD:HHMMSS.SSSSSS:YYYYMMDD:HHMMSS.SSSSSS:no:(backup source unique name).zip.###

first YYYYMMDD is full-backup date.
first HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
second YYYYMMDD is incremental-backup date.
second HHMMSS.SSSSSS is incremental-backup time. HH is in 24 hour time format.
'no' indicates that this is not an upload-completion indicator AWS S3 object.

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name):YYYYMMDD:HHMMSS.SSSSSS:YYYYMMDD:HHMMSS.SSSSSS:yes:(backup source unique name)

first YYYYMMDD is full-backup date.
first HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
second YYYYMMDD is incremental-backup date.
second HHMMSS.SSSSSS is incremental-backup time. HH is in 24 hour time format.
'yes' indicates that this is an upload-completion indicator AWS S3 object.

2.2.1.3. DEPRECATED [10/30/2020 1:10 PM CST]

[10/30/2020 12:20 PM CST]

[10/22/2020 7:48 PM CST] the following AWS S3 backup object name formats enable easy backup AWS S3 bucket querying.

[10/30/2020 11:54 AM CST] wait a minute. I need to first resolve the issue of a backup source unique name in the backup AWS S3 object name containing all or part of "incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)". I think preventing including "incremental-backup " in code is the best way to go. (not needed now, for my own use. needed later in . actually, a simple solution is making the backup AWS S3 object name uniform. I'll do that now.
https://stackoverflow.com/questions/1976007/what-characters-are-forbidden-in-windows-and-linux-directory-names

search "AWS S3 object name unallowed characters" online.
Amazon S3 Bucket Naming Requirements
The bucket name can be between 3 and 63 characters long, and can contain only lower-case characters, numbers, periods, and dashes.
Each label in the bucket name must start with a lowercase letter or number.
More items...
Amazon S3 Bucket Naming Requirements - AWS CloudTrail
docs.aws.amazon.com › awscloudtrail › latest › userguide

search "AWS S3 object name length limit" online.
https://stackoverflow.com/questions/6870824/what-is-the-maximum-length-of-a-filename-in-s3
As follows from the Amazon documentation,
These names are the object keys. The name for a key is a sequence of Unicode characters whose UTF-8 encoding is at most 1024 bytes long.
The max filename length is 1024 characters. (NOTE [10/30/2020 12:23 PM CST] so it can contain any Windows directory path [e.g. shadow-mounted directory path])

[10/30/2020 1:07 PM CST] do I want to support restoring data backup from multiple backup-source locations, on the same computer? for now, probably no.

[10/30/2020 12:22 PM CST] RESTRICTIONS
a unique backup-source location name cannot contain colon.

<u>full-backup AWS S3 backup object name format</u>
(a unique backup-source location name):YYYYMMDD:HHMMSS.SSSSSS:0:0.0:no:(backup source unique name).zip.###

YYYYMMDD is full-backup date.
HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
0:0.0 indicates this is full backup, not incremental backup.
'no' indicates that this is not an upload-completion indicator AWS S3 object.

<u>full-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name):YYYYMMDD:HHMMSS.SSSSSS:0:0.0:yes:(backup source unique name)

YYYYMMDD is full-backup date.
HHMMSS.SSSSSS is full-backup time. HH is in 24 hour time format.
0:0.0 indicates this is full backup, not incremental backup.
'yes' indicates that this is an upload-completion indicator AWS S3 object.

<u>incremental-backup AWS S3 backup object name format</u>
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name).zip.###

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name)_upload_completed

2.2.1.4. DEPRECATED [10/30/2020 12:20 PM CST]
[10/30/2020 12:02 PM CST]

[10/22/2020 7:48 PM CST] the following AWS S3 backup object name formats enable easy backup AWS S3 bucket querying.

[10/30/2020 11:54 AM CST] wait a minute. I need to first resolve the issue of a backup source unique name in the backup AWS S3 object name containing all or part of "incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)". I think preventing including "incremental-backup " in code is the best way to go. (not needed now, for my own use. needed later in . actually, a simple solution is making the backup AWS S3 object name uniform. I'll do that now.
https://stackoverflow.com/questions/1976007/what-characters-are-forbidden-in-windows-and-linux-directory-names

search "AWS S3 object name unallowed characters" online.
Amazon S3 Bucket Naming Requirements
   The bucket name can be between 3 and 63 characters long, and can contain only lower-case characters, numbers, periods, and dashes.
   Each label in the bucket name must start with a lowercase letter or number.
More items...
Amazon S3 Bucket Naming Requirements - AWS CloudTrail
docs.aws.amazon.com › awscloudtrail › latest › userguide

search "AWS S3 object name length limit" online.
https://stackoverflow.com/questions/6870824/what-is-the-maximum-length-of-a-filename-in-s3
As follows from the Amazon documentation,

These names are the object keys. The name for a key is a sequence of Unicode characters whose UTF-8 encoding is at most 1024 bytes long.
The max filename length is 1024 characters.

full-backup AWS S3 backup object name format
(a unique backup-source location name):::full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_false_00000000_000000.000000_(backup source unique name).zip.###

full-backup upload completion indicator AWS S3 backup object name format
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_false_00000000_000000.000000_upload_completed_(backup source unique name)

incremental-backup AWS S3 backup object name format
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name).zip.###

incremental-backup upload completion indicator AWS S3 backup object name format
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name)_upload_completed

2.2.1.5. DEPRECATED [10/30/2020 12:02 PM CST]

[10/28/2020 3:20 PM CST]

[10/22/2020 7:48 PM CST] the following AWS S3 backup object name formats enable easy backup AWS S3 bucket querying.

full-backup AWS S3 backup object name format
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name).zip.###

full-backup upload completion indicator AWS S3 backup object name format
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name)_upload_completed

incremental-backup AWS S3 backup object name format (a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name).zip.###

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-source location name)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique name)_upload_completed

2.2.1.6. DEPRECATED [10/28/2020 3:20 PM CST]

[10/22/2020 7:50 PM CST]

[10/22/2020 7:48 PM CST] the following AWS S3 backup object name formats enable easy backup AWS S3 bucket querying.

<u>full-backup AWS S3 backup object name format</u>
(a unique backup-set prefix)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag).zip.###

<u>full-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-set prefix)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag)_upload_completed

<u>incremental-backup AWS S3 backup object name format</u>
(a unique backup-set prefix)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag).zip.###

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-set prefix)_full-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag)_upload_completed

2.2.1.7. DEPRECATED [10/22/2020 7:50 PM CST]

[10/15/2020 10:09 AM CST]
<u>full-backup AWS S3 backup object name format</u>
(a unique backup-set prefix)_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag)_full-backup.zip.###

<u>full-backup upload completion indicator AWS S3 backup object name format</u>

(a unique backup-set prefix)_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag)_full-backup_upload_completed

<u>incremental-backup AWS S3 backup object name format</u>
(a unique backup-set prefix)_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_ (backup source unique prefix or tag)_full-backup-date-time-YYYYMMDD-HHMMSS.SSSSSS.zip.###

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(a unique backup-set prefix)_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_ (backup source unique prefix or tag)_full-backup-date-time-YYYYMMDD-HHMMSS.SSSSSS_upload_completed

DEPRECATED BELOW [10/15/2020 10:09 AM CST]
[10/14/2020 8:22 AM CST]

<u>full-backup AWS S3 backup object name format</u>
(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag)_full-backup.zip.###

<u>full-backup upload completion indicator AWS S3 backup object name format</u>
(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_(backup source unique prefix or tag)_full-backup_upload_completed

<u>incremental-backup AWS S3 backup object name format</u>
(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_ (backup source unique prefix or tag)_full-backup-date-time-YYYYMMDD-HHMMSS.zip.###

<u>incremental-backup upload completion indicator AWS S3 backup object name format</u>
(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_ (backup source unique prefix or tag)_full-backup-date-time-YYYYMMDD-HHMMSS_upload_completed

DEPRECATED BELOW [10/14/2020 8:22 AM CST]
[10/10/2020 1:09 PM CST]
<u>full-backup AWS S3 backup object name format</u>
(unique prefix or tag)_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_full-backup.zip.###

<u>incremental-backup AWS S3 backup object name format</u>

(unique prefix or tag)_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)_full-backup-date-time-YYYYMMDD-HHMMSS.zip.###

2.2.2. log content format

[10/15/2020 2:40 PM CST]

[10/20/2020 2:37 PM CST] everything.

[10/21/2020 7:21 AM CST] (with section heading at the start of each section below)
user-executed AAM Auto Backup command parameters.
(if applicable) file lock acquisition failure.
Backup execution starting time and date.
the latest completed-upload AWS S3 full and incremental backup dates and times.
the directories to backup, the directories to shadow mount, and folders and files to exclude.
(if applicable) the ignored the directories and non-wildcard files that do not exist.
(if applicable) AAM Auto Backup instruction plain text file reading error.
(if applicable) AWS S3 backup files directory creation error.
(if applicable) a temporary plain text file creation error.
(if applicable) ShadowSpawn execution error.
(if applicable) a temporary plain text file reading error.
ShadowSpawn outputs.
(if applicable) ShadowSpawn execution error.
(if applicable) AWS upload error.
(if applicable) AWS S3 backup files directory deletion error.
(if applicable) temporary output files deletion error.
(if applicable) email report sending error.

2.2.3. email report title formats

2.2.3.1. backup email report title format

[10/21/2020 9:27 AM CST]

SUCCESS/FAILURE, YYYYMMDD HH:MM:SS.SSSSS (full|incremental) backup!

(optional [10/22/2020 5:46 PM CST] time zone after the time in the title)

2.2.3.2. restoration email report title format

[10/22/2020 5:46 PM CST]

SUCCESS/FAILURE, YYYYMMDD HH:MM:SS.SSSSS data restoration!

(optional [10/22/2020 5:46 PM CST] time zone after the time in the title)

2.2.4. email report content formats

2.2.4.1. backup email report content formats

[10/15/2020 2:40 PM CST]

[10/21/2020 9:35 AM CST]
(backup destination types. local, AWS S3, or both.)

(if applicable) (the latest completed-upload AWS S3 full and incremental backup dates and times.)

~~(AWS S3 backup type. full or incremental.)~~ (in the email title, if applicable)

"For more info, refer to the local backup log."
(nothing else!)

[10/20/2020 2:37 PM CST] no ShadowSpawn execution log.
think and decide on not including the ShadowSpawn execution log in the email reports. no good will ever come out of emailing all the ShadowSpawn.exe, Robocopy and 7z.exe outputs in unencrypted emails. also, for the full backups, the Robocopy and maybe 7z.exe outputs will be too big to email—so emailing those outputs should be avoided. I'm very heavily leaning toward including the ShadowSpawn execution log in the AAM Auto Backup log only. think and decide on this, document the decision, and implement the decision!!!! [10/19/2020 8:30 AM CST]

2.2.4.2. restoration email report content formats

[10/22/2020 5:48 PM CST]

Unique Backup-Set Prefix: xx

(the latest completed-upload AWS S3 full and incremental backup dates and times, used in the data restoration.)

"For more info, refer to the local backup log."
(nothing else!)

2.2.5. ======DEPRECATED BELOW [10/20/2020 11:16 AM CST]====

~~[10/14/2020 10:09 AM CST]~~
~~AAM Auto Backup creates and updates a plain text file in its program source code file directory, for keeping the latest AWS S3 full and incremental backup dates and times, that is used in deciding which files and directories to incrementally back up to AWS S3 Deep Glacier.~~

2.2.6. last AWS S3 backup dates and times plain text file content

[10/15/2020 2:41 PM CST]
last AWS S3 full backup: (date in YYYYMMDD) (time in HHMMSS.SSSSSS, 24 hour time format)
last AWS S3 incremental backup: (date in YYYYMMDD) (time in HHMMSS.SSSSSS, 24 hour time format)

2.3. Intermediate Outputs

During a backup operation, AAM Auto Backup produces one or more of the following intermediate output files that will be deleted before the backup operation ends. That is, an AAM Auto Backup operation includes intermediate-outputs cleanup.

The shadow-copy mounted directory info plain text file, that contains the inclusion and exclusion information about the directory that will be shadow-copy mounted.

The dynamically generated batch file for performing the full backup to a local or network-attached data storage, that contains the cd command, the dynamically generated Robocopy command, the constant or static 7z.exe command.

The dynamically generated batch file for performing the incremental backup to a local or network-attached data storage, that contains the cd command, the dynamically generated Robocopy command, the constant or static 7z.exe command.

The dynamically generated 7z.exe list file for performing the full backup to AWS S3 Glacier Deep Archive.

The dynamically generated 7z.exe list file for performing the incremental backup to AWS S3 Glacier Deep Archive.

The dynamically generated one or more of the AWS S3 backup files in the multi-volume encrypted ZIP file format.

2.3.1. shadow-copy mounted directory info plain text file content (shadow_copy_instruction.txt)

2.3.1.1. [11/30/2020 2:14 PM CST]

list of included folders (the shadow-mounted drive folder absolute paths, not the backup-source folder absolute paths. no relative folder paths. no wild cards. [11/29/2020 1:56 PM CST]) (must not end in path separator) (for use by 7z.exe)

list of user-excluded folders (the shadow-mounted drive folder absolute paths, and folder names. no backup-source folder absolute paths; no relative folder paths of any kind, in the shadow-mounted drive or in the backup source directory. wild cards allowed only in directory names, not in folder paths. each entry must not end with path separator. if no entry in the list, this should be an empty line. [11/29/2020 1:56 PM CST]) (for use by Robocopy and 7z.exe)

list of user-excluded files (the shadow-mounted drive file absolute paths, and file names. no backup-source file absolute paths; no relative file paths of any kind, in the shadow-mounted drive or in the backup source directory. wild cards allowed only in file names, not in file paths. each entry must not end with path separator. if no entry in the list, this should be an empty line or no entry in the file. [11/29/2020 1:56 PM CST]) (for use by Robocopy and 7z.exe)

[11/30/2020 2:17 PM CST]
the following three data variables are built before the shadowspawn.exe execution, for creating the shadow-copy mounted directory info plain text file content above.
self._str_dictShadowMountDirectoryFullPathsToIncludedDirectoryLists
self._str_dictShadowMountDirectoryFullPathsToUserExcludedDirectoryLists
self._str_dictShadowMountDirectoryFullPathsToUserExcludedFileLists
None of the above three data variables will contain a full path that doesn't exist in the shadow-mounted drive.

[11/30/2020 2:16 PM CST]
the shadowspawn execution function generates
self._str_listShadowMountedDriveNonIncludedDirectoryFullPaths, by scanning the shadow-mounted drive contents, to use in building the Robocopy command.

list of non-included folders (the shadow-mounted drive folder absolute paths, not the backup-source folder absolute paths. no relative folder paths. no wild cards. [11/30/2020 10:14 AM CST]) (must not end in path separator) (for use in the Robocopy command)

2.3.1.2. DEPRECATED [11/30/2020 2:14 PM CST]

[11/30/2020 10:10 AM CST]

TO RESOLVE [11/30/2020 10:39 AM CST] hang on. hang on! should I generate these lists in the shadow-mount call handler, using the shadow-mounted drive contents, to be more exact? think on that, decide on that, and then implement the decision! if this is implemented, there will be no need to generate shadow_copy_instruction.txt; but there will be redundant processing of the backup instruction file! I don't want to do the redundant backup instruction file processing! but I want accurate backup, and less coding; so I may implement this after all!)

([11/30/2020 10:45 AM CST] I knew this would happen! even more design update! in all likely chance, I'll do it, because it feels right!!!!)
([11/30/2020 10:58 AM CST] I just searched "can mac do shadow copy?" online. it seems like Mac does not need anything special for copying open file; you just do it—as it should be the case.)
(POTENTIAL SOLUTION [11/30/2020 11:00 AM CST] execute a Robocopy command per included directory with the user-excluded directories and files, instead of executing a single Robocopy command on the entire shadow-mounted directory. do not use shadow_copy_instruction.txt. process the backup instruction file for each shadowspawn call. this has redundant processing of the same text file, and multiple executions of Robocopy. I don't like the waste, so I won't do it.)
[11/30/2020 12:45 PM CST] How about not using Robocopy? Using xcopy per included directory?
[11/30/2020 12:45 PM CST] I'm leaning toward using the shadow copy instruction file, and executing robocopy per included directory, and using all of user-excluded folders and files in every robocopy execution--to reduce my coding time and effort. No reason to complicate the problem that I must solve in code (for some performance gain in the number of Robocopy commands executed). In all likely chance, I'll implement the above.
POTENTIAL FINAL SOLUTION [11/30/2020 12:46 PM CST] in all honesty, I want to execute Robocopy only once per backup source or per shadow-mounted directory, because that feels right. however, I do not want any redundant text file content processing. in all likely chance, I'll use shadow_copy_instruction.txt with the following three data variables that are built before the shadowspawn.exe execution.
self._str_dictShadowMountDirectoryFullPathsToIncludedDirectoryLists
self._str_dictShadowMountDirectoryFullPathsToUserExcludedDirectoryLists
self._str_dictShadowMountDirectoryFullPathsToUserExcludedFileLists
None of the above three data variables will contain a full path that doesn't exist in the shadow-mounted drive.
I'll generate str_listShadowMountDriveNonIncludedDirectoryFullPaths, by scanning the shadow mount drive contents, in the shadowspawn execution function to use in building the Robocopy command.
the above seems to be the most efficient and proper solution. I'll consider implementing it.
the above feels right, because it does everything I want—i.e. avoiding non-included directory exclusion failure in Robocopy, and no redundant text file processing—while being the most computationally efficient. I may implement the above. unlike before, I feel like the above is the final design change and update I will make.
FINAL DECISION [11/30/2020 1:40 PM CST] I'm implementing the above. I will start the document update and the implementation when I feel like it, when my brain is rested and recharged.

list of included folders (the shadow-mount drive folder absolute paths, not the backup-source folder absolute paths. no relative folder paths. no wild cards. [11/29/2020 1:56 PM CST]) (must not end in path separator) (for use by 7z.exe) list of non-included folders (the shadow-mount drive folder absolute paths, not the backup-source folder absolute paths. no relative folder paths. no wild cards. [11/30/2020 10:14 AM CST]) (blank line if no entry.) (must not end in path separator) (for use by Robocopy)

list of user-excluded folders (each entry must end with path separator. the shadow-mount drive folder absolute paths, and folder names. no backup-source folder absolute paths; no relative folder paths of any kind, in the shadow-mount drive or in backup source. wild cards allowed only in directory names, not in folder paths. if no entry in the list, this should be an empty line. [11/29/2020 1:56 PM CST]) (must not end in path separator) (for use by Robocopy and 7z.exe)

list of user-excluded files (each entry must not end with path separator. the shadow-mount drive file absolute paths, and file names. no backup-source file absolute paths; no relative file paths of any kind, in the shadow-mount drive or in backup source. wild cards allowed only in file names, not in file paths. if no entry in the list, this should be an empty line or no entry in the file. [11/29/2020 1:56 PM CST]) (for use by Robocopy and 7z.exe)

TO RESOLVE [11/29/2020 2:03 PM CST] (wait a minute! do I want to use all of the 'list of folders to exclude' in the 7z.exe non-recursive and recursive list files? since the 'list of folders to include' is used by the inclusion 7z.exe list file, the 'list of folders to exclude' includes absolute folder paths that are already excluded. what do I do about that?)
TO ANSWER [11/29/2020 2:39 PM CST] do I want two separate folder exclusion lists? that would certainly complicate coding. maybe not too much; an empty line can be an empty entry in shadow_copy_instruction.txt.
NOTE [11/29/2020 2:47 PM CST] I'm heavily leaning toward computing efficiency. I don't want to dump on 7z.exe a work that it doesn't have to do. so, two separate exclusion lists? one for Robocopy only (system-specified folder exclusion list, or shadow-mounted folder exclusion list?), and another common (for both Robocopy and 7z.exe) (user-specified folder exclusion list)?
NOTE [11/29/2020 3:05 PM CST] currently, in the AAM Auto Backup source code, self._str_listUserSpecifiedDirectoriesToExclude is not used to exclude directories. I need to fix that.
TO ANSWER [11/29/2020 3:11 PM CST] so, build and use the following two data variables?
~~self._str_dictShadowMountDirectoryFullPathsToSystemExclusionDirectoryLists~~
self._str_dictShadowMountDirectoryFullPathsToNonIncludedDirectoryLists (I think this is better than the above)
~~self._str_dictShadowMountDirectoryFullPathsToUserExclusionDirectoryLists~~
~~self._str_dictShadowMountDirectoryFullPathsToExclusionDirectoryLists~~
~~self._str_dictShadowMountDirectoryFullPathsToUserExclusionDirectoryLists~~
self._str_dictShadowMountDirectoryFullPathsToUserExcludedDirectoryLists
[11/29/2020 3:51 PM CST] so use the two above data variables, and both non-included directory list and exclusion directory list in shadow_copy_instruction.txt? very likely so?
NOTE [11/29/2020 4:06 PM CST] yeah, I'm leaning toward implementing the above.
[11/29/2020 9:59 PM CST] yeah, in all likely chance, I will implement the above. I don't have an objection.

FINAL DECISION [11/30/2020 10:09 AM CST] **I am implementing the above.**
FINAL DECISION [11/30/2020 10:17 AM CST] I'm also using self._str_dictShadowMountDirectoryFullPathsToIncludedDirectoryLists.

2.3.1.3. DEPRECATED [11/30/2020 10:10 AM CST]

[11/29/2020 1:49 PM CST]

list of folders to include (the shadow-mount drive folder absolute paths, not the backup-source folder absolute paths. no relative folder paths. no wild cards. [11/29/2020 1:56 PM CST]) (for use by 7z.exe)

list of folders to exclude (each entry must end with path separator. the shadow-mount drive folder absolute paths, and folder names. no backup-source folder absolute paths; no relative folder paths of any kind, in the shadow-mount drive or in backup source. wild cards allowed only in directory names, not in folder paths. if no entry in the list, this should be an empty line. [11/29/2020 1:56 PM CST]) (for use by Robocopy and 7z.exe)

list of files to exclude (each entry must not end with path separator. the shadow-mount drive file absolute paths, and file names. no backup-source file absolute paths; no relative file paths of any kind, in the shadow-mount drive or in backup source. wild cards allowed only in file names, not in file paths. if no entry in the list, this should be an empty line or no entry in the file. [11/29/2020 1:56 PM CST]) (for use by Robocopy and 7z.exe)

2.3.1.4. DEPRECATED [11/29/2020 1:49 PM CST]

list of folders to include (for 7z.exe) (no wild card support. the shadow-mount drive folder absolute paths, not the backup-source folder absolute paths. no relative folder paths. [11/28/2020 2:24 PM CST])

list of folders to exclude (for Robocopy [and 7z.exe also, it seems]) (wild card support, in any directory name. the shadow-mount drive folder absolute paths, folder relative paths in the shadow-mount drive, or just folder names. no backup-source folder absolute path. [11/28/2020 2:24 PM CST])

list of files to exclude (for 7z.exe and Robocopy) (wild card support, in any directory and file name. the shadow-mount drive file absolute paths, file relative paths in the shadow-mount drive, or just file names. no backup-source file absolute path. [11/28/2020 2:24 PM CST])

2.4. [UPDATE] wild cards support in backup inclusion and exclusion directory and file names

[11/28/2020 3:50 PM CST]

1.1.1.1. make sure that the code and comments on self._str_dictShadowMountDirectoryFullPathsToExclusionDirectoryLists and self._str_dictShadowMountDirectoryFullPathsToExclusionFileLists are right. right now, I sense that they aren't completely right. check the code that uses those two variables to make sure that the building and the using of those two data variables are perfectly synchronized. [11/28/2020 2:02 PM CST] (no wild card support in directory names? think on that, decide on that, and then implement the decision!) ([11/28/2020 3:06 PM CST] do I want to document and test more Robocopy commands in the "backup directories computation algorithms" section of the AAM Auto Backup document, for supporting wild cards?) (NOTE [11/28/2020 3:22 PM CST] I want to support wild cards in both directory and file names; I'll definitely support them in . however, doing so would take more coding. I need to decide whether to do that coding now or later. the policy is never to wait. in order to decide on whether to do coding now, I first need to find out whether Robocopy supports using wild cards in exclusion directory and file names.)

1.1.1.1.1. devise, document, and perform additional Robocopy commands for finding out whether Robocopy supports wild cards in exclusion directory and file names. [11/28/2020 3:34 PM CST] document the tests and results in the "backup directories computation algorithms" section of the AAM Auto Backup document. [11/28/2020 3:48 PM CST]

1.1.1.1.2. make the final decision and document it. [11/28/2020 3:36 PM CST]

1.1.1.1.2.1. update the "[UPDATE] wild cards support in backup inclusion and exclusion directory and file names" section in the AAM Auto Backup document as needed. [11/28/2020 3:53 PM CST]

1.1.1.1.2.2. update the "How to create the AAM Auto Backup instruction plain text file" section in the AAM Auto Backup document as needed. [11/28/2020 3:36 PM CST]

1.1.1.1.2.3. update the "shadow-copy mounted directory info plain text file content (shadow_copy_instruction.txt)" section in the AAM Auto Backup document as needed. [11/28/2020 3:36 PM CST]

1.1.1.1.2.4. update the "Backup-source data variables" section in the AAM Auto Backup source code as needed. [11/28/2020 3:36 PM CST]

1.1.1.1.2.5. update the "ShadowSpawn.exe execution data variables" section in the AAM Auto Backup source code as needed. [11/28/2020 3:36 PM CST]

1.1.1.1.3. update the applicable source code as needed. [11/28/2020 3:43 PM CST]

2.4.1. Robocopy exclusion capability conclusions

[11/28/2020 4:47 PM CST]

Robocopy supports only directory name and full/absolute directory path exclusion, no relative path exclusion.

Robocopy has no wild card support in exclusion directory full/absolute path. wild card support only in directory names, not absolute directory paths.

Robocopy supports only file name and full/absolute file path exclusion, no relative file path exclusion.

Robocopy has no wild card support in exclusion file full/absolute path. wild card support only in file names, not absolute file paths.

2.4.2. 7z.exe exclusion capability conclusions

[11/29/2020 9:37 AM CST]

7z.exe supports directory name and full/absolute directory path exclusion.

7z.exe supports wild cards in directory names.

7z.exe supports file name and full/absolute file path exclusion.

7z.exe supports wild cards in file names.

2.4.3. final software design decision

FINAL DECISION [11/29/2020 10:39 AM CST] **support everything that Robocopy supports.** namely, directory name and full directory path exclusion, wild card in directory name, file name and full file path exclusion, wild card in file name.

[11/28/2020 4:33 PM CST]
~~no wild card support in exclusion directory names (at least for now), since Robocopy does not support it.~~

For local backup, I won't do more than what Robocopy supports.

For AWS backup, I won't do more than what Robocopy supports.

I'm heavily leaning toward supporting everything that Robocopy supports—namely, directory name and full directory path exclusion, wild card in directory name, file name and full file path exclusion, wild card in file name.

UPDATE [11/28/2020 5:16 PM CST] again, I'm very heavily leaning toward supporting everything that Robocopy supports.

UPDATE [11/29/2020 7:57 AM CST]
I'm leaning toward simplifying the directory and file inclusion and exclusion in AAM Auto Backup. No wild cards support in the inclusion directory paths. Wild cards support in the exclusion directory names, only if 7z.exe supports it. Update the code to exit the program if there is a full directory or file path with a wild card.

FINAL DECISION 1 [11/29/2020 10:07 AM CST]
[Producer] [Auto] Backup v1.0 in , a renamed version of AAM Auto Backup, will support everything that Robocopy supports—namely, directory name and full directory path exclusion, wild card in directory name, file name and full file path exclusion, wild card in file name. also, Producer Backup v1.0 will support running on Mac using Mac software programs.

NOTE [11/29/2020 10:16 AM CST] I may or may not hire a software engineer to update AAM Auto Backup v1.0 to Producer Backup v1.0.

NOTE [11/29/2020 10:21 AM CST] I probably won't implement the AAM Auto Backup features in directly, in C++ code: I don't see an incentive in doing that, since it is reinventing a wheel—other software programs that are available already should be used for auto backup. AAM Auto Backup features will be supported using a separate program ([Producer] [Auto] Backup v1.0) and a free process package. however, will support storing its files in the Cloud (a Cloud Sync feature), when a monthly fee is paid, for accessing from all devices.

[11/29/2020 10:47 AM CST] [Producer] [Auto] Backup v1.0 will probably produce and use Robocopy job files to handle an unlimited number of exclusion directories and files.

[11/29/2020 10:38 AM CST] Windows edition will be Windows Shadow Copy aware or support Windows Shadow Copying of its files.

NOTE [11/29/2020 10:26 AM CST] If I or a hired software engineer implement(s) the directory name wild card support later, it will be much more expensive time-wise, because the entire software program code and document must be re-studied and recalled. if I implement the directory name wild card support now, it will be very cheap time-wise, less than six hours. so, in all likely chance, I'll do what is cheap—I'll implement the directory name wild card support now.

FINAL DECISION [11/29/2020 10:39 AM CST] **support everything that Robocopy supports.**

3. AWS S3 Glacier Deep Archive usage costs

[10/13/2020 7:40 AM CST]
https://aws.amazon.com/s3/pricing/

storage cost

S3 Glacier Deep Archive ** - For long-term data archiving that is accessed once or twice in a year and can be restored within 12 hours (180 days or six months mandatory storage and payment period)
All Storage / Month: $0.00099 per GB

for 1GB of data: $0.00594 (for six months of data storage)
for 1TB of data: $5.94 (for six months of data storage)

S3 Glacier ** - For long-term backups and archives with retrieval option from 1 minute to 12 hours (90 days or six months mandatory storage and payment period)
All Storage / Month: $0.004 per GB
for 1GB of data: $0.012 (for three months of data storage)
for 1TB of data: $12 (for three months of data storage)

bulk retrieval cost
$0.0025 per GB

for 1GB of data: $0.0025
for 1TB of data: $2.5

regular retrieval cost
$0.02 per GB

for 1GB of data: $0.02
for 1TB of data: $20

data transfer cost
Data Transfer OUT From Amazon S3 To Internet: $0.09 per GB

for 1GB of data: $0.09
for 1TB of data: $90

4. Backup files upload time
[12/21/2020 10:56 AM CST]

measure the upload speed, and calculate and document how long it takes to upload 4GByte of data. [12/21/2020 10:46 AM CST]

79.1 Mbps download
20.9 Mbps upload (2.6125 megabytes per second)

382.7751196 seconds, or 6.379585327 minutes to upload 1 GByte of data.
25.51834131 minutes to upload 4 GByte of data.
127.5917065 minutes to upload 20 GByte of data.

Fios Gigabit Internet Connection - Fiber Optic ... - Verizon
www.verizon.com › home › fios-gigabit-connection
Fios offers the fastest Speeds up 940/880 Mbps available in select areas. , most reliable internet with bandwidth for multiple devices and 4K- compatible TV picture quality.

https://fiber.google.com/

5. Using main work computers with boot password and directories encryption

[10/15/2020 9:00 AM CST]
Get main work desktop and laptop computers, with directory encryption feature.

Keep the user data directory, and the main work data replication directory on the USB hard drive, encrypted at all times. Do not encrypt the entire Windows-user Documents directory; rather, create a sub-directory in the user Documents directory, named 'secure', and encrypt the secure sub-directory in the Documents directory (e.g. C:\Users\Allen\Documents\secure); encrypt other sensitive-data sub-directories in the Windows-user Documents directory, such as the source code directories.

Password-protect the booting of every main work computer—the password should be unique and different from the data restoration passwords. Look into Secure Boot.

When switching to another USB hard drive, delete all the main work data on the previous USB hard drive.

search "Dell laptops with TPM 2.0 and InstantGo" online.

Look into directory encryption with a Mac laptop.

Note that Windows requires using the NTFS file format for directory encryption. As such, the USB hard drive cannot be used on both Windows and Mac with directory encryption. Pick one platform and commit to it. I'm leaning toward using Windows for the main work computer, both desktop and laptop, due to the wide availability of hardware devices and more software programs.

5.1. What TMP is

[10/15/2020 9:57 AM CST]
https://en.wikipedia.org/wiki/Trusted_Platform_Module

https://www.howtogeek.com/237232/what-is-a-tpm-and-why-does-windows-need-one-for-disk-encryption/
https://www.howtogeek.com/236719/whats-the-difference-between-bitlocker-and-efs-encrypting-file-system-on-windows/

5.2. Windows 10 Pro for Directory Encryption

[10/15/2020 8:57 AM CST]

https://www.microsoft.com/en-us/windows/windows-10-specifications
BitLocker Drive Encryption (available with Windows 10 Pro or Windows 10 Enterprise only) requires a Trusted Platform Module (TPM) 1.2 or higher and Trusted Computing Group (TCG)-compliant BIOS or UEFI. BitLocker can be used on devices without TPM, but you will need to save a startup key on a removable device such as a USB flash drive. TPM 2.0 and InstantGo support is required when you want to automatically encrypt the local drive when joining a device to Azure Active Directory (AAD). Check with your PC manufacturer to confirm if your device supports the correct TPM version and InstantGo for the scenario you want to enable.

https://superuser.com/questions/973905/how-to-encrypt-home-directory-on-windows-10-home/974109
File / Device encryption requires device support of a TPM 2.0 chipset and InstantGo. A 3rd party app would be required for file encryption in your situation as well since EFS is not supported in Windows 10 Home.
Further details can be found via Microsoft's Windows 10 Specifications:
http://www.microsoft.com/en-us/windows/windows-10-specifications

5.3. Computers with UEFI, Secure Boot, TPM 2.0, and InstantGo

[10/15/2020 9:09 AM CST]

search "Dell laptops with TPM 2.0 and InstantGo" online.
Which Dells have UEFI, Secure Boot, and TPM 2.0
https://www.dell.com/community/Windows-10/Which-Dells-have-UEFI-Secure-Boot-and-TPM-2-0/td-p/6101393

search "what is Secure Boot?" online.
https://www.howtogeek.com/116569/htg-explains-how-windows-8s-secure-boot-feature-works-what-it-means-for-linux/
Ubuntu, Fedora, Red Hat Enterprise Linux, and openSUSE currently support Secure Boot, and will work without any tweaks on modern hardware. There may be others, but these are the ones we’re aware of. Some Linux distributions are philosophically opposed to applying to be signed by Microsoft.

[10/15/2020 9:39 AM CST] before buying, message Dell to get the list of the desktop and laptop computers with UEFI, Secure Boot, TPM 2.0, and InstantGo. especially, ask if the highest end Alienware laptop and desktop computers support those needed data-security features.

5.4. How to check for UEFI

[10/15/2020 9:46 AM CST] in the booting screen, press the appropriate key to enter the BIOS/xx setup mode.

5.5. How to check for Secure Boot

[10/15/2020 9:47 AM CST]
search "how to check for Secure Boot" online.

5.6. How to check for TPM

[10/15/2020 9:47 AM CST]
search "how to check for TPM" online.

5.7. How to check for InstantGo

[10/15/2020 9:47 AM CST]
search "how to check for InstantGo" online.

II. AAM Auto Backup Installation and Configuring

[10/5/2020 9:09 PM CST]

1. How to configure AWS for AAM Auto Backup

[10/14/2020 9:49 AM CST]
For proper, maximum-cybersecurity operations, AAM Auto Backup requires a backup-only AWS account, a write-only AWS IAM user for data backup, and an admin AWS IAM user for data restoration.

[10/2/2020 5:59 AM CST]
For data backup, AAM Auto Backup requires an AWS IAM user with a write-only access to the backup AWS S3 bucket, and a permission to send emails via AWS SES (for sending backup report emails).

[10/14/2020 9:45 AM CST]

For data restoration, AAM Auto Backup requires an AWS IAM user with a read and restoration access to the backup AWS S3 Glacier Deep Archive bucket, and a permission to send emails via AWS SES (for sending restoration report emails).

UPDATE [10/19/2020 1:39 PM CST]
create the backup AWS S3 bucket as the admin user, so that its objects can be restored and downloaded without adding additional permissions.
UPDATE [10/19/2020 1:47 PM CST] actually, the admin user can upload to and download from the backup AWS S3 bucket created by the root user. no AWS S3 bucket creation by the admin user needed.

[10/7/2020 7:56 PM CST]
Refer to the "[Optional] Using AWS for one-way data backup" section in the Online Content Marketing Website Mastery audiovisual script book, for the instructions on setting up a backup-only AWS account, adding an AWS IAM user for uploading backup files to the backup AWS S3 bucket and sending report emails, and adding an admin AWS IAM user for data restoration from AWS S3.

check the log, and debug the email sending error. [12/21/2020 10:49 AM CST] the email sender address was verified on the main AWS account, not on the backup-only AWS account; that was why the email sending error occurred. think and resolve this issue, and document the solution.
in all likely chance, I probably need an IAM account on the main AWS account for sending emails only. look into verifying the email address on the backup-only AWS account still.
[12/21/2020 10:53 AM CST]
make sure that sending email works. [12/21/2020 11:49 AM CST]

[12/21/2020 12:48 PM CST] autoreporter@allenyoung.dev email address verification.
log into the backup-only AWS account.
navigate to the AWS SES console.
select the right region.
select "Email Addresses" on the left pane.
select "Verify a New Email Address".
enter autoreporter@allenyoung.dev, and select "Verify This Email Address".
log into the main AWS account with the email reception.
navigate to the AWS S3 console.
download the received email.
open the received email in Mozilla Thunderbird.
copy the link and paste on the relevant web browser to verify the email address.

[12/21/2020 1:24 PM CST] the above method requires verifying the recipient email address too.
maybe try to find a way to not do that. (maybe make the sender and the recipient both the same email address?)

TO DO [12/21/2020 1:14 PM CST] code update for .
when creating the AWS SES client object in the Python code, make sure to select the AWS region with the verified email address.
better yet, if possible, automate the email address verification via Python code.

2. How to install AAM Auto Backup

[10/7/2020 7:17 PM CST]
Execute the latest Python installer.

Install the AWS Python API by executing the following command in an admin Windows terminal.

```
pip install boto3
```

(the Boto3 installation reference: https://aws.amazon.com/sdk-for-python/)

Copy the AAM Auto Backup program file to a directory of your choice.

[10/18/2020 4:34 PM CST] install the FileLock Python module also
https://github.com/dmfrey/FileLock
https://pypi.org/project/filelock/

```
pip3 install filelock
```

[10/14/2020 10:03 AM CST]
7z.exe and ShadowSpawn.exe must be in the AAM Auto Backup program directory.

[10/14/2020 11:23 AM CST]
AAM Auto Backup executes ShadowSpawn.exe for backing up open and in-use files via the Windows Shadow Copy Service.

AAM Auto Backup executes 7z.exe for backup-files compression and decompression with encryption, for storing and restoring data backup to and from AWS S3 Glacier Deep Archive.

3. How to create the AAM Auto Backup instruction plain text file

3.1. file name format

[10/6/2020 1:37 PM CST]

the file name can be anything, but it should end in .txt to indicate that the file is a plain text file.

[10/6/2020 1:39 PM CST] the full path to this file is provided as an argument in the AAM Auto Backup command.

3.2. file form format

[10/14/2020 11:42 AM CST]
UTF-8. For Unicode directory and file names support.

3.3. file content format

3.3.1. [11/29/2020 10:52 AM CST]

```
(the list of the directories to backup.) [full or absolute paths
only.  no wild card support]

(the list of the directories to exclude [directory names and full
paths supported.  no relative directory path support.  wild cards
supported only in directory names, not in directory full paths.
an exclusion directory entry must end in \])

(the list of the files to exclude [file names and full paths
supported.  no relative file path support.  wild cards supported
only in file names, not in file full paths.  an exclusion file
entry must NOT end in \])
```

(exclude the MS Word temp files. ~WRL*.tmp. e.g. ~WRL3620.tmp)

AAM Auto Backup will factor and calculate each backup root directory (somehow). (I will create and document a Python algorithm for doing this! [10/7/2020 4:08 PM CST])

The full path of each backup root directory, with : and \ (or / on Mac and Linux) replaced with -, is used by AAM Auto Backup as the unique backup-source name in the backup AWS S3 objects. e.g. "c:\users\allen\documents\" will be "c--users-allen-documents-".

~~The list of the directories to exclude is for the rendered audiovisual files backup, for skipping backing up all the sub-directories of the 'audiovisual' directory that starts with -. [10/16/2020 12:08 PM CST]~~ Instead of using an AAM Auto Backup command parameter, the AAM Auto Backup instruction plain text file content is used for controlling the rendered audiovisual files backup. ~~([10/7/2020 8:25 AM CST] one or more arguments for rendered audiovisual files backup. (get this absolutely right! for now, I think only one directory full path as the argument will do, because the name of each first-level sub-directory can be checked for - at the start for deciding on auto-backup.))~~ ~~(test Robocopy for the folder omission. see if specifying wildcard in Robocopy command for excluding folders work in the way that serve my need. [10/16/2020 2:16 PM CST])~~

3.3.2. DEPRECATED [11/29/2020 10:52 AM CST]

[10/7/2020 4:01 PM CST]

```
 (the list of the directories to backup.) [no wild card support]

 (the list of the directories to exclude [wild card supported in
any directory name, must end in \][both relative and absolute
directory paths supported?  perform Robocopy test(s) and find
out!])

 (the list of the files to exclude [wild card supported in any
directory and file name, must not end in \][both file full paths
and file full names supported?  perform Robocopy test(s) and find
out!])
```

(exclude the MS Word temp files. ~WRL*.tmp. e.g. ~WRL3620.tmp)

AAM Auto Backup will factor and calculate each backup root directory (somehow). (I will create and document a Python algorithm for doing this! [10/7/2020 4:08 PM CST])

The full path of each backup root directory, with : and \ (or / on Mac and Linux) replaced with -, is used by AAM Auto Backup as the unique backup-source name in the backup AWS S3 objects. e.g. "c:\users\allen\documents\" will be "c--users-allen-documents-".

The list of the directories to exclude is for the rendered audiovisual files backup, for skipping backing up all the sub-directories of the 'audiovisual' directory that starts with -. [10/16/2020 12:08 PM CST] Instead of using an AAM Auto Backup command parameter, the AAM Auto Backup instruction plain text file content is used for controlling the rendered audiovisual files backup. ([10/7/2020 8:25 AM CST] one or more arguments for rendered audiovisual files backup. (get this absolutely right! for now, I think only one directory full path as the argument will do, because the name of each first-level sub-directory can be checked for - at the start for deciding on auto-backup.)) (test Robocopy for the folder omission. see if specifying wildcard in Robocopy command for excluding folders work in the way that serve my need. [10/16/2020 2:16 PM CST])

([10/7/2020 4:04 PM CST] now the above feels right! feels very right!!!!!!)

3.3.3. DEPRECATED [10/7/2020 4:01 PM CST]

[10/6/2020 1:23 PM CST]

(~~the user-specified unique name of the backup. this name will be in the compressed [backup] file name, and this name will be the name of the level-1 sub-directory of each restored backup in the restoration root directory.~~ actually, the full path of the backup root directory, with : and \ replaced with -, will be used as the unique backup name. e.g. "c:\users\allen\documents\" will be "c--users-allen-documents-".)
(the directory to backup [must be a full path that ends in \])
(the list of the sub-directories to exclude [must be a full path that ends in \])
(the list of the files to exclude [wildcard supported, must not end in \])

(the directory to backup [must be a full path that ends in \])
(the list of the sub-directories to exclude [must be a full path that ends in \])
(the list of the files to exclude [wildcard supported, must not end in \])

...

3.4. what happens when the AAM Auto Backup instruction plain text file content changes

3.4.1. [11/11/2020 7:14 PM CST] [11/13/2020 12:06 PM CST]

After changing the AAM Auto Backup instruction plain text file content, the user should manually delete the outdated and no-longer-needed folders in the local backup destination (if not using the option to auto-delete them via Robocopy, or Robocopy won't delete the obsolete folders).

Always performing auto-executed regular backup, with the "automatic force full backup" argument of the regular-backup command set to "yes", is highly recommended, in order to make sure that a full backup is automatically done if the AAM Auto Backup instruction plain text file content changes, and the user has not performed manual full backup after changing the AAM Auto Backup instruction plain text file content.

The user should perform regular backup manually with 'force full backup', immediately after changing the AAM Auto Backup instruction plain text file content, to create a full backup on AWS S3 that will include all the new folders and files.

If additional folders are included in a backup source (a common directory that is shadow mounted for backup), one or more files in those folders may not be included in the next incremental backup, depending on their creation and modification times; full backup after changing the AAM Auto Backup instruction plain text file content, is required to ensure that all files get backed up to AWS S3.

Changing the AAM Auto Backup instruction plain text file content will not result in AAM Auto Backup error, since AAM Auto Backup simply follows the user-specified AAM Auto Backup instruction in the file, and is unaware of the previous version of the backup instruction file.

Robocopy will perform correctly after changing the AAM Auto Backup instruction plain text file content.

If needed, another AWS S3 full-backup archive in the same month will be created by AAM Auto Backup. Refer to the "AWS S3 data archive file creation rules" section in Part III for more info.

decide whether to do the following now or later. (FINAL DECISION [11/11/2020 7:34 PM CST] in , add an argument to perform full backup if the AAM Auto Backup instruction plain text file modification or creation time is after the last AWS S3 backup time. [or do this now? no?] [I'm leaning toward doing it now. it seems very good.]) [11/11/2020 7:36 PM CST] (CONSIDERATION [11/12/2020 10:08 AM CST] is there value in coding this now? will I actually use it? it won't take much coding to implement this, so code it now? any decisive reason to wait until later? NOTE [11/12/2020 10:14 AM CST] there is already code to generate datetimeLastAwsS3FullBackup and datetimeLastIncrementalAwsS3FullBackup in the stage 3 of PerformShadowSpawnExecutedBackUpOperation(). it would be very easy to implement the force backup when the AAM Auto Backup instruction plain text file modification or creation time is after the last AWS S3 backup time. so, do it now or later? since it is very easy and fast to do it, why not do it now?) (UPDATE [11/12/2020 10:27 AM CST] I'm leaning toward implementing it, because it is very simple, easy, and fast to implement it.) (UPDATE [11/12/2020 10:30 AM CST] I'm very heavily leaning toward implementing this. just a few more lines of code. in all likely chance, I'll do it.) (FINAL DECISION [11/12/2020 10:39 AM CST] I'm doing it. it's so easy and fast to code it, so I'm doing it now!)

implement the automatic force full backup. [11/12/2020 10:46 AM CST]
update the document.
update the code.

UPDATE [11/12/2020 11:37 AM CST] when 'automatic force backup' is on, the backup mode is incremental-only, and the AAM Auto Backup instruction plain text file modification or creation time is after the last AWS S3 full backup time, error is displayed and the software program aborts the execution.

3.4.2. DEPRECATED [11/11/2020 7:14 PM CST]

[11/10/2020 5:35 PM CST]

The user does not need to do anything additional after changing the AAM Auto Backup instruction plain text file content—other than maybe manually deleting some folders in the local backup destination.

Also, if wanted, the user should perform regular backup manually with 'force full backup', immediately after changing the AAM Auto Backup instruction plain text file content, to create a full backup as needed.

Robocopy will perform correctly after changing the AAM Auto Backup instruction plain text file content.

If needed, more than one AWS S3 full-backup archive in the same month will be created by AAM Auto Backup. Refer to the "AWS S3 data archive file creation rules" section in Part III for more info.

3.4.3. ORIGINAL NOTES

completely resolve the issue of the content of the AAM Auto Backup instruction plain text file changing. update the AAM Auto Backup document as needed. [10/26/2020 7:50 AM CST]

implement 'force full backup' as needed. [11/10/2020 5:51 PM CST] first, review the existing code to find out what is being done, and what needs to be done. [11/10/2020 6:04 PM CST] (NOTE [11/10/2020 6:17 PM CST] doing full or incremental backup is decided in PerformShadowSpawnExecutedBackUpOperation(). to implement 'force full backup', I'll probably have to add a command argument to both regular backup and ShadowSpawn command arguments. very easy and simple to add that additional code.)

plan how to document every update in software design in the AAM Auto Backup document. document the plan somewhere appropriate in the AAM Auto Backup document. [10/26/2020 2:06 PM CST] (UPDATE [11/11/2020 6:02 AM CST] for now, I think only the "operation mode overview" sections should be updated.) (UPDATE [11/11/2020 6:40 AM CST] I probably need to update the "software process specs" part also.) (TO DO [11/11/2020 6:47 AM CST] go through the AAM Auto Backup document from the top to the bottom, check every section, and list below each section to update. afterward, update each applicable section.)

document every update in the AAM Auto Backup software design in the AAM Auto Backup document. [10/26/2020 1:06 PM CST]

4. How to configure Windows Task Scheduler to auto-run AAM Auto Backup everyday

[10/2/2020 5:59 AM CST]
this backup program should be auto-executed daily via the Windows Task Scheduler.

[11/27/2020 11:46 AM CST] the AAM Auto Backup command executed by Windows Task Scheduler should have force full backup off, and automatic force full backup on.

4.1. process spec

4.1.1. [3/29/2021 2:52 PM CST]

Create Task

General
Name: Daily AAM Auto Backup
Location: (cannot change this.)
select "Run whether user is logged on or not"
DO NOT check "Do not store password. The task will only have access to local computer resources." ("Run whether user is logged on or not" and storing the password is needed, so that when Python.exe is executed, no Windows terminal window is displayed. Refer to the "Windows 10 task execution failure debugging" sub-section below for more info.)
check "Run with highest privileges"
DO NOT check "Hidden" (this is for hiding task in the Windows Task Scheduler, not hiding the terminal window.)
Select "Windows 10" for "Configure for".

Triggers
Select New.
Configure a new trigger for daily triggering at 12:30am local time.

Actions
Select New.
Action: Start a program
Program/script (must be in double quotes): "C:\Users\Allen\AppData\Local\Programs\Python\Python39\python.exe" (use the Browser button to select python.exe.)
Add arguments (optional): `"(AAM_auto_backup.py full path)" regular-backup user "(python.exe full path)" 600 7200 "(backup log directory full path)" True "AAM Backup Report" "(backup report sender email address)" "(backup report recipient email address)" "(backup instructions text file full path)" "q" "local|AWS" "(backup destination directory full path)" False "(backup password)" "aam-data-backup" "main-work-computer" "(AWS secret key)" "(AWS secret key password)" "no" "yes"`
Start in (optional): (`AAM_auto_backup.py` program file full path) (this should not be in double quotes. the task won't execute if it is.)
Select OK.

Conditions
Uncheck everything.

Settings
leave everything at default.
for testing, make sure "Allow task to be run on demand" is checked.

Select OK.

for testing, run the task on demand by doing the following.
Select "Task Scheduler Library" on the left pane.
Select the name of the new Windows task created in the list at the top of the center pane.
In the right pane, under "Selected Item", select Run to test the newly created Windows task.
press F5 to refresh the task status list: it doesn't auto-update.

4.2. How to temporarily disable the Windows Task Scheduler AAM Auto Backup job

[10/14/2020 11:51 AM CST]
AAM Auto Backup uses a file lock to ensure that only one backup operation is performed at any given time on a computer.

AAM Auto Backup correctly handles all the scheduled and manual backup operation timing and synchronization issues.

As such, you don't need to disable the Windows Task Scheduler AAM Auto Backup job, when you want to manually perform a regular or incremental-only backup operation.

However, for whatever reason, if you want to disable the Windows Task Scheduler AAM Auto Backup job, (do xx).

5. How to place backup shortcuts on Windows Desktop

5.1. [11/10/2020 5:52 PM CST]

[1/16/2021 4:20 PM CST] absolutely minimally needed shortcuts.

(NOTE [1/16/2021 4:49 PM CST] In Windows 7, a shortcut to AAM_auto_backup.py cannot be set to run as administrator. a shortcut to Python.exe has a limited Target value and cannot accomodate the entire command, so it cannot be used, although it can be set to run as administrator. a batch file cannot be set to run as administrator. a shortcut to a batch file can be set to run as administrator. hence, the following method must be used.)

1. Create the backup batch files.
   1.1. In Windows Explorer, navigate to AAM Auto Backup program file directory.
   1.2. Create a folder named "backup batch files", then navigate to that folder.
   1.3. Create a batch file named `Regular backup.bat`, with the following two lines.
   `cd /D "(AAM Auto Backup program file full path)"` and

    `AAM_auto_backup.py regular-backup user "(python.exe full path)" 600 7200 "(log files full path)" True "AAM Backup Report" "(backup report sender email address)" "(backup report recipient email address)" "(backup instructions text file full path)" "q" "local|AWS" "(local backup destination directory full path)" False "(backup files encryption key)" "aam-data-backup" "main-work-computer" "(AWS secret key)" "(AWS secret key password)" "no" "yes"`.

   1.4. Create a batch file named `Incremental-only backup.bat`, with the following two lines. `cd /D "AAM Auto Backup program file full path"` and `AAM_auto_backup.py incremental-only-backup user "python.exe full path" 600 7200 "(log files full path)" True "AAM Backup Report" "(backup report sender email address)" "(backup report recipient email address)" "(backup instructions text file full path)" "q" "local|AWS" "(local backup destination directory full path)" False "(backup files encryption key)" "aam-data-backup" "main-work-computer" "(AWS secret key)" "(AWS secret key password)" "no" "yes"`.

2. Create backup shortcuts.
   2.1. Create a shortcut of `Regular backup.bat`, then rename it to `Regular backup`. Under its properties, set it to run as administrator.
   2.2. Create a shortcut of `Incremental-only backup.bat`, then rename it to `Incremental-only backup`.
   2.3. On Windows Desktop, create a folder named "Backup shortcuts".
   2.4. Copy the shortcuts to the "Backup shortcuts" folder.
3. Test every shortcut by executing it.

(UPDATE [11/12/2020 1:10 PM CST] create and put all of the following shortcuts on the Windows desktop in a separate folder named 'AAM Auto Backup shortcuts'. also keep them in an automatically backed up folder.)

three incremental-only backup shortcuts
incremental-only backup to local or network-attached data storage only
incremental-only backup to AWS S3 Glacier Deep Archive
incremental-only backup to both/all

two manual regular backup shortcuts (for use during travel)
regular main work data backup to both/all
regular audiovisual data backup to AWS S3 only
force full backup to both/all

shortcut to a python.exe, with the arguments for executing AAM Auto Backup.

each shortcut must be set to 'Run as administrator'.

(complete this document sector, after finalizing the AAM Auto Backup command formats.
[10/14/2020 11:58 AM CST])

III. AAM Auto Backup Operation Modes and Command Formats

[10/14/2020 12:08 PM CST]
To serve Asian-American Man's data backup and restoration needs, AAM Auto Backup ~~offers/supports/~~provides a number of data backup and restoration operation modes, as specified in the chapters and sections of this part.

1. Data backup and restoration terms

1.1. DRAFT 2

[10/28/2020 1:30 PM CST]
(Data) backup
(Data) restoration

(Data) backup source (directory or URL)
(Data) backup source (sub-directory or URL)
(Data) backup source location (a computer or a data storage, where the actual backup source is located at) (there can be one or more backup sources in the same backup source location)
(Data) backup source location name (a computer or data-storage name)
(Data) backup destination (directory or URL)
(Data) backup destination (sub-directory or URL)
(Data) backup destination location (a computer or a data storage)
(Data) backup destination location name (a computer or data-storage name)

Mirror backup
Full backup
Incremental backup

(Data) restoration source
(Data) restoration source location
(Data) restoration destination
(Data) restoration destination location

~~Backup~~ archive ~~(file)~~
Monovolume or single-volume ~~backup~~ archive~~?~~
Multivolume ~~backup~~ archive (~~Backup~~ archive volume ~~(file)~~ set)
~~Backup~~ archive volume ~~(file)~~
archive set (all the archives with all the backup sources)

Full-backup archive
Full-backup (backup-source)?? archive set
Incremental-backup archive
Incremental-backup (backup-source)?? archive set

(Full-backup / incremental-backup) complete-upload ~~backup~~ archive
(Full-backup / incremental-backup) complete-upload ~~backup~~ (backup-source)?? archive set
(Full-backup / incremental-backup) incomplete-upload ~~backup~~ archive
(Full-backup / incremental-backup) incomplete-upload ~~backup~~ (backup-source)?? archive set

Backup (or archive) set (a full-backup archive and its incremental-backup archives) (e.g. "latest backup/archive set with full-backup complete-upload archive for every backup source")
(Monovolume) Backup (or archive) set
(Multivolume) Backup (or archive) set
(~~unique) Backup (or archive) set name~~ (or (unique) Backup (or archive) source location name) (for indicating where backup source is located, on which computer)
Backup (or archive) set period (the period during which the backups were produced)
valid backup (or archive) set? (with a complete-upload full-backup archive)
invalid backup (or archive) set? (without a complete-upload full-backup archive)

ADDITION [11/4/2020 1:08 PM CST]
backup (operation) type: regular backup or incremental-only backup
backup scope: full backup or incremental backup
(beautiful! the above is absolutely beautiful!!!! I LOVE it!!!!! [11/4/2020 1:09 PM CST])

1.2. DRAFT 1

[10/28/2020 1:29 PM CST]
Backup
Restoration

Backup archive (file)
Monovolume or single-volume backup archive?
Multivolume backup archive
Backup archive volume (file)
Backup archive volume (file) set
(Multi-volume) Backup archive set
Full backup
Full backup archive
Incremental backup
Incremental backup archive

Backup (archive) set
Backup source
Backup location/origin/xx
(Full-backup / incremental-backup) complete-upload backup archive
(Full-backup / incremental-backup) incomplete-upload backup archive
(Anything else?) (Read the 7-Zip user manual to read the terms used in that manual.)

2. Data backup and restoration rules summary

[10/27/2020 1:09 PM CST]

AAM Auto Backup adheres to the following rules when performing data backup and restoration.

2.1. AAM Auto Backup data backup rules

2.1.1. local data backup rules

- use Robocopy to perform the local data mirror backup on Windows (with or without deleting the files in the backup destination[s] that are not present in the backup source[s]).
- [10/26/2020 8:58 PM CST] (one more thing to handle) the local backup destination directory structure—the shadow-mount directory naming in the local backup destination directory. how to do this? use a directory path mapping text file in the local backup destination directory? it can be very unreliable—the file could go missing or get damaged. I don't like the idea of creating a folder per directory path element, since that can and will result in copied file path overflow. {UPDATE [10/27/2020 11:14 AM CST] no need to use the short directory path name mapping file: simply use the full directory replication in the backup-destination folder, starting with the drive letter. Robocopy will handle everything needed! check out the additional Robocopy test 2 to find out why!} {A POSSIBILITY [10/26/2020 1:28 PM CST] in the backup destination, each shadow-mounted backup source directory name can be something else, like SHA-256 or SHA-1, or mapped to a shorter directory name like 1, to avoid the path length limit overflow. a separate plain text file in the backup destination directory can be used for saving the mapping, each line containing a directory mapping using a tap character to separate the original directory path and the mapped or transformed directory path.} {A MUST [10/26/2020 1:35 PM CST] path length limit overflow must be avoided.} (~~using shadow-mounted backup source directory name mapping or shortening with a mapping plain text file.~~)

2.1.2. AWS S3 data backup rules

2.1.2.1. AWS S3 data archive file creation rules

- use 7z.exe to create AWS S3 data archive files. other data-compression CLI software programs are not as good as 7z.exe in features. create the data archive files in the ZIP form with the AES-256 encryption. [10/27/2020 3:13 PM CST]
- **use the shadow mounts. no reason to rely on local backup for AWS S3 backup.** {TO CONSIDER [10/26/2020 12:51 PM CST] how about running the 7z command on the backup destination files after executing all the robocopy commands? the file modified time can be used for incremental backup. no need to include the absolute path (only relative), and no need to append to an existing archive file(s). the only risk is some the files in the backup destination being in use. consider this option also.} {[10/26/2020 12:58 PM CST] -ssw may have to be used if running the 7z command on the backup destination files.} {consider running the 7z command on the backup destination files after executing all the robocopy commands. [10/26/2020 12:57 PM CST]} {decide on the final backup ~~(restoration?)~~ destination path naming and 7z.exe usage method. store full path or not in the backup destination? what should be 7z.exe file source? shadow mount drive, or local backup destination? answer and document everything. [10/26/2020 1:05 PM CST]} {using the local backup destination files for AWS S3 backup, so that one set of AWS S3 backup files can be used, instead of multiple for each shadow-mounted backup source directory. (this requires always doing local backup for AWS S3 backup, which is not what I want.)}
- **use the original method of creating multiple sets of archives for uploading to AWS S3, one for each shadow-mounted backup source directory.** {TO CONSIDER [10/26/2020 10:08 AM CST] if 7z.exe can append to existing archive files, and multiple source directory issue can be handled somehow (so that there is source directory separation in the archived files), creating only one set of archive files, instead of multiple, is an option. I need to test 7z.exe commands to find out for sure. is there a way to add directory in the archived file relative paths?} {NOTE [10/26/2020 10:24 AM CST] finding and trying some other program, such as GNU ZIP on Windows, is also an option for appending to existing archive files, and providing relative path head or root in the archived files.}
- **support more than one AWS S3 full-backup archive in the same month (for use when incomplete-upload AWS S3 full backup, and/or missing or additional shadow-mounted backup source directory[ies] in complete-upload AWS S3 full backup [due to the AAM Auto Backup instruction plain text file content change]).** (NOTE, REMINDER [10/26/2020 10:04 AM CST] if the latest complete-upload AWS S3 full backup does not contain all the backup sources or shadow-mounted directories (and their contents), AAM Auto Backup performs a full backup.)   (resolve the issue of the content of the AAM Auto Backup instruction plain text file changing. what should I do with incomplete full backup AWS S3 objects, with incomplete uploads and/or missing backup sources, or additional backup sources? do I delete the backup AWS S3 objects with incomplete uploads and/or irrelevant or outdated backup sources? no, I can't delete with the upload-only account for cybersecurity reasons. I'm leaning toward creating additional full-backup in the same month as needed, when the content of the AAM Auto Backup instruction plain text file changes (i.e. the shadow-mounted directories do not match what is in the backup AWS S3 bucket).) (in fact, in order to properly handle full backup upload failure, I must implement handling more than one full backup in the same month in the backup AWS S3 bucket. so, I will do so. [10/22/2020 8:02 PM CST]) (UPDATE [10/26/2020 7:55 AM CST] I'm leaning toward implementing multiple AWS full backup in the same month, and force full backup on AWS.) {NOTE, REMINDER [10/26/2020 10:04 AM CST] if the latest complete-upload AWS S3

full backup does not contain all the backup sources or shadow-mounted directories ~~(and their contents)~~, AAM Auto Backup performs a full backup.} (NOTE [10/26/2020 10:06 AM CST] AAM Auto Backup does not consider a full backup with an incomplete-upload backup source as a complete-upload full backup.)

- **support force AWS S3 full backup via an additional regular-backup command argument.** (TO DECIDE [10/26/2020 8:02 PM CST] do 'force full backup on AWS', check the modification date and time on the AAM Auto Backup instruction plain text file to auto-decide doing full backup, or do both? I think the second option of auto-deciding is really bad and unreliable, so I won't do it. well, maybe I will. I think I'll support both. maybe support another command argument, 'full backup when AAM Auto Backup instruction plain text file is modified after the last backup date and time'. note that an AWS S3 full backup is required, because new sub-directories within the same shadow-mounted directory can contain files that are older than the last backup date and time, which would prevent incremental backup of the new files in the new sub-directories.) {TO ADDRESS [10/26/2020 7:39 AM CST] do I need to provide a 'force full backup on AWS' capability to address the above issue? to enable the user to manually perform full backup on AWS after changing the AAM Auto Backup instruction plain text file content? if I do provide that option (in all likely chance I probably will), the implementation method probably will be providing an additional argument.} {TO DECIDE [10/26/2020 8:02 PM CST] do 'force full backup on AWS', check the modification date and time on the AAM Auto Backup instruction plain text file to auto-decide doing full backup, or do both? I think the second option of auto-deciding is really bad and unreliable, so I won't do it. well, maybe I will. I think I'll support both. maybe support another command argument, 'full backup when AAM Auto Backup instruction plain text file is modified after the last backup date and time'. note that an AWS S3 full backup is required, because new sub-directories within the same shadow-mounted directory can contain files that are older than the last backup date and time, which would prevent incremental backup of the new files in the new sub-directories.} (UPDATE [10/26/2020 7:55 AM CST] I'm leaning toward implementing multiple AWS full backup in the same month, and force full backup on AWS.)
- when doing incremental-only backup, all the backup sources or shadow-mounted directories in the AAM Auto Backup instruction plain text file will have to be checked in the backup AWS S3 bucket to decide that all the complete-upload full backups exist in the backup AWS S3 bucket. (NOTE [10/26/2020 10:06 AM CST] AAM Auto Backup does not consider a full backup with an incomplete-upload backup source as a complete-upload full backup.)

2.1.2.1.1. ORIGINAL NOTES

- use 7z.exe to create AWS S3 data archive files. other data-compression CLI software programs are not as good as 7z.exe in features. create the data archive files in the ZIP form with the AES-256 encryption. [10/27/2020 3:13 PM CST]
- **use the shadow mounts. no reason to rely on local backup for AWS S3 backup.** {TO CONSIDER [10/26/2020 12:51 PM CST] how about running the 7z command on the backup destination files after executing all the robocopy commands? the file modified time can be used for incremental backup. no need to include the absolute path (only relative), and no need to append to an existing archive file(s). the only risk is some the files in the backup destination being in use. consider this option also.} {[10/26/2020 12:58 PM CST] -ssw may have to be used if running the 7z command on the backup destination files.} {consider

running the 7z command on the backup destination files after executing all the robocopy commands. [10/26/2020 12:57 PM CST]} {decide on the final backup ~~(restoration?)~~ destination path naming and 7z.exe usage method. store full path or not in the backup destination? what should be 7z.exe file source? shadow mount drive, or local backup destination? answer and document everything. [10/26/2020 1:05 PM CST]} {using the local backup destination files for AWS S3 backup, so that one set of AWS S3 backup files can be used, instead of multiple for each shadow-mounted backup source directory. (this requires always doing local backup for AWS S3 backup, which is not what I want.)}

- **use the original method of creating multiple sets of AWS S3 backup files, one for each shadow-mounted backup source directory.** {TO CONSIDER [10/26/2020 10:08 AM CST] if 7z.exe can append to existing archive files, and multiple source directory issue can be handled somehow (so that there is source directory separation in the archived files), creating only one set of archive files, instead of multiple, is an option. I need to test 7z.exe commands to find out for sure. is there a way to add directory in the archived file relative paths?} {NOTE [10/26/2020 10:24 AM CST] finding and trying some other program, such as GNU ZIP on Windows, is also an option for appending to existing archive files, and providing relative path head or root in the archived files.}
- **support more than one AWS S3 full backup in the same month (for use when incomplete-upload AWS S3 full backup, and/or missing or additional shadow-mounted backup source directory[ies] in complete-upload AWS S3 full backup [due to the AAM Auto Backup instruction plain text file content change]).** (NOTE, REMINDER [10/26/2020 10:04 AM CST] if the latest complete-upload AWS S3 full backup does not contain all the backup sources or shadow-mounted directories (and their contents), AAM Auto Backup performs a full backup.) (resolve the issue of the content of the AAM Auto Backup instruction plain text file changing. what should I do with incomplete full backup AWS S3 objects, with incomplete uploads and/or missing backup sources, or additional backup sources? do I delete the backup AWS S3 objects with incomplete uploads and/or irrelevant or outdated backup sources? no, I can't delete with the upload-only account for cybersecurity reasons. I'm leaning toward creating additional full-backup in the same month as needed, when the content of the AAM Auto Backup instruction plain text file changes (i.e. the shadow-mounted directories do not match what is in the backup AWS S3 bucket).) (in fact, in order to properly handle full backup upload failure, I must implement handling more than one full backup in the same month in the backup AWS S3 bucket. so, I will do so. [10/22/2020 8:02 PM CST]) (UPDATE [10/26/2020 7:55 AM CST] I'm leaning toward implementing multiple AWS full backup in the same month, and force full backup on AWS.) {NOTE, REMINDER [10/26/2020 10:04 AM CST] if the latest complete-upload AWS S3 full backup does not contain all the backup sources or shadow-mounted directories (and their contents), AAM Auto Backup performs a full backup.} (NOTE [10/26/2020 10:06 AM CST] AAM Auto Backup does not consider a full backup with an incomplete-upload backup source as a complete-upload full backup.)
- **support force AWS S3 full backup via an additional regular-backup command argument.** (TO DECIDE [10/26/2020 8:02 PM CST] do 'force full backup on AWS', check the modification date and time on the AAM Auto Backup instruction plain text file to auto-decide doing full backup, or do both? I think the second option of auto-deciding is really bad and unreliable, so I won't do it. well, maybe I will. I think I'll support both. maybe support another command argument, 'full backup when AAM Auto Backup instruction plain text file

is modified after the last backup date and time'. note that an AWS S3 full backup is required, because new sub-directories within the same shadow-mounted directory can contain files that are older than the last backup date and time, which would prevent incremental backup of the new files in the new sub-directories.) {TO ADDRESS [10/26/2020 7:39 AM CST] do I need to provide a 'force full backup on AWS' capability to address the above issue? to enable the user to manually perform full backup on AWS after changing the AAM Auto Backup instruction plain text file content? if I do provide that option (in all likely chance I probably will), the implementation method probably will be providing an additional argument.} {TO DECIDE [10/26/2020 8:02 PM CST] do 'force full backup on AWS', check the modification date and time on the AAM Auto Backup instruction plain text file to auto-decide doing full backup, or do both? I think the second option of auto-deciding is really bad and unreliable, so I won't do it. well, maybe I will. I think I'll support both. maybe support another command argument, 'full backup when AAM Auto Backup instruction plain text file is modified after the last backup date and time'. note that an AWS S3 full backup is required, because new sub-directories within the same shadow-mounted directory can contain files that are older than the last backup date and time, which would prevent incremental backup of the new files in the new sub-directories.} (UPDATE [10/26/2020 7:55 AM CST] I'm leaning toward implementing multiple AWS full backup in the same month, and force full backup on AWS.)

- when doing incremental-only backup, all the backup sources or shadow-mounted directories in the AAM Auto Backup instruction plain text file will have to be checked in the backup AWS S3 bucket to decide that all the complete-upload full backups exist in the backup AWS S3 bucket. (NOTE [10/26/2020 10:06 AM CST] AAM Auto Backup does not consider a full backup with an incomplete-upload backup source as a complete-upload full backup.)

2.1.2.2. AWS S3 data archive file format rules

- **use multi-volume archive files. It does not make sense to upload one big file to the Cloud.** {FINAL DECISION/CONCLUSION 1 [10/26/2020 1:17 PM CST] I need multi-volume archive files. Uploading one big archive file that can and would be tens or more gigabytes or even terabytes is not doable or a good method. Since I need multi-volume archive files, appending to 7z.exe created zip files cannot be done (using 7z.exe at the least).}
- **use relative path in archived files to avoid path limit overflow.** {decide on the final backup ~~(restoration?)~~ destination path naming and 7z.exe usage method. store full path or not in the backup destination? what should be 7z.exe file source? shadow mount drive, or local backup destination? answer and document everything. [10/26/2020 1:05 PM CST]}

2.1.2.3. AWS S3 data archive file upload rules

TO CONSIDER [10/26/2020 1:20 PM CST] when there is archive file upload failure, how about auto-trying it later, instead of deleting all the archive files, remaking all the archive files, then reuploading the next day? well, I don't think it would be a good design choice; due to the main work computer files constantly being updated, if there is a upload failure, the archive files should be generated again—for now, that's what I think. upload failure won't happen often anyway. [10/28/2020 7:13 AM CST] do the cleanup and delete the intermediate output files when there is an upload failure, before terminating the program.

2.2. AAM Auto Backup data query rules

[10/28/2020 7:35 AM CST]
AAM Auto Backup ~~data~~ query rules

2.2.1. regular-backup AAM Auto Backup AWS S3 query rules

[10/28/2020 4:29 PM CST]

Get the AWS S3 object names of the current month, and process the AWS S3 object names to find the complete-upload full-backup archive(s)/([backup-source]? archive set?) with all of the backup sources; if no complete-upload full-backup archive(s) for all the backup sources, in the current month, conclude that a full backup is required when there is no full-backup archive set; otherwise, conclude that an incremental backup is required for this instance of the regular-backup operation.

2.2.2. incremental-only backup AAM Auto Backup AWS S3 query rules

[10/28/2020 4:29 PM CST]

Query the backup AWS S3 bucket in the descending month order. Starting with the current month, get the AWS S3 object names of the current month, and process the AWS S3 object names to find the complete-upload archive(s) with all of the backup sources; if no complete full-backup archive(s) in the month, check the last month's backup set; check as far back as 180 days or six months ago; if no complete full-backup archive set found, after checking the current and last five months, conclude that an incremental-only backup operation cannot be performed; otherwise, save the latest complete-upload and complete-backup-source full-backup and incremental-backup dates and times in the archive names for performing the incremental-only backup.

2.2.3. data-restoration AAM Auto Backup AWS S3 query rules

[10/28/2020 4:29 PM CST]

Query the backup AWS S3 bucket in the descending month order. Starting with the current month, get the AWS S3 object names of the current month, and process the AWS S3 object names to find the complete-upload archive(s) with all of the backup sources; if no complete full-backup archive(s) in the month, check the last month's backup set; check as far back as 180 days or six months ago; if no complete full-backup archive set found, after checking the current and last five months, conclude that there is no data-restoration source to perform the data restoration; otherwise, save the latest complete-upload and complete-backup-source full-backup and incremental-backup AWS S3 object names for performing the data restoration.

2.3. AAM Auto Backup data restoration rules

- **when restoring AWS S3 backup, use a directory named with a number starting at 1 for each sub-set of backup, each shadow-mounted backup source directory. (or better yet, starting at a. [not in this version, later on].)**
- if download failure, ~~(what to do exactly? specify right here! [10/27/2020 3:18 PM CST])~~ display the error in terminal, save the error in the local log file if this is one-step data restoration operation, then abort the data-restoration operation.

3. Data-backup Operation Modes

3.1. Data Backup Operation Modes Overview and Usage Cases

[10/14/2020 1:49 PM CST]

3.1.1. Three main data backup storage mediums

[10/14/2020 3:03 PM CST]

local or network-attached data storage. the data storage is within the proximity of the computer with the data to back up.

the Cloud. the data storage is somewhere in the world, managed as a complete data storage service on the Internet.

backup tape. for lowest-cost, long-term data archiving (for one or two decades or more).

3.1.2. Three main types of data backup operations, and when to execute each

[10/14/2020 2:55 PM CST]

regular backup to a local or network-attached data storage, and AWS S3 Glacier Deep Archive. auto-execute regular backup once a day using Windows Task Scheduler.

incremental-only backup to a local or network-attached data storage, AWS S3 Glacier Deep Archive, or both. manually execute incremental-only backup whenever you want to quickly have your new or updated files backed up.

(future support) tape backup to a backup tape in a tape drive. manually (or automatically??) execute it when the local or network-attached data storage is full and when you want to put away the data storage for archiving purposes, or empty the data storage and reuse it for other data. take your backup tape(s) to a safe deposit once or twice in a year for long-term storage.

AAM Auto Backup uses an (atomic) file lock to control the backup operation executions, and runs only one backup operation at a time. As such, any backup operation, both regular and incremental-only, both scheduled and manual, can be initiated at any time without causing any data backup problem. For more info on this, refer to the "Use of an (atomic) file lock for regular and incremental-only backup timing control" section in this document.

3.1.2.1. scheduled (auto-executed) and manually-executed data backups

[10/15/2020 1:05 PM CST]
scheduled, auto-executed daily regular backup

user-executed regular and incremental-only backup (to USB hard drive and AWS S3 Glacier Deep Archive)

3.1.2.2. same-day incremental-only backup to AWS S3

[10/15/2020 1:06 PM CST]
AAM Auto Backup supports performing incremental-only backup as many times as wanted in the same day.

3.1.3. AAM Auto Backup execution computers, backup storage mediums, and unique backup-set prefixes

[10/15/2020 10:07 AM CST]
<u>the main work computer</u>
back up the main work data to both the attached USB drive and AWS S3 Glacier Deep Archive.
AWS S3 Glacier Deep Archive backup prefix: work-files

back up the audiovisual data to AWS S3 Glacier Deep Archive only (when traveling to safeguard against audiovisual data loss).
AWS S3 Glacier Deep Archive backup prefix: audiovisual-files

<u>the highest-security computer</u>
backup to AWS S3 Glacier Deep Archive only.
AWS S3 Glacier Deep Archive backup prefix: highest-security
(secure-computer login info files auto-backup [10/9/2020 11:16 AM CST])

3.1.3.1. main work computer work files auto-backup (scheduled regular backup)

[10/2/2020 5:59 AM CST]
for the main work computer data, daily mirror backup (data replication) to the USB hard drive (to a folder).
for the main work computer data, monthly full backup, daily incremental backup to AWS S3 Deep Glacier.

3.2. Regular-backup Operation Mode

[10/14/2020 1:44 PM CST]
As the user of AAM Auto Backup, you execute AAM Auto Backup in its regular-backup operation mode, in order to automatically perform full and incremental backup to a local or network-attached data storage, AWS S3 Glacier Deep Archive, or both.

3.2.1. operation mode overview

3.2.1.1. [11/13/2020 12:44 PM CST]

In the regular backup operation mode, AAM Auto Backup performs the following actions.

When performing the regular backup to a local data storage (e.g. internal hard drive, USB hard drive, or network-attached hard drive), AAM Auto Backup replicates the backup-source directory files to the user-specified data replication (backup-destination) directory on the local data storage, using Robocopy on Windows that is executed via ShadowSpawn.exe. AAM Auto Backup preserves the full path of each copied file in the backup-destination directory, by pre-creating every intermediate folder in the shadow-mounted directory path as needed, starting with the driver-letter folder in the data-replication destination directory, to indicate the path of the shadow-mounted directory, in the backup-destination folder, and then executing Robocopy to copy files to the mirror directory in the backup destination. (Robocopy handles both full and incremental backup automatically. No need to be concerned about deciding on full or incremental backup in the Python code when doing backup locally using Robocopy.)
(in the user manual, all the elaborate details of implementation above should be hidden; they are only for the developers. [11/20/2020 2:36 PM CST])

When performing the regular backup to AWS S3, AAM Auto Backup queries the backup AWS S3 bucket to find out the latest full and incremental backup dates and times, and uses the queried dates and times from the backup AWS S3 bucket objects to decide which type of backup to perform (full or incremental), and which backup-source files to include in the regular-backup archive to AWS S3.

If the backup AWS S3 bucket cannot be queried, due to an internet or AWS S3 connection problem, AAM Auto Backup reports the error in terminal and later in the email report and the local backup log file, and skips creating and uploading the AWS S3 backup files.

AAM Auto Backup queries the backup AWS S3 bucket, instead of using a local text file for keeping the latest full and incremental backup dates and times, to check the internet and AWS S3 connection before producing the encrypted and compressed AWS S3 backup files, and to simplify coding—since the backup AWS S3 bucket must be queried when a local text file for keeping the latest full and incremental backup dates and times does not exist, it is ~~much~~ simpler in code to not use such a local text file at all. Moreover, querying the backup AWS S3 bucket every time is more error-resistant and error-handling-and-recovery-friendly, since the AWS S3 backup files are checked directly on their source—a local text file could contain erroneous information due to the file being corrupted or AAM Auto Backup being terminated (due to a power outage, for example) before updating the latest full and incremental backup dates and times plain-text file. All in all, maintaining an extra file that is not absolutely required is a waste that should not be done. Note that querying AWS S3 incurs a very small monetary cost (US$0.005 per 1,000 LIST requests).

If the full backup for the current month does not exist in the backup AWS S3 bucket according to the backup AWS S3 object list query, AAM Auto Backup performs the full backup; otherwise, it performs the incremental backup.

If there is no file to incrementally backup, because there is no change or new file, AAM Auto Backup reports the no incremental backup status in terminal and later in the local log file and the email report.

Even when there is no file to incremental backup in a backup source, the upload completion indicator AWS S3 object is created and uploaded to AWS S3, in order to ~~(indicate that the processing has been done properly on the local computer)/~~(use the last incremental-backup time in the next incremental backup).

AAM Auto Backup supports multiple AWS full backups in the same month, force AWS full backup option, and automatic force full backup option—for properly handling AAM Auto Backup instruction plain text file content change, and full backup upload failure. AAM Auto Backup can and will properly handle the scenarios when there must be, or are, more than one full backup in the same month.

At the end of each regular backup operation, AAM Auto Backup creates and emails a backup report via AWS SES, if the email report setting is set to on in the AAM Auto Backup command; after sending the report email (if applicable), AAM Auto Backup saves the local backup log file.

3.2.1.2. DEPRECATED [11/13/2020 12:44 PM CST]

[10/20/2020 10:51 AM CST]

TO ADD [10/30/2020 4:12 PM CST] even when there is no incremental backup (for a backup source), the upload completion indicator AWS S3 object must be uploaded to indicate that the processing has been done properly on the local computer.

In the regular backup operation mode, AAM Auto Backup performs the following actions.

When performing the regular backup to a local data storage (e.g. internal hard drive, USB hard drive, or network-attached hard drive), AAM Auto Backup replicates the backup source directory files to the user-specified data replication directory on the local data storage, using Robocopy on Windows that is executed via ShadowSpawn.exe. AAM Auto Backup pre-creates every intermediate folder as needed, starting with the driver-letter folder in the data-replication destination directory. (Robocopy handles both full and incremental backup automatically. No need to be concerned about deciding on full or incremental backup in the Python code when doing backup locally using Robocopy.)
(TO DECIDE [10/20/2020 4:05 PM CST] instead of creating every intermediate folder in the local backup destination directory, how about using the unique backup-set prefix appended by a counter starting at 1? that would reduce the total path length. think on this, decide on this, and then implement the decision.)

(AN ADDITIONAL ISSUE TO ADDRESS [10/26/2020 7:43 AM CST] AAM Auto Backup instruction plain text file content change, multiple AWS full backups in the same month, and force AWS full backup option.)

When performing the regular backup to AWS S3, AAM Auto Backup queries the backup AWS S3 bucket to find out the latest full and incremental backup dates and times, and uses the queried dates and times from the backup AWS S3 bucket objects to decide which type of backup to perform (full or incremental), and which backup-source files to include in the regular backup to AWS S3.

If the backup AWS S3 bucket cannot be queried, due to an internet or AWS S3 connection problem, AAM Auto Backup reports the error in terminal and later in the email report and the local backup log file, and skips creating and uploading the AWS S3 backup files.

AAM Auto Backup queries the backup AWS S3 bucket, instead of using a local text file for keeping the latest full and incremental backup dates and times, to check the internet and AWS S3 connection before producing the encrypted and compressed AWS S3 backup files, and to simplify coding—since the backup AWS S3 bucket must be queried when a local text file for keeping the latest full and incremental backup dates and times does not exist, it is ~~much~~ simpler in code to not use such a local text file at all. Moreover, querying the backup AWS S3 bucket every time is more error-resistant and error-handling-and-recovery-friendly, since the AWS S3 backup files are checked directly on their source—a local text file could contain erroneous information due to the file being corrupted or AAM Auto Backup being terminated (due to a power outage, for example) before updating the latest full and incremental backup dates and times plain-text file. All in all, maintaining an extra file that is not absolutely required is a waste that should not be done. Note that querying AWS S3 incurs a very small monetary cost (US$0.005 per 1,000 LIST requests).

If the full backup for the current month does not exist in the backup AWS S3 bucket according to the backup AWS S3 object list query, AAM Auto Backup performs the full backup; otherwise, it performs the incremental backup.

If there is no file to incrementally backup, because there is no change or new file, AAM Auto Backup reports the no incremental backup status in terminal and later in the local log file and the email report.

At the end of each regular backup operation, AAM Auto Backup creates and emails a backup report via AWS SES, if the email report setting is set to on in the AAM Auto Backup command; after sending the report email (if applicable), AAM Auto Backup saves the local backup log file.

3.2.1.3. DEPRECATED [10/20/2020 10:51 AM CST]

[10/15/2020 1:10 PM CST]

In the regular backup operation mode, AAM Auto Backup performs the following actions.

When performing the regular backup to a local or network-attached data storage, AAM Auto Backup replicates the backup source directory files to the user-specified data replication directory on the local or network-attached data storage, using Robocopy on Windows.

When performing the regular backup to AWS S3, AAM Auto Backup checks the last AWS S3 backup dates and times plain text file content to decide the files to include in the incremental backup. If the last AWS S3 backup date and time plain text file does not exist, AAM Auto Backup reports the nonexistence in terminal and later in the email report, queries the backup AWS S3 bucket to find out the latest backup date and time, and uses the date and time info from the backup AWS S3 bucket to decide which files to include in the regular backup.

~~AAM Auto Backup queries the backup AWS S3 bucket to find out whether the full backup for the current month exists or not.~~

If the full backup for the current month ~~(does not exist in the backup AWS S3 bucket)/~~(is not done yet according to the last AWS S3 backup dates and times plain text file content, or the backup AWS S3 object list query), AAM Auto Backup performs the full backup; otherwise, it performs the incremental backup.

If there is no file to incrementally backup, because there is no change or new file, AAM Auto Backup reports the no incremental backup status in terminal and later in the email report.

After performing each AWS S3 backup, AAM Auto Backup creates or updates the last AWS S3 backup dates and times plain text file content. When a full backup is done, AAM Auto Backup sets both the 'last AWS S3 full backup' and 'last AWS S3 incremental backup' entries to the latest full backup date and time; when an incremental backup is done, AAM Auto Backup sets the 'last AWS S3 incremental backup' entry to the latest incremental backup date and time.

At the end of each regular backup operation, AAM Auto Backup creates and emails a backup report via AWS SES, if the email report setting is set to on in the AAM Auto Backup command.

(ORIGINAL NOTE [10/17/2020 11:40 AM CST] how to decide when to do full and incremental backups.)
TO DO [10/6/2020 10:31 AM CST] if there is no backup-destination root folder on the USB hard drive, full backup. actually, if there is no last backup date plain text file in the program file folder, full backup to the USB hard drive. (UPDATE [10/17/2020 11:38 AM CST] Robocopy handles both full and incremental backup automatically. no need to be concerned about that in the Python code when doing backup locally using Robocpy.) also, full backup at the 1st of every month to AWS S3 Glacier Deep Archive, incremental backup to AWS S3 Glacier Deep Archive for other days of the month; also, full backup to AWS S3 Glacier Deep Archive if there is no backup file at the 1st of the month—this requires checking AWS S3 using the AWS CLI program with an appropriate command.

3.2.2. log and email-report outputs

[10/17/2020 3:33 PM CST]
operation summary. (operation type, etc.)
errors.
ShadowSpawn.exe execution output.

3.2.3. command arguments and format

[10/15/2020 12:47 PM CST] all the arguments must be present. no named arguments supported.

1. (program execution settings)
    - 1.1. [argument 1] `regular-backup` (operation type)
    - 1.2. [argument 2] `user` (execution agent type. the valid values are 'user' and 'ShadowSpawn'.)
    - 1.3. [argument 3] [10/7/2020 3:24 PM CST] an argument: the full path of the Python executable to be called by ShadowSpawn. e.g. C:\Users\Allen\AppData\Local\Programs\Python\Python38\python.exe ([10/14/2020 2:42 PM CST] the full path of the Python.exe that will be executed via a ShadowSpawn command.)
    - 1.4. (wait times) ([10/13/2020 5:02 PM CST] the user provides the on-going backup (i.e. file-lock) waiting period and the total on-going backup (i.e. file-lock) wait time for both the regular and the incremental-only backup processes.)

        1.4.1. [argument 4] ongoing backup (i.e. file-lock) waiting period in seconds
        1.4.2. [argument 5] total ongoing backup (i.e. file-lock) wait time in seconds
    1.5. [argument 6] (log directory full path) ([10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).)
    1.6. [argument 7] [10/14/2020 9:38 AM CST] an argument to indicate whether to send email report or not. must be either 'True' for sending an email report, or 'False' for not sending an email report.
    1.7. [argument 8] email sender name [10/18/2020 5:54 PM CST]
    1.8. [argument 9] email sender address [10/18/2020 5:54 PM CST]
    1.9. [argument 10] email recipient address [10/18/2020 5:42 PM CST]
2. (backup source setting[s?])
    2.1. [argument 11] [10/6/2020 1:39 PM CST] the full path to the AAM Auto Backup instruction plain text file is provided as an argument in the AAM Auto Backup command. ([10/15/2020 10:44 AM CST] an AAM Auto Backup instruction plain text file full path)
    2.2. [[argument 12] 10/6/2020 5:29 PM CST] an argument: shadow copy drive name (letter)
3. (backup destination settings)
    3.1. [argument 13] (backup destination types) `local`~~`/proximity(use which one?)/intranet?/non-Cloud?`~~ | ~~`Cloud/Internet?/`~~`AWS` ~~|both~~ (FINAL DECISION [10/16/2020 4:01 PM CST] use flags with | as the separator. the supported flags are 'local' and 'AWS') ([10/14/2020 8:10 AM CST] an argument to indicate the backup destination types: local or network-attached data storage, AWS S3 Glacier Deep Archive, or both.) ([10/16/2020 3:52 PM CST] how about just using 'local' to denote all the data storages that are physically nearby, such as internal data storage, attached data storage such as USB hard drive, network-attached data storage, and data storage on a LAN? I'm very strongly leaning toward that; actually, I'll use this. search "how distant can Wide area network can be?" online. https://www.apposite-tech.com/blog/whats-difference-metropolitan-area-network-man-wide-area-network-wan/ https://en.wikipedia.org/wiki/Wide_area_network) [this argument is also used in the incremental-only backup AAM Auto Backup command] ([10/10/2020 12:51 PM CST] AAM Auto Backup should take command arguments for making incremental-only backups to USB drive only, AWS S3 only, or both USB drive and AWS S3.) ([10/9/2020 10:16 AM CST] an argument: incremental backup to AWS only) ([10/14/2020 7:55 AM CST] incremental-only backup destinations argument: local or network-attached data storage (done using Robocopy), AWS S3 Glacier Deep Archive (Done using AWS Python API), or both.)
    3.2. [argument 14] (a local ~~or network-attached~~ data storage backup-destination directory full path) ([10/6/2020 3:48 PM CST] an argument: the local (non-Cloud) backup destination root folder location full path.) (TO DO [10/8/2020 1:47 PM CST] support for auto-backup to AWS S3 Glacier Deep Archive only. if this argument is empty or blank (i.e. ""), then local backup is skipped. this is for backing up the login infos on the

ultra-security laptop.) ([10/15/2020 1:32 PM CST] user-specified data replication destination directory full path.)

3.3. [argument 15] (keep missing backup sources files in the backup destination) [10/14/2020 8:08 AM CST] an argument to indicate whether to leave the files in the backup (data replication) destination that are no longer present in the backup source. must be 'True' or 'False'. (very easy and fast to implement, using the single Robocopy argment.) (FINAL DECISION [11/2/2020 4:28 PM CST] 'keep' leaves the files in the backup destination that do not exist in the backup source; 'delete' deletes them.)

3.4. (backup-destination AWS S3 info)

~~3.4.1. [10/7/2020 8:32 AM CST] an argument: backup-only AWS account AWS CLI program profile name (used in executing the backup AWS CLI commands). (no need. AWS Python API, Boto3, will be used for authentication in each backup [and restoration] operation. [10/16/2020 1:26 PM CST])~~

3.4.2. [argument 16] [10/9/2020 8:53 AM CST] an argument: backup encryption AES-256 key.

3.4.3. [argument 17] [10/15/2020 1:22 PM CST] the backup AWS S3 bucket name.

3.4.4. [argument 18] (a unique backup-source location name)~~/(a unique backup-set prefix)~~ [10/10/2020 1:05 PM CST] ~~AWS S3 bucket name for the backup.~~ better yet, the unique backup file tag or prefix. all stored in the same AWS S3 bucket. ([10/15/2020 10:08 AM CST] a unique (backup-set)/~~computer~~ prefix) (UPDATE [10/28/2020 3:32 PM CST] a unique backup-source location name)

3.4.5. [argument 19] [10/14/2020 9:41 AM CST] an argument for the backup AWS IAM user access key ID

3.4.6. [argument 20] [10/14/2020 9:41 AM CST] an argument for the backup AWS IAM user secret access key

3.4.7. [argument 21] [11/11/2020 5:02 PM CST] Force full backup. The value must be "yes" or "no". The value applies only when regular backup. if this is set to 'yes', the regular backup performs full backup. ordinarily, this should be set to 'no', and should be 'yes' only when forcing full backup after the AAM Auto Backup instruction plain text file content change.

3.4.8. [argument 22] [11/12/2020 10:48 AM CST] Automatic force full backup. The value must be "yes" or "no". when this is set to 'yes', force full backup is enforced automatically if the AAM Auto Backup instruction plain text file modification or creation time is after the last AWS S3 backup time.

([10/15/2020 1:37 PM CST] [support in] backup package or form format specification for backup to local or network-attached storage—replication or multi-volume encrypted and compressed files.)

3.2.4. [OLD NOTES, OUTDATED, INCOMPLETE][10/17/2020 11:55 AM CST] regular backup to AWS S3

[10/13/2020 5:44 PM CST]

AAM Auto Backup regular backup process queries the backup AWS S3 bucket object list(s) to find out whether a full backup exists in the backup AWS S3 bucket for this month; if the full backup for this month does not exist in the backup AWS S3 bucket, AAM Auto Backup performs the full backup to AWS S3 for this month.

if the full backup for this month already exists in the backup AWS S3, AAM Auto Backup regular backup process performs the incremental backup to AWS S3, using the latest AWS S3 backup date time plain text file. after the backup completion, it updates the text file, and releases the file lock before completing the process and quitting.

if the full backup for this month already exists in the backup AWS S3, and no latest AWS S3 backup date time plain text file, (what should AAM Auto Backup do in this error case? it should definitely indicate the error in both the terminal and the email. other than that, what to do? think, decide, and then implement the decision!!!!! [10/13/2020 5:52 PM CST]) (FINAL DECISION [10/13/2020 6:06 PM CST] **include both backup date and time in each AWS S3 backup object name.** then, use the latest backup date and time in the latest backup set with a fully-uploaded full backup, for incremental backup. **this method should be used only when the latest AWS S3 backup date time plain text file is missing, since querying AWS S3 costs money, and shouldn't be used unless absolutely necessary. storage in, and restoration from, AWS S3 costs money: deciding what to store in AWS S3 should not cost money when it can be avoided.**)

3.3. Incremental-only Backup Operation Mode

[10/14/2020 1:44 PM CST]
~~As the user of AAM Auto Backup, you~~ execute AAM Auto Backup in its incremental-only backup operation mode, in order to backup only the updated or new files to a local ~~or network-attached~~ data storage, AWS S3 Glacier Deep Archive, or both.

[10/20/2020 1:53 PM CST] the incremental-only backup enables the same-day incremental backup to a local data storage and AWS S3.

the incremental-only backup is for quickly initiating an incremental backup before I go out and leave my computer. create a shortcut on the OS desktop with admin access to run this out-of-schedule incremental backup. [10/8/2020 8:01 AM CST] (groovy! so damn groovy!!!!! [10/20/2020 1:59 PM CST])

3.3.1. operation mode overview

3.3.1.1. [11/20/2020 2:55 PM CST]

In the incremental-only backup operation mode, AAM Auto Backup performs the following actions.

When performing the incremental-only backup to a local data storage (e.g. internal hard drive, USB hard drive, or network-attached hard drive), AAM Auto Backup replicates the backup source directory files to the user-specified data replication directory on the local data storage, using Robocopy on Windows that is executed via ShadowSpawn.exe. AAM Auto Backup pre-creates every intermediate folder as needed, starting with the driver-letter folder in the data-replication destination directory. (Robocopy handles both full and incremental backup automatically. No need to be concerned about deciding on full or incremental backup in the Python code when doing backup locally using Robocopy.)

AAM Auto Backup does not perform a full backup when performing the incremental-only backup to AWS S3, for the fastest backup.

When performing the incremental-only backup to AWS S3, AAM Auto Backup queries the backup AWS S3 bucket object list to retrieve the completely-uploaded latest full backup date and time in the backup AWS S3 bucket, and uses the AWS S3 retrieved latest completely-uploaded full backup date and time in its incremental backup file names. Also, AAM Auto Backup uses the backup AWS S3 bucket object list query result to retrieve the latest completed-upload backup date and time in the backup AWS S3 bucket, and uses the latest AWS S3 backup date and time to decide which files to incrementally back up to the backup AWS S3 bucket.
[10/15/2020 10:34 AM CST] AAM Auto Backup queries the AWS S3 bucket object list(s) as needed to determine the latest full backup to use, for both incremental-only data backup and data restoration from AWS S3.

If there is no completely-uploaded full backup in the backup AWS S3 bucket, AAM Auto Backup displays the error in the terminal and the local backup log, and aborts the incremental-only backup operation.

AAM Auto Backup checks both the date and time of the files on the shadow copy mount, to build the list of the files to incrementally back up to AWS. Because of this feature, there can be a full backup file set, and an incremental backup file set, on the same day; also, there can be more than one incremental backup file set on the same day. AAM Auto Backup handles all these cases properly.

No incremental backup files generation and upload to AWS S3, when there is no changed file
[10/13/2020 4:47 PM CST] When there are no updated or new files to incrementally backup to AWS S3, it is reported in terminal and the local backup log.
this applies to both the regular and the incremental-only backup processes.
[10/15/2020 10:58 AM CST] when there is no file to incrementally backup, it is reported in both the terminal and email report.

TO ADD [10/30/2020 4:12 PM CST] even when there is no incremental backup (for a backup source), the upload completion indicator AWS S3 object ~~(must be uploaded to indicate that the processing has been done properly on the local computer)/~~(is created and uploaded for using the last incremental backup time in the next incremental backup).

When doing data restoration from AWS S3, AAM Auto Backup queries the backup AWS S3 bucket to retrieve the latest completely-uploaded full backup AWS S3 objects in the backup AWS S3 bucket, and all of the full backup's incremental backup AWS S3 objects, and uses the AWS S3 retrieved latest full backup and all of its incremental backups in its data restoration operation. As such, AAM Auto Backup will always restore the latest data backed up to AWS S3, whether the incremental-only backup operation has been used or not.

At the end of each incremental-only backup operation, AAM Auto Backup does not email a backup report via AWS SES, saves the local backup log, and waits for a keyboard user input before exiting the program to ensure that the user sees all the backup messages in the terminal. [10/13/2020 4:38 PM CST] **at the end of the incremental-only backup process, AAM Auto Backup waits for the user input of any key press before terminating the program: this is done to ensure that the terminal window stays open and the user can check the terminal output before the terminal window gets closed. without this feature, executing a command shortcut on the OS desktop would close the terminal window automatically when AAM Auto Backup quits.**

3.3.1.2. DEPRECATED [11/20/2020 2:55 PM CST]

[10/20/2020 1:35 PM CST]

TO ADD [10/30/2020 4:12 PM CST] even when there is no incremental backup (for a backup source), the upload completion indicator AWS S3 object must be uploaded to indicate that the processing has been done properly on the local computer.

In the incremental-only backup operation mode, AAM Auto Backup performs the following actions.

When performing the incremental-only backup to a local data storage (e.g. internal hard drive, USB hard drive, or network-attached hard drive), AAM Auto Backup replicates the backup source directory files to the user-specified data replication directory on the local data storage, using Robocopy on Windows that is executed via ShadowSpawn.exe. AAM Auto Backup pre-creates every intermediate folder as needed, starting with the driver-letter folder in the data-replication destination directory. (Robocopy handles both full and incremental backup automatically. No need to be concerned about deciding on full or incremental backup in the Python code when doing backup locally using Robocopy.)

AAM Auto Backup does not perform a full backup when performing the incremental-only backup to AWS S3, for the fastest backup.

When performing the incremental-only backup to AWS S3, AAM Auto Backup queries the backup AWS S3 bucket object list to retrieve the completely-uploaded latest full backup date and

time in the backup AWS S3 bucket, and uses the AWS S3 retrieved latest completely-uploaded full backup date and time in its incremental backup file names. Also, AAM Auto Backup uses the backup AWS S3 bucket object list query result to retrieve the latest completed-upload backup date and time in the backup AWS S3 bucket, and uses the latest AWS S3 backup date and time to decide which files to incrementally back up to the backup AWS S3 bucket.
[10/15/2020 10:34 AM CST] AAM Auto Backup queries the AWS S3 bucket object list(s) as needed to determine the latest full backup to use, for both incremental-only data backup and data restoration from AWS S3.

If there is no completely-uploaded full backup in the backup AWS S3 bucket, AAM Auto Backup displays the error in the terminal and the local backup log, and aborts the incremental-only backup operation.

AAM Auto Backup checks both the date and time of the files on the shadow copy mount, to build the list of the files to incrementally back up to AWS. Because of this feature, there can be a full backup file set, and an incremental backup file set, on the same day; also, there can be more than one incremental backup file set on the same day. AAM Auto Backup handles all these cases properly.

No incremental backup files generation and upload to AWS S3, when there is no changed file
[10/13/2020 4:47 PM CST] When there are no updated or new files to incrementally backup to AWS S3, it is reported in terminal and the local backup log.
this applies to both the regular and the incremental-only backup processes.
[10/15/2020 10:58 AM CST] when there is no file to incrementally backup, it is reported in both the terminal and email report.

When doing data restoration from AWS S3, AAM Auto Backup queries the backup AWS S3 bucket to retrieve the latest completely-uploaded full backup AWS S3 objects in the backup AWS S3 bucket, and all of the full backup's incremental backup AWS S3 objects, and uses the AWS S3 retrieved latest full backup and all of its incremental backups in its data restoration operation. As such, AAM Auto Backup will always restore the latest data backed up to AWS S3, whether the incremental-only backup operation has been used or not.

At the end of each incremental-only backup operation, AAM Auto Backup does not email a backup report via AWS SES, saves the local backup log, and waits for a keyboard user input before exiting the program to ensure that the user sees all the backup messages in the terminal.
[10/13/2020 4:38 PM CST] **at the end of the incremental-only backup process, AAM Auto Backup waits for the user input of any key press before terminating the program: this is done to ensure that the terminal window stays open and the user can check the terminal output before the terminal window gets closed. without this feature, executing a command shortcut on the OS desktop would close the terminal window automatically when AAM Auto Backup quits.**

3.3.1.3. DEPRECATED [10/20/2020 1:35 PM CST]

[10/15/2020 2:01 PM CST]
When performing the regular backup to a local or network-attached data storage, AAM Auto Backup replicates the backup source directory files to the user-specified data replication directory on the local or network-attached data storage, using Robocopy on Windows.

When performing the incremental-only backup to AWS S3, AAM Auto Backup uses the last AWS S3 backup dates and times plain text file content, to decide the completely-uploaded latest full backup date and time in the backup AWS S3 bucket, and the files to incrementally backup to AWS S3. This ensures that the incremental-only backup always correctly uses the latest completely-uploaded full backup date and time in the backup AWS S3 bucket—even and especially when regular and/or incremental-only backup is performed twice in the same day, on the day of a full backup, for whatever reason.

If the last AWS S3 backup dates and times plain text file is missing, AAM Auto Backup queries the backup AWS S3 bucket object list to retrieve the completely-uploaded latest full backup date and time in the backup AWS S3 bucket, and uses the AWS S3 retrieved latest full backup date and time in its incremental backup file names. Also, AAM Auto Backup uses the backup AWS S3 bucket object list query result to retrieve the latest backup date and time in the backup AWS S3 bucket, and uses the latest AWS S3 backup date and time to decide which files to incrementally back up to the backup AWS S3 bucket.

Querying the backup AWS S3 bucket object list, instead of using the last AWS S3 backup dates and times plain text file content when the text file is missing, ensures that incremental-only backup works even when the text file is missing for whatever reason.

AAM Auto Backup does not perform a full backup when performing the incremental-only backup to AWS S3, for the fastest backup.

If there is no completely-uploaded full backup in the backup AWS S3 bucket, AAM Auto Backup displays the error in the terminal, and aborts the incremental-only backup operation.

When doing data restoration from AWS S3, AAM Auto Backup queries the backup AWS S3 bucket to retrieve the completely-uploaded latest full backup date and time in the backup AWS S3 bucket, and uses the AWS S3 retrieved latest full backup date and time in its data restoration operation. As such, AAM Auto Backup will always restore the latest data backed up to AWS S3, whether the incremental-only backup operation has been used or not.

After performing each AWS S3 incremental-only backup, AAM Auto Backup creates or updates the last AWS S3 backup dates and times plain text file content. If the text file is missing, AAM Auto Backup sets the 'last AWS S3 full backup' to the value retrieve from AWS S3, and the 'last AWS S3 incremental backup' to the latest incremental backup to AWS S3 that was just done;

when the text file is present, AAM Auto Backup sets the 'last AWS S3 incremental backup' entry to the latest incremental backup date and time.

At the end of each incremental-only backup operation, AAM Auto Backup does not email a backup report via AWS SES, and it waits for a keyboard user input before exiting the program to ensure that the user sees all the backup messages in the terminal.

[10/15/2020 1:10 PM CST]

update the AAM Auto Backup document to support the same-day incremental backup to AWS. AAM Auto Backup checks both the date and time of the files on the shadow copy mount, to build the list of the files to incrementally back up to AWS. because of this feature, there can be a full backup file set, and an incremental backup file set, on the same day; also, there can be more than one incremental backup file set on the same day. AAM Auto Backup must and will handle all those cases properly. this feature is for quickly initiating an incremental backup before I go out and leave my computer. create a shortcut with admin access to run this out-of-schedule incremental backup. document all this. [10/8/2020 8:01 AM CST]

[10/13/2020 4:38 PM CST] **at the end of the incremental-only backup process, AAM Auto Backup waits for the user input of any key press before terminating the program: this is done to ensure that the terminal window stays open and the user can check the terminal output before the terminal window gets closed. without this feature, executing a command shortcut on the OS desktop would close the terminal window automatically when AAM Auto Backup quits.**

No incremental backup files generation and upload to AWS S3, when there is no changed file
[10/13/2020 4:47 PM CST]
this applies to both the regular and the incremental-only backup processes.
[10/15/2020 10:58 AM CST] when there is no file to incrementally backup, it is reported in both the terminal and email report.

[10/15/2020 10:34 AM CST] AAM Auto Backup queries the AWS S3 bucket object list(s) as needed to determine the latest full backup to use, for both incremental-only data backup and data restoration from AWS S3.

3.3.2. log and email-report outputs

[10/17/2020 3:33 PM CST]
operation summary. (operation type, etc.)
errors.
ShadowSpawn.exe execution output.

3.3.3. command arguments and format
[10/15/2020 12:47 PM CST] [10/16/2020 4:20 PM CST]

the same as the 'Regular-backup Operation Mode' command arguments and format, except the first argument being `incremental-only-backup` instead of `regular-backup`.

([10/15/2020 1:37 PM CST] [support in] backup package or form format specification for backup to local or network-attached storage—replication or multi-volume encrypted and compressed files.)

3.3.4. [OLD NOTES, OUTDATED, INCOMPLETE][10/17/2020 11:55 AM CST] incremental-only backup software process
[10/13/2020 5:18 PM CST]

AAM Auto Backup queries the backup AWS S3 bucket object list(s) to decide which full backup to ~~indicate~~/include in its incremental-only backup file names. AAM Auto Backup uses the latest full-backup AWS S3 bucket object set with the upload completion indication AWS S3 bucket object.

due to the use of the (atomic) file lock in backup processes, the full backup indicated in the incremental-only backup file names are guaranteed to be the latest.

[10/13/2020 5:53 PM CST]
AAM Auto Backup AWS S3 incremental-only backup process displays an error in the terminal, tells the user to perform full backup first, and quits if there is no full backup AWS S3 objects.

3.4. ShadowSpawn-executed Backup Operation Mode
ShadowSpawn-executed Backup (to USB hard drive and AWS S3 Glacier Deep Archive)
ShadowSpawn-executed Backup Operation Mode
ShadowSpawn-executed ~~AWS S3~~ Backup Operation Mode
ShadowSpawn-execution Backup Operation Mode
ShadowSpawn-executed Backup Operation Mode

User-executed backup vs. ShadowSpawn-executed backup (to USB hard drive and AWS S3 Glacier Deep Archive)

[10/14/2020 2:27 PM CST] this operation mode is never directly executed by the user; it is executed by AAM Auto Backup via executing ShadowSpawn.exe.

[10/14/2020 2:20 PM CST]

When doing the full and incremental data backup to AWS S3, AAM Auto Backup executes a ShadowSpawn.exe command that executes AAM Auto Backup, so that the ShadowSpawn-executed AAM Auto Backup can uses the shadow copy mounted drive to build the 7z.exe list file that includes the paths of the files to incrementally back up.

[10/14/2020 2:33 PM CST]
The Python CLI program that is executed by ShadowSpawn.exe can execute another command such as a batch file (for example, by executing os.system() with the batch file full path argument). As such, AAM Auto Backup can execute a dynamically generated batch file when executed by ShadowSpawn.exe.

However, it is also possible for ShadowSpawn.exe to execute a Windows terminal batch file that executes AAM Auto Backup with the proper arguments, so that the 7z.exe list file gets generated before calling 7z.exe.

AAM Auto Backup uses the first/second(which one?) method of ShadowSpawn.exe AAM Auto Backup execution.

[10/7/2020 3:15 PM CST]
for the background info on why this software process is required in AAM Auto Backup, refer to the "[failure] test for omitting all the non-backup folders in the backup directory in the Robocopy command by calling AAM Auto Backup again in the ShadowCopy command executed by AAM Auto Backup 1" section below.

[10/2/2020 5:59 AM CST]
this backup program uses Windows Shadow Backup (somehow).

3.4.1. operation mode overview
[10/15/2020 1:10 PM CST]

[11/3/2020 10:30 AM CST] **how to determine whether to perform full or incremental backup.**
if regular backup, and "the latest AWS S3 full-backup date and time string" indicates that there is no full backup in the current month, perform full backup.
[11/3/2020 10:43 AM CST] write and run empty_string_argument_tester.py.
[11/3/2020 11:02 AM CST] ok, empty string argument works!
think and decide whether full or incremental backup should be indicated in a ShadowSpawn.exe command argument. how should the ShadowSpawn.exe executed AAM_auto_backup.py know whether to perform full or incremental backup? document everything applicable; update code as needed. [11/3/2020 10:01 AM CST] (NOTE [11/3/2020 10:27 AM CST] is an empty string argument allowed? if so, "the lastest AWS S3 full-backup date and time string" and "the lastest AWS S3 incremental-backup date and time string" can be used to determine whether to perform full or incremental backup. I must perform the empty string argument test first before moving on.)

3.4.2. command arguments and format

[10/15/2020 12:47 PM CST] [10/16/2020 5:06 PM CST]

1. (program execution settings)
    1.1. [argument 1] `regular-backup` or `incremental-only-backup`. (operation type)
    1.2. [argument 2] `ShadowSpawn` (execution agent type. the valid values are 'user' and 'ShadowSpawn'.)
    1.3. [argument 3] the latest AWS S3 full-backup date and time string.
    1.4. [argument 4] the latest AWS S3 incremental-backup date and time string.
    1.5. [argument 5] the backup execution start date and time string.
2. (backup source settings)
    2.1. [argument 6] local backup source directory full path.
    2.2. [argument 7] [10/6/2020 5:29 PM CST] an argument: shadow copy drive name (letter)
3. (backup destination settings)
    3.1. [argument 8] (backup destination types) `local`~~`/proximity(use which one?)/intranet?/non-Cloud?`~~ | ~~`Cloud/Internet?/`~~`AWS` ~~|both~~ (FINAL DECISION [10/16/2020 4:01 PM CST] use flags with | as the separator. the supported flags are 'local' and 'AWS') ([10/14/2020 8:10 AM CST] an argument to indicate the backup destination types: local or network-attached data storage, AWS S3 Glacier Deep Archive, or both.) ([10/16/2020 3:52 PM CST] how about just using 'local' to denote all the data storages that are physically nearby, such as internal data storage, attached data storage such as USB hard drive, network-attached data storage, and data storage on a LAN? I'm very strongly leaning toward that; actually, I'll use this. search "how distant can Wide area network can be?" online. https://www.apposite-tech.com/blog/whats-difference-metropolitan-area-network-man-wide-area-network-wan/ https://en.wikipedia.org/wiki/Wide_area_network) [this argument is also used in the incremental-only backup AAM Auto Backup command] ([10/10/2020 12:51 PM CST] AAM Auto Backup should take command arguments for making incremental-only backups to USB drive only, AWS S3 only, or both USB drive and AWS S3.) ([10/9/2020 10:16 AM CST] an argument: incremental backup to AWS only) ([10/14/2020 7:55 AM CST] incremental-only backup destinations argument: local or network-attached data storage (done using Robocopy), AWS S3 Glacier Deep Archive (Done using AWS Python API), or both.)
    3.2. [argument 9] (a local ~~or network-attached~~ data storage backup-destination directory full path) ([10/6/2020 3:48 PM CST] an argument: the local (non-Cloud) backup destination root folder location full path.) (TO DO [10/8/2020 1:47 PM CST] support for auto-backup to AWS S3 Glacier Deep Archive only. if this argument is empty or blank (i.e. ""), then local backup is skipped. this is for backing up the login infos on the ultra-security laptop.) ([10/15/2020 1:32 PM CST] user-specified data replication destination directory full path.)
    3.3. [argument 10] [10/14/2020 8:08 AM CST] an argument to indicate whether to leave the files in the backup (data replication) destination that are no longer present in the

backup source. (very easy and fast to implement, using the single Robocopy argment.) ~~(FINAL DECISION [11/2/2020 4:28 PM CST] 'keep' leaves the files in the backup destination that do not exist in the backup source; 'delete' deletes them.)~~ (FINAL DECISION [11/27/2020 1:54 PM CST] should be True or False.)

3.4. (backup-destination AWS S3 info)

~~3.4.1. [10/7/2020 8:32 AM CST] an argument: backup-only AWS account AWS CLI program profile name (used in executing the backup AWS CLI commands). (no need. AWS Python API, Boto3, will be used for authentication in each backup [and restoration] operation. [10/16/2020 1:26 PM CST])~~

3.4.2. [argument 11] [10/9/2020 8:53 AM CST] an argument: backup encryption AES-256 key.

3.4.3. [argument 12] (UPDATE [10/28/2020 3:40 PM CST] a unique backup-source location name) [10/10/2020 1:05 PM CST] ~~AWS S3 bucket name for the backup.~~ better yet, the unique backup file tag or prefix. all stored in the same AWS S3 bucket. ([10/15/2020 10:08 AM CST] a unique (backup-set)/~~computer~~ prefix)

3.4.4. [argument 13] AWS S3 backup files directory full path.

3.4.5. [argument 14] [11/11/2020 5:02 PM CST] Force full backup. The value must be "yes" or "no". The value applies only when regular backup.

3.4.6. [argument 15] [12/21/2020 11:32 AM CST] backup scope. The value must be "full-backup" or "incremental-backup".

([10/15/2020 1:37 PM CST] [support in] backup package or form format specification for backup to local or network-attached storage—replication or multi-volume encrypted and compressed files.)

3.5. (future support) backup-tape data backup operation mode

[10/14/2020 12:26 PM CST]
for storing encrypted and compressed backup ZIP files in backup tapes.

4. Data-restoration Operation Modes

4.1. Data Restoration Operation Modes Overview and Usage Cases

[10/14/2020 1:49 PM CST]

4.1.1. Three main types of data restoration operations, and when to execute each

[10/14/2020 2:59 PM CST]

one-step AWS S3 data restoration

multi-step AWS S3 data restoration

(future support) backup-tape data restoration

4.1.2. How to restore data from AWS S3 Glacier Deep Archive using AAM Auto Backup

[10/7/2020 7:14 PM CST]

(complete this, after finalizing the AAM Auto Backup command formats. [10/14/2020 11:58 AM CST])

4.1.2.1. multi-step data restoration

[10/17/2020 3:22 PM CST] to perform a multi-step data restoration, execute each process in the multi-step data restoration in the right order (i.e. restore to AWS S3 from AWS S3 Glacier Deep Archive, check AWS S3 restoration status, download to local computer from AWS S3, uncompress all the downloaded full and incremental backup files on the local computer)

4.1.2.2. one-step data restoration

[10/17/2020 3:22 PM CST] to perform a one-step data restoration, simply execute a one-step data restoration AAM Auto Backup command.

4.1.2.3. what to do if the upload-only AWS IAM user account credential has been compromised, and the hacker uploaded files

[10/8/2020 4:57 PM CST]
manually delete all the files in AWS S3 (via web console or AWS CLI). then run AAM Auto Backup for data restoration.

the following AWS CLI command can be used to delete files (if using an AWS IAM account with the capability to delete AWS S3 objects).
xx (complete this! [10/14/2020 12:00 PM CST])

4.2. One-step AWS S3 Data Restoration Operation Mode

Restore to local computer in one step

4.2.1. operation mode overview

[10/15/2020 1:10 PM CST]

TO DO [10/30/2020 7:16 PM CST]
make sure to output every AWS S3 backup objects used, both full and incremental, in both terminal and local restoration log.

resolve the backup source issue when restoring data from AWS S3. what to do about that?
[10/30/2020 6:49 PM CST]

```
    #TO DO [10/30/2020 6:30 PM CST]
    #hold on.  an issue to resolve first before coding.
    #when restoring, AAM Auto Backup does not know about the backup sources,
    #and it shouldn't care about the backup sources in the backup
    #instruction file(s).  so, what to do about that?
    #how about simply getting the latest full backup with complete upload
    #for all backup sources,
    #then using the incremental backups of that full backup?
    #I'll probably do that.  _QueryBackupAwsS3BucketForDataRestoration()
    #might have its own AWS S3 code.
```

(AN ADDITIONAL ISSUE TO ADDRESS [10/26/2020 7:43 AM CST] AAM Auto Backup instruction plain text file content change, multiple AWS full backups in the same month, and force AWS full backup option.)
NOTHING TO ADDRESS [11/20/2020 4:14 PM CST] AAM Auto Backup simply uses the latest completed-upload AWS S3 full backup and its incremental backups. from the AWS S3 data restoration point of view, it is completely irrelevant whether there are more than one AWS S3 full backup in the same month.

[10/8/2020 9:59 PM CST]
this process executes all the processes in the multi-step AWS S3 data restoration.

[10/6/2020 10:39 AM CST]
to restore to the local computer, first restore and download all the complete-upload backup files of the latest month and date and time in AWS S3 Glacier Deep Archive (i.e. requires the latest backup AWS S3 objects querying using ~~[the appropriate AWS CLI command]~~/[AWS Python API]). then, decompress each backup file in the ascending date order, with overwrite (i.e. maximum of 31 or more files to decompress since full backup is done at the 1st of every month, or at the earliest backup date of the month if there was no backup program running at the 1st of the month). at the end, (optionally) delete all the downloaded AWS S3 objects.
(great! this is so damn great!!!!!! I am precisely on the right track! I'll simply keep on executing and improving all my plans NO MATTER WHAT!!!!!! [10/6/2020 10:42 AM CST])

[10/15/2020 2:54 PM CST]
When performing an AWS S3 data restoration operation, AAM Auto Backup queries the backup AWS S3 bucket object list to retrieve the latest completely-uploaded full backup AWS S3 object names, and all the names of all the completely-uploaded incremental-backup AWS S3 objects of the latest completely-uploaded AWS S3 full backup.

Using the retrieved list of the latest completely-uploaded backup AWS S3 object names, AAM Auto Backup restores those AWS S3 objects, downloads the restored AWS S3 objects, then decompresses the downloaded AWS S3 backup files to the user-specified AWS S3 backup restoration directory.

AAM Auto Backup does not use the last AWS S3 backup dates and times plain text file in AWS S3 data restoration, since AWS S3 data restoration will almost always done on a computer without the text file (due to the main work computer being broken or missing), and to ensure that AWS S3 data restoration is done correctly using the latest data backup AWS S3 objects.

([10/15/2020 3:03 PM CST] look at the above! so simple and easy compared to the backup operations! good, great!!!! I'm precisely on the right track!!!! I'll simply keep on executing and improving all my plans, NO MATTER WHAT!!!!!!!)

[10/5/2020 8:10 PM CST] the program keeps on running with 30 minute sleep in each cycle of checking the AWS S3 Glacier Deep Archive object restoration statuses, and downloads and restores the backup data to the local computer using just one AAM Auto Backup command execution.

[10/5/2020 8:15 PM CST] displays the restoration initiation time in the terminal at the start of the program.

[10/9/2020 11:19 AM CST] email notification of backup report via AWS SES.

[1/12/2021 5:37 PM CST] note that this operation can be executed again while the data-backup AWS S3 objects are being restored (such as after the automatic reboot of the computer). AAM Auto Backup correctly handles such case.

4.2.2. log and email-report outputs
[10/13/2020 4:25 PM CST]
operation summary. (operation type, etc. the AWS S3 Glacier Deep Archive backup files used.)
errors.
ShadowSpawn.exe execution output.

4.2.3. command arguments and format

4.2.3.1. [11/10/2020 12:54 PM CST]

1. (program execution settings)
    1.1. [argument 1] `one-step-aws-data-restoration` (operation type)
    1.2. [argument 2] (log directory full path) ([10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).)
    1.3. [argument 3] [10/14/2020 9:38 AM CST] an argument to indicate whether to send email report or not.
    1.4. [argument 4] email sender name [10/18/2020 5:54 PM CST]
    1.5. [argument 5] email sender address [10/18/2020 5:54 PM CST]
    1.6. [argument 6] email recipient address [10/18/2020 5:42 PM CST]
~~2.~~ (restoration source setting[s?]) ~~([10/9/2020 5:18 PM CST] update the relevant info in the AAM Auto Backup document for providing the AWS CLI profile name argument for the auto-restoration.) (DESIGN CHANGE [10/16/2020 5:32 PM CST] AWS IAM user credential directly used via the latest AWS Python API, Boto3.)~~
    2.1. [argument 7] [10/15/2020 2:58 PM CST] AWS S3 object restoration period in ~~hours,~~ days (this~~), weeks, or months—whatever AWS allows~~.
    2.2. [argument 8] [10/9/2020 8:53 AM CST] an argument: backup decryption AES-256 key.
    2.3. [argument 9] [10/15/2020 1:22 PM CST] the backup AWS S3 bucket name.
    2.4. [argument 10] (UPDATE, FINAL DECISION [11/10/2020 10:19 AM CST] a unique backup-source location name.) ~~(UPDATE [10/30/2020 1:16 PM CST] one or more unique backup-source location names, in the Python list format. e.g. "['test', 'test2']")~~ (UPDATE [10/28/2020 3:42 PM CST] a unique backup-source location name) [10/10/2020 1:05 PM CST] ~~AWS S3 bucket name for the backup.~~ better yet, the unique backup file tag or prefix. all stored in the same AWS S3 bucket. ([10/15/2020 10:08 AM CST] a unique (backup-set)/~~computer~~ prefix)
    2.5. [argument 11] [10/14/2020 9:41 AM CST] an argument for the restoration AWS IAM user access key ID
    2.6. [argument 12] [10/14/2020 9:41 AM CST] an argument for the restoration AWS IAM user secret access key
3. (restoration destination settings)
    3.1. [argument 13] [10/15/2020 3:00 PM CST] AWS S3 data restoration destination directory full path.

4.2.3.2. DEPRECATED [11/10/2020 12:54 PM CST]

[11/10/2020 10:08 AM CST] [11/10/2020 12:28 PM CST]

4. (program execution settings)
    4.1. [argument 1] `one-step-aws-data-restoration` (operation type)
    4.2. [argument 2] (log directory full path) ([10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).)
    4.3. [argument 3] [10/14/2020 9:38 AM CST] an argument to indicate whether to send email report or not.
    4.4. [argument 4] email sender name [10/18/2020 5:54 PM CST]
    4.5. [argument 5] email sender address [10/18/2020 5:54 PM CST]
    4.6. [argument 6] email recipient address [10/18/2020 5:42 PM CST]

~~5.~~ (restoration source setting[s?]) ~~([10/9/2020 5:18 PM CST] update the relevant info in the AAM Auto Backup document for providing the AWS CLI profile name argument for the auto-restoration.) (DESIGN CHANGE [10/16/2020 5:32 PM CST] AWS IAM user credential directly used via the latest AWS Python API, Boto3.)~~
    5.1. [argument 7] [10/15/2020 2:58 PM CST] AWS S3 object restoration period in hours, day, weeks, or month—whatever AWS allows.
    5.2. [argument 8] [10/9/2020 8:53 AM CST] an argument: backup decryption AES-256 key.
    5.3. [argument 9] [10/15/2020 1:22 PM CST] the backup AWS S3 bucket name.
    5.4. [argument 10] (UPDATE, FINAL DECISION [11/10/2020 10:19 AM CST] a unique backup-source location name.) ~~(UPDATE [10/30/2020 1:16 PM CST] one or more unique backup-source location names, in the Python list format. e.g. "['test', 'test2']")~~ (UPDATE [10/28/2020 3:42 PM CST] a unique backup-source location name) [10/10/2020 1:05 PM CST] ~~AWS S3 bucket name for the backup.~~ better yet, the unique backup file tag or prefix. all stored in the same AWS S3 bucket. ([10/15/2020 10:08 AM CST] a unique (backup-set)/~~computer~~ prefix)
    5.5. [argument 11] [10/14/2020 9:41 AM CST] an argument for the restoration AWS IAM user access key ID
    5.6. [argument 12] [10/14/2020 9:41 AM CST] an argument for the restoration AWS IAM user secret access key
    5.7. [argument xx] AWS S3 archive object query starting year.
    5.8. [argument xx] AWS S3 archive object query starting month.
    5.9. [argument xx] the number of months to backtrack at the maximum (36 maximum—if smaller than 1 or bigger than 36, error display and quit).

6. (restoration destination settings)
    6.1. [argument 13] [10/15/2020 3:00 PM CST] AWS S3 data restoration destination directory full path.

4.2.3.3. DEPRECATED [11/10/2020 10:08 AM CST]

[10/15/2020 12:47 PM CST]

7. (program execution settings)
    7.1. [argument 1] `one-step-aws-data-restoration` (operation type)

    7.2. [argument 2] (log directory full path) ([10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).)
    7.3. [argument 3] [10/14/2020 9:38 AM CST] an argument to indicate whether to send email report or not.
    7.4. [argument 4] email sender name [10/18/2020 5:54 PM CST]
    7.5. [argument 5] email sender address [10/18/2020 5:54 PM CST]
    7.6. [argument 6] email recipient address [10/18/2020 5:42 PM CST]

~~8.~~ (restoration source setting[s?]) ~~([10/9/2020 5:18 PM CST] update the relevant info in the AAM Auto Backup document for providing the AWS CLI profile name argument for the auto-restoration.) (DESIGN CHANGE [10/16/2020 5:32 PM CST] AWS IAM user credential directly used via the latest AWS Python API, Boto3.)~~
    8.1. [argument 7] [10/15/2020 2:58 PM CST] AWS S3 object restoration period in hours, day, weeks, or month—whatever AWS allows.
    8.2. [argument 8] [10/9/2020 8:53 AM CST] an argument: backup decryption AES-256 key.
    8.3. [argument 9] [10/15/2020 1:22 PM CST] the backup AWS S3 bucket name.
    8.4. [argument 10] (UPDATE [10/30/2020 1:16 PM CST] one or more unique backup-source location names, in the Python list format. e.g. "['test', 'test2']") (UPDATE [10/28/2020 3:42 PM CST] a unique backup-source location name) [10/10/2020 1:05 PM CST] ~~AWS S3 bucket name for the backup.~~ better yet, the unique backup file tag or prefix. all stored in the same AWS S3 bucket. ([10/15/2020 10:08 AM CST] a unique (backup-set)/~~computer~~ prefix)
    8.5. [argument 11] [10/14/2020 9:41 AM CST] an argument for the restoration AWS IAM user access key ID
    8.6. [argument 12] [10/14/2020 9:41 AM CST] an argument for the restoration AWS IAM user secret access key

9. (restoration destination settings)
    9.1. [argument 13] [10/15/2020 3:00 PM CST] AWS S3 data restoration destination directory full path.

4.3. Multi-step AWS S3 Data Restoration Operation Modes

FINAL DECISION [10/17/2020 3:37 PM CST] no log and email report in all of the multi-step AWS S3 data restoration processes.

FINAL DECISION [10/16/2020 7:59 PM CST] it is not worth the effort to create a separate command format for each operation mode in the multi-step AWS S3 data restoration, since the multi-step AWS S3 data restoration will be used for testing and emergency purposes only. I'll reuse the one-step AWS S3 data restoration command format with the different operation type. think about the actual use cases of the operation modes in 'Multi-step AWS S3 Data Restoration Operation Modes'. any use other than testing? no. think about what to do about this. decide and document whether it is worth creating customized command arguments and formats for those operation modes. I'm think it's definitely not worth the effort. [10/16/2020 5:37 PM CST] FINAL DECISION [10/17/2020 1:17 PM CST] reuse the one-step AWS S3 data restoration operation command arguments and format. designing the custom command arguments and format for each process in the multi-step AWS S3 data restoration, is not worth the effort.

4.3.1. Restore to AWS S3 from AWS S3 Glacier Deep Archive

4.3.1.1. operation mode overview

[10/15/2020 1:10 PM CST]

[10/17/2020 12:53 PM CST] use the latest AWS Python API (Boto3)
~~[10/8/2020 9:55 PM CST] CLI program(s) to use in this process~~
~~AWS CLI program (configured with the local-data backup uploader AWS IAM user profile)~~

[10/8/2020 7:56 AM CST] use the bulk restoration mode for the cheapest cost. find out and document how to do this via the ~~(AWS CLI program)/~~(the latest AWS Python API).

~~[10/9/2020 5:13 PM CST] at the start of the software process, create the AWS CLI program admin profile. (if the profile can be created with a single AWS CLI command without on-screen user input.)~~

[10/13/2020 5:30 PM CST] **AAM Auto Backup queries the backup AWS S3 bucket list(s) to decide which backup set to restore: it restores the latest backup set with the completely uploaded full backup (i.e. the upload completion AWS S3 bucket object is present). AAM Auto Backup reports in terminal and email which backup set in the backup AWS S3 bucket was used for data restoration.**

4.3.1.2. Decision: using the AAM Auto Backup instruction plain text file

[10/13/2020 4:50 PM CST]
(to use the text file content or not? require the user to supply the backup info in the command arguments, or not? think and decide on this, and then implement the decision!)

FINAL DECISION [11/12/2020 2:14 PM CST] the AAM Auto Backup instruction plain text file is not required whatsoever in AWS S3 data restoration.

4.3.1.3. command arguments and format

[10/15/2020 12:47 PM CST] [10/17/2020 1:18 PM CST]
use the one-step AWS S3 data restoration operation command arguments and format.
`restore-to-aws-s3-from-aws-s3-glacier-deep-archive` operation type.

[10/17/2020 12:56 PM CST] an argument: restoration period

~~[10/9/2020 5:18 PM CST]~~
~~update the relevant info in the AAM Auto Backup document for providing the AWS CLI profile name argument for the auto-restoration.~~
~~UPDATE [10/9/2020 5:43 PM CST] the profile name probably isn't needed. only the programmatic access credential is needed. the profile name can be anything—AAM Auto Backup can use some unique name.~~

[10/14/2020 9:41 AM CST]
an argument for the restoration AWS IAM user access key ID
an argument for the restoration AWS IAM user secret access key

4.3.1.4. [OUTDATED, NOT USED][10/17/2020 12:57 PM CST] AWS CLI program admin profile creation and deletion for restoration

[10/9/2020 5:13 PM CST]
Search "aws cli how to list profiles" online
Search "aws cli how to set default profile" online
Search "aws cli how to remove profile" online

[10/9/2020 9:57 PM CST]
search "aws cli reference" online.
https://docs.aws.amazon.com/cli/latest/reference/
https://docs.aws.amazon.com/cli/latest/reference/configure/index.html
https://docs.aws.amazon.com/cli/latest/reference/configure/set.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-profiles.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-options.html

[10/9/2020 9:59 PM CST]
Search "aws cli how to remove profile" online
https://docs.aws.amazon.com/cli/latest/reference/alexaforbusiness/delete-profile.html
https://docs.aws.amazon.com/cli/latest/reference/opsworks/delete-user-profile.html
https://docs.aws.amazon.com/cli/latest/reference/iam/delete-login-profile.html
https://stackoverflow.com/questions/46319880/how-do-i-clear-the-credentials-in-aws-configure

[10/9/2020 10:01 PM CST]
Search "aws cli how to set default profile" online
https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-profiles.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-envvars.html
https://stackoverflow.com/questions/31012460/how-do-i-set-the-name-of-the-default-profile-in-aws-cli

```
export AWS_DEFAULT_PROFILE=user2
set AWS_DEFAULT_PROFILE=user2
```

4.3.2. Check AWS S3 restoration status

4.3.2.1. operation mode overview

[10/15/2020 1:10 PM CST]

[10/17/2020 12:53 PM CST] use the latest AWS Python API (Boto3)
~~[10/8/2020 9:55 PM CST] CLI program(s) to use in this process~~
~~AWS CLI program (configured with the local-data backup uploader AWS IAM user profile)~~

4.3.2.2. command arguments and format

[10/17/2020 1:22 PM CST]
use the one-step AWS S3 data restoration operation command arguments and format.
`check-aws-s3-restoration-status` operation type.

~~[10/9/2020 5:18 PM CST]~~
~~update the relevant info in the AAM Auto Backup document for providing the AWS CLI profile name argument for the auto-restoration.~~
~~UPDATE [10/9/2020 5:43 PM CST] the profile name probably isn't needed. the profile name can be anything—AAM Auto Backup can use some unique name.~~

[10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).

[10/14/2020 9:41 AM CST]
an argument for the restoration AWS IAM user access key ID
an argument for the restoration AWS IAM user secret access key

4.3.3. Download to local computer from AWS S3

4.3.3.1. operation mode overview

[10/15/2020 1:10 PM CST]

[10/17/2020 12:53 PM CST] use the latest AWS Python API (Boto3)
~~[10/8/2020 9:55 PM CST] CLI program(s) to use in this process~~
~~AWS CLI program (configured with the local-data backup uploader AWS IAM user profile)~~

[10/10/2020 1:00 PM CST] no need to delete the restored objects on AWS S3 since they will be auto-deleted after the user-specified period.

~~[10/9/2020 5:13 PM CST] at the end of the software process, delete the AWS CLI program admin profile. (if the profile can be created with a single AWS CLI command without on-screen user input.)~~

4.3.3.2. command arguments and format

[10/15/2020 12:47 PM CST] [10/17/2020 1:22 PM CST]
use the one-step AWS S3 data restoration operation command arguments and format.
`download-to-local-computer-from-aws-s3` operation type.

[10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).

[10/14/2020 9:41 AM CST]
an argument for the restoration AWS IAM user access key ID
an argument for the restoration AWS IAM user secret access key

4.3.4. Decompress all the downloaded full and incremental backup files on the local computer

~~Un~~compress all the downloaded full and incremental backup files on the local computer
Decompress all the downloaded full and incremental backup files on the local computer

4.3.4.1. operation mode overview

[10/15/2020 1:10 PM CST]

4.3.4.2. command arguments and format

[10/15/2020 12:47 PM CST] [10/17/2020 1:22 PM CST]
use the one-step AWS S3 data restoration operation command arguments and format.
`decompress-all-downloaded-backup-files` operation type.

[10/9/2020 8:53 AM CST] an argument: backup decryption AES-256 key.

~~[10/9/2020 5:18 PM CST]~~

~~update the relevant info in the AAM Auto Backup document for providing the AWS CLI profile name argument for the auto-restoration.~~
~~UPDATE [10/9/2020 5:43 PM CST] the profile name probably isn't needed. the profile name can be anything—AAM Auto Backup can use some unique name.~~

[10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).

4.4. (future support) backup-tape data restoration

[10/14/2020 12:26 PM CST]
for restoring backed-up data from encrypted and compressed backup ZIP files in backup tapes.

IV. Software Processes Master Design

1. Software process key features

[10/10/2020 5:18 PM CST]
use of (atomic) file locking for collaborating regular and incremental-only backups.

[10/10/2020 5:40 PM CST]
find and shadow-copy mount each common-most directory or folder using ShadowSpawn.

[10/10/2020 5:39 PM CST]
control which directories or folders get backed up, by using the directory or folder omission parameter, and the 7z.exe list file.

([10/13/2020 4:45 PM CST] God, I absolutely LOVE all the software implementation features: they enable everything I want in my own backup software!!!! great, so damn great!!!! I'm precisely on the right track!!!! **I'll simply keep on executing and improving all my plans, no matter what, no matter what, no matter what, no matter what, NO MATTER WHAT!!!!!!**)

1.1. CLI programs used by AAM Auto Backup

[10/17/2020 11:20 AM CST]

[10/8/2020 9:55 PM CST] CLI programs to use in this process

ShadowSpawn.exe (command line program for backing up files with shadow copy)
Robocopy (the advanced CLI copy program on Windows)
7z.exe (the 7-zip command line program)

AWS CLI program is not used. AWS Python API (Boto3) is used instead.
~~AWS CLI program (configured with the local-data backup uploader AWS IAM user profile)~~

1.1.1. Robocopy use

UPDATE [11/12/2020 2:17 PM CST] backup to AWS S3 does NOT use the last AWS S3 backup dates and times plain text file.
~~UPDATE [10/17/2020 11:22 AM CST] backup to AWS S3 does use the last AWS S3 backup dates and times plain text file, when it exists, to avoid incurring the AWS S3 query cost.~~
~~TO DO [10/7/2020 3:30 PM CST] at the end of each backup, create or replace (i.e. overwrite) a plain text file, named "lastest_backup_date_and_time.txt", in the AAM Auto Backup program file directory, for keeping and using the latest backup date and time.~~
UPDATE [10/7/2020 3:39 PM CST] lastest_backup_date_and_time.txt is not needed when using Robocopy. robocopy will do both full and incremental backup to the USB hard drive automatically: no distinction or difference in the robocopy command format used for both full and incremental backup. for backing up to AWS S3 Glacier Deep Archive, AAM Auto Backup gets the existing AWS S3 object names in the backup AWS S3 bucket (with a filter to get only the current month AWS S3 bucket objects), parses the names of the object names, detects the latest backup date, and detects whether the full backup for the current month exists or not. AAM Auto Backup uses the extracted AWS S3 bucket object name info to build the 7z.exe list file content.

TO DO [10/6/2020 1:47 PM CST] perform the backup destination validation check after each Robocopy command execution: build the list of the relative paths of all the files in the backup source, and check that those files exist in the backup destination also. if an error, include the error in the daily backup report email.

A DECISION [10/6/2020 11:02 AM CST] no file and sub-directory enumeration in Python code. Robocopy will handle all the copying: no custom code for doing copying. (hell yeah! great, so damn great!!!!!!!)

OUTDATED AND DEPRECATED BELOW [10/17/2020 12:48 PM CST]
how to omit all the non-backup folders in the backup directory in the Robocopy command by calling AAM Auto Backup again in the ShadowCopy command executed by AAM Auto Backup [10/7/2020 3:17 PM CST]
use the technique of calling AAM_auto_backup.py through ShadowSpawn with the proper argument to indicate the call or command-execution type.
AAM_auto_backup.py must check the shadow copy drive contents using os.listdir(), and dynamically create the batch file with the dynamically generated robocopy command.

1.1.2. Output capturing to include in logs and email reports

[10/17/2020 11:42 AM CST]

TO UPDATE [10/5/2020 1:20 PM CST] using the > in command, store and email the ShadowSpawn output!
~~UPDATE [10/6/2020 10:31 AM CST] actually, use the Robocopy log output parameter.~~ (this is pointless and unnecessarily complicating, since the 7z.exe output must be captured and included in logs and email reports as well.)

1.2. Use of an AWS IAM user that has write-only access to the backup AWS S3 bucket and AWS SES email sending

[10/13/2020 4:27 PM CST]

1.3. Use of an admin AWS IAM user for emergency data restoration

[10/13/2020 4:30 PM CST]

1.4. Dynamic creation and deletion of the Windows terminal batch files executed by ShadowSpawn

[10/14/2020 2:25 PM CST]
for the dynamic Robocopy command generation using the AAM Auto Backup instruction plain text file.

1.5. scheduled and manually-executed regular and incremental-only data backup, and data restoration, implementation key techniques

1.5.1. how to solve the incremental-only backup restoration problem

[10/10/2020 12:51 PM CST]
AAM Auto Backup should take command arguments for making incremental-only backups to USB drive only, AWS S3 only, or both USB drive and AWS S3.

[10/10/2020 12:54 PM CST] incremental-only backup to USB drive does not require anything new: just the same robocopy command that will do file overwriting on its own.

[10/10/2020 12:54 PM CST] the problem is the incremental-only backup to AWS S3, because it takes a full backup to restore the backup.

[10/10/2020 12:55 PM CST] **how about putting the full backup date in each incremental backup AWS S3 object name? the latest full backup and all of its incremental backup AWS S3 objects can be used for the restoration. each incremental backup to AWS S3 uses the latest full backup on AWS S3. all the info should be displayed on the terminal. I think this is the cleanest and easiest solution. in all likely chance, I'll do this!!!!!**

1.5.2. Use of an (atomic) file lock for regular and incremental-only backup timing control

file lock check and acquisition at the start of each backup process [10/13/2020 5:49 PM CST]

[10/13/2020 4:25 PM CST]
only one backup process running on a computer at any given time, be it regular or incremental-only.

if the file lock is present, the AAM Auto Backup regular backup process reports the status in the terminal, waits for 10 minutes, then tries again: it repeats the waiting and retrying for up to two hours, before quitting and reporting the status in the terminal and via email.

~~if the file lock is present, the AAM Auto Backup incremental-only backup process reports the status in the terminal, then quits. no email reporting in the incremental-only backup process, since the incremental-only backup process is always executed manually by the user.~~
if the file lock is present, the AAM Auto Backup incremental-only backup process reports the status in the terminal, then asks the user to whether perform the incremental-only backup after the completion of the current (regular) backup being executed. if the user chooses no (n), AAM Auto Backup quits. if the user chooses yes (y), the AAM Auto Backup incremental-only backup process reports the status in the terminal, waits for 10 minutes, then tries again: it repeats the waiting and retrying for up to two hours, before quitting and reporting the status in the terminal and via email. no email reporting in the incremental-only backup process, since the incremental-only backup process is always executed manually by the user.

[10/13/2020 5:02 PM CST] the user provides the on-going backup (i.e. file-lock) waiting period and the total on-going backup (i.e. file-lock) wait time for both the regular and the incremental-only backup processes.

[10/13/2020 5:10 PM CST] only one ShadowSpawn can run at any time, regardless of the shadow-copy directory difference. as such, ShadowSpawn is used one at a time, in sequence, and one (atomic) file lock is used for all the directories that are shadow copy mounted (sequentially) for backup.

[10/13/2020 4:40 PM CST]
Note that the use of an (atomic) file lock also solves the issue of incremental-only backup only restoration: because of the use of the lock, incremental-only backups are guaranteed to correspond to the right full backups.

1.5.2.1. file lock Python module reference

[10/13/2020 5:59 PM CST]
Search "Python how to place atomic lock on file" online.
https://stackoverflow.com/questions/489861/locking-a-file-in-python

Search "Python module FileLock" online.
https://en.m.wikipedia.org/wiki/File_locking#:~:text=When%20using%20lock%20files%2C%20care,creating%20it%20in%20the%20meantime.&text=System%20calls%20that%20create%20a,if%20the%20file%20already%20exists.

Search "Windows atomic file lock" online.
https://stackoverflow.com/questions/46074200/atomically-open-and-lock-file
Consider using other type of interprocess communication? A memory-based one?
Ok, so it looks like file locking is the way to go! The method for system-wide locking!

Search "python interprocess lock" online.
https://stackoverflow.com/questions/6931342/system-wide-mutex-in-python-on-linux
https://docs.python.org/3/library/fcntl.html#fcntl.lockf

1.5.2.2. OUTDATED AND NO-LONGER-VALID NOTES [10/15/2020 10:07 AM CST]

[10/10/2020 1:03 PM CST]

<u>manually executing incremental-only backup</u>
what happens if manual execution of incremental-only backup is done while Windows Task Scheduler executes the scheduled AAM Auto Backup execution, vice versa?

I'm thinking, if an instance of AAM Auto Backup is already running, another instance of AAM Auto Backup should quit, and do not perform the backup—for easier coding.

I don't know if two separate instances of ShadowSpawn can run and create two different shadow copy mounts with the same drive letter. probably not. so, when one AAM Auto Backup is already running, another instance of AAM Auto Backup will probably generate error due to the ShadowSpawn error.

however, to make sure, I want AAM Auto Backup to check another instance of it running, and if there's another instance already running, display the situation on the terminal, and do not run.

search "Python how to check another instance of Python program already running" online.

https://stackoverflow.com/questions/36799192/check-if-python-script-is-already-running/43370539
https://stackoverflow.com/questions/788411/check-to-see-if-python-script-is-running

search "Windows list running processes" online.
tasklist

[10/10/2020 1:31 PM CST] think about this issue some more.

[10/10/2020 1:34 PM CST] how about using two separate shadow copy mount drive names? one for regular backup, and another for incremental-only backup?
[10/10/2020 1:35 PM CST] or, how about not supporting incremental-only backup? if I don't support incremental-only backup, I might be able to use the Windows Task Scheduler CLI program to manually execute the backup Windows task, and have that as a shortcut on desktop.

[10/10/2020 1:36 PM CST] I'm leaning toward using two separate shadow copy mount drive names? one for regular backup, and another for incremental-only backup. the backup start date and time should be logged at the start of the AAM Auto Backup backup operation, and somehow be used to ensure correct backup and restoration?

I don't like the solution being too complex. there must be a very simple way that supports both regular backup and incremental backup! I will find a way!!!!

(think about the above issue? is that what I want?)

search "windows task scheduler CLI" online.
schtasks commands | Microsoft Docs
docs.microsoft.com › ... › schtasks

1.5.3. Use of an empty AWS S3 object to indicate the backup file batch upload completion

[10/13/2020 4:41 PM CST]
the absence of this file in the AWS S3 bucket, indicates that backup file batch (i.e. multi-volume encrypted and compressed zip files) upload is not yet completed.

this mechanism is used to ensure that all the backup files are restored and retrieved for data restoration, without any missing backup file.

[10/13/2020 6:00 PM CST]
"multivolume zip files upload completed indicator AWS S3 object".

1.5.4. ========DEPRECATED BELOW [11/12/2020 2:23 PM CST]=========

1.5.5. Use of a latest backup date time plain text file for the incremental backup to AWS S3 (Glacier Deep Archive)

[10/13/2020 5:26 PM CST]
the content of this text file, the latest backup date time, is used in building the 7-Zip list file(s).

FINAL DECISION [11/12/2020 2:23 PM CST] AWS S3 bucket query is used for greater accuracy and internet connection and AWS S3 operation check.

1.6. Auto-generation of dated and timed log files, and handling of no internet connection

[10/13/2020 5:34 PM CST] the user provides the auto-generated backup and restoration log directory full path in every applicable AAM Auto Backup command. the auto-generated dated and timed backup and restoration log plain text file name contains the date, time, and operation type (regular backup, incremental backup, or restoration).

when there is no internet connection, AAM Auto Backup cannot back up data to AWS S3, and it cannot send operation report email. in such a case, AAM Auto Backup gracefully handles the error, and quits, after saving the auto-generated dated and timed backup and restoration log plain text file, with the indication of the no internet connection error at the top of the log.

1.7. Use of the AWS Python API

[10/13/2020 3:48 PM CST] TO DO
In the software master design part, add the "Use of the AWS Python API" sector.

1.8. if not enough or wrong arguments in the AAM Auto Backup command, display an info on displaying the usage info

[10/13/2020 5:39 PM CST]
very likely, use `AAM_auto_backup.py -help` for displaying help.

([11/12/2020 5:30 AM CST] not implemented now. this will be implemented in .)

2. Software process key behaviors ([10/17/2020 3:57 PM CST] this document sector is probably not required. I might just delete it later on—for now, I'm not sure what to do with it exactly.)

[10/10/2020 5:25 PM CST]

2.1. regular backup

- xx

2.2. incremental-only backup

- xx

2.3. restoration

- xx

3. NOTES [10/10/2020 5:07 PM CST]

Incremental-backup only start timing

Before regular backup (regular backup not running)

At the same time as regular backup

After regular backup (regular backup not running)

After regular backup (regular backup running)

How about simply placing a lock on the latest backup date and time plain text file in an atomic operation? If the lock is on, don't execute AAM Auto Backup. I don't think this can be done reliably without an atomic lock.

Note that at the most, I would lose only one day of work or data.

How about using the atomic file lock, and not supporting incremental-only backup? For manually executing AAM Auto Backup whenever wanted?

When I am traveling or when I leave my office, I want the speed of incremental-only backup on my laptop or desktop computer. Also, when regular backup is happening, I should not turn off my main work laptop either unless it's a terrible emergency. How about using the incremental-only backup, and the atomic file lock, and not executing AAM Auto Backup when the file is locked, and reporting the status via AWS SES?

FINAL DECISION
Support incremental-only backup, because I want it.

I'm very heavily leaning toward using the atomic file lock, and not executing AAM Auto Backup when the file is locked.

I think the problem can be solved completely, if using two atomic-lock files, one for regular backup, one for incremental-only backup. AAM Auto Backup can take appropriate measures depending on the file-lock status(es).

AAM Auto Backup regular backup mode atomically locks the latest regular backup date time file. After completing the regular backup, AAM Auto Backup atomically locks the latest incremental backup date time file, and updates the latest incremental backup date time, only if the latest incremental backup date time in the file is older than the latest regular backup date time.

AAM Auto Backup incremental-only backup mode checks the lock status of the latest regular backup date time file. If the latest regular backup date time file is locked, and this is the 1st of the month when full backup is done, AAM Auto Backup uses the previous month's full backup in the AWS S3 incremental backup object name(s); if the latest regular backup date time file is not locked, and the file content indicates that this month's full backup is done, AAM Auto Backup uses the previous month's full backup in the AWS S3 incremental backup object name(s) (the latest regular backup date time file will have to be locked before checking, and AAM Auto Backup regular backup mode will have to handle that properly); otherwise, AAM Auto Backup uses the current month's full backup in the AWS S3 incremental backup object name(s). (Note that detecting the current month's full backup must be done flawlessly. I think the second line of the latest regular backup date time file should be the date time of the latest full backup, starting with "latest full backup date time: ".)

AAM Auto Backup auto-creates all the missing latest backup date time files.

AAM Auto Backup AWS S3 incremental-only backup mode displays an error, tells the user to perform full backup first, and quits if there is no latest regular backup date time file, since that indicates that no full backup AWS S3 object(s). This error is emailed via AWS SES also.

The restoration mode should use the regular backup latest date time file atomic locking for reading the file contents, and deciding which full backup on AWS S3 to use. If it is 1st day of a month, and the regular backup latest date time file is locked, the previous month's full backup should be used; otherwise, the current month's full backup should be used.

I also need to test whether a file gets listed in AWS S3 before the upload completion. I should use AWS S3 console and/or CLI program to list the S3 bucket objects while a big file is being uploaded to the Deep Archive. The result will decide what to do exactly when getting AWS S3 objects for restoration.
(NOTE an option is using the last regular and incremental-only backup data time files for deciding restoration AWS S3 objects to use. If no date time file or files, use what is on AWS S3. Or, require having those files.)

I need to test if two instances of shadowspawn can run with two different shadow copy mount drive names, and the same source directory.

I need to sort out and specify all of the above methods completely.

How about the above? I think it will work well. It may seem complex but I think it is the best solution yet, and likely the simplest solution.

This backup program is much more complex to develop than I envisioned, but it has absolutely everything I want! absolutely everything! I'm precisely on the right track! I'll simply keep on executing and improving all my plans no matter what!

(Ok, now my brain is working. I am getting what I want to get in the software design! I'll do some more research, to finalize the software process designs completely!)

Search "Python how to place atomic lock on file" online.
https://stackoverflow.com/questions/489861/locking-a-file-in-python

Search "Python module FileLock" online.
https://en.m.wikipedia.org/wiki/File_locking#:~:text=When%20using%20lock%20files%2C%20care,creating%20it%20in%20the%20meantime.&text=System%20calls%20that%20create%20a,if%20the%20file%20already%20exists.

An issue to resolve and document: backup and restoration done on two different computers. (In this case, the AWS S3 object list will have to be used with multivolume zip files upload completed indicator AWS S3 object? Also, making sure that an AWS S3 object is completely uploaded, if needed.) Regular backup and incremental-only backup are done on the same computers.

See if two shadowspawn commands can run simultaneously on the same source directory with two different shadow copy mount drive names. (what could be the actual usage case of this? I can't recall. [10/13/2020 11:03 AM CST] I need to read this entire note from the top to the bottom to find out what the potential application of this was! [10/13/2020 11:03 AM CST]) (in all likely chance, this was for running both regular and incremental-backup software processes at the same time, by two different instances of AAM Auto Backup.)

Check whether AWS S3 being uploaded to Deep Archive via AWS CLI program shows up or get listed while being uploaded.

Search "Windows atomic file lock" online.
https://stackoverflow.com/questions/46074200/atomically-open-and-lock-file
Consider using other type of interprocess communication? A memory-based one?
Ok, so it looks like file locking is the way to go! The method for system-wide locking!

Search "python interprocess lock" online.
https://stackoverflow.com/questions/6931342/system-wide-mutex-in-python-on-linux
https://docs.python.org/3/library/fcntl.html#fcntl.lockf

Ok, with the above, I think I got the AAM Auto Backup software processes design complete! I don't think any more is needed in that design!

3.1. tests and results [10/13/2020 11:06 AM CST]

([10/13/2020 11:06 AM CST] any other tests? read the entire notes in the parent section, and list all the tests to perform and document the results of!)

3.1.1. Dual ShadowSpawn execution test

[10/13/2020 12:23 PM CST]

3.1.1.1. test overview

See if two shadowspawn commands can run simultaneously on the same source directory with two different shadow copy mount drive names. (what could be the actual usage case of this? I can't recall. [10/13/2020 11:03 AM CST] I need to read this entire note from the top to the bottom to find out what the potential application of this was! [10/13/2020 11:03 AM CST]) (in all likely chance, this was for running both regular and incremental-backup software processes at the same time, by two different instances of AAM Auto Backup.)

3.1.1.2. test set 1

3.1.1.2.1. test procedure

[10/13/2020 12:25 PM CST]
open two admin cmd.exe windows.

execute each of the following ShadowSpawn commands, in each terminal, back to back, with no pause between the two executions. cd to "C:\Users\Allen\Documents\AAM Auto Backup" first in each admin terminal.

```
shadowspawn C:\temp\backup_source Q: dir q:\

shadowspawn C:\temp\backup_source R: dir r:\
```

3.1.1.2.2. test results

[10/13/2020 12:33 PM CST]

3.1.1.2.2.1. 1st ShadowSpawn execution

```
C:\Users\Allen\Documents\
\AAM Auto Backup>shadowspawn C:\temp\backup_source Q: dir q:\
ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing C:\temp\backup_source at Q:
Launching command: dir q:\
Aborting backup.
Dismounting device: Q:
Deleting snapshot.
There was an error calling CreateProcess. Process: dir q:\  Error: The system ca
nnot find the file specified.
 (Error number 2)

C:\Users\Allen\Documents\
\AAM Auto Backup>
```

3.1.1.2.2.2. 2nd ShadowSpawn execution

```
C:\Users\Allen\Documents\
\AAM Auto Backup>shadowspawn C:\temp\backup_source R: dir r:\
ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing C:\temp\backup_source at R:
Aborting backup.
There was a COM failure 0x80042316 - ShadowSpawn.cpp (281)

C:\Users\Allen\Documents\
\AAM Auto Backup>
```

3.1.1.2.3. conclusion

[10/13/2020 12:34 PM CST]

only one shadow copy mount of a target location can be created at a time.

I'll have to use the (atomic) file lock to manage running only one instance of ShadowSpawn at a time. this will make my job of coding AAM Auto Backup much easier, since there is no race condition to consider at all, when creating incremental backups, and hence the AWS S3 Glacier Deep Archive restoration coding will be very simple and straightforward.

3.1.1.3. test set 2

3.1.1.3.1. test procedure

[10/13/2020 12:25 PM CST]
open two admin cmd.exe windows.

execute each of the following ShadowSpawn commands, in each terminal, back to back, with no pause between the two executions. cd to "C:\Users\Allen\Documents\AAM Auto Backup" first in each admin terminal.

```
shadowspawn C:\temp\backup_source Q: basic_shadowspawn_test.bat

shadowspawn C:\temp\backup_source R: basic_shadowspawn_test2.bat
```

basic_shadowspawn_test.bat contains the following.

```
cd /D q:
dir
```

basic_shadowspawn_test2.bat contains the following.

```
cd /D r:
dir
```

3.1.1.3.2. test results

[10/13/2020 12:40 PM CST]

3.1.1.3.2.1. 1st ShadowSpawn execution

```
C:\Users\Allen\Documents\
\AAM Auto Backup>shadowspawn C:\temp\backup_source Q: basic_shadowspawn_test.bat

ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing C:\temp\backup_source at Q:
Launching command: basic_shadowspawn_test.bat

C:\Users\Allen\Documents\
\AAM Auto Backup>cd /D q:
```

```
Q:\>dir
 Volume in drive Q is OS
 Volume Serial Number is 4407-9849

 Directory of Q:\

10/06/2020  01:18 PM    <DIR>          .
10/06/2020  01:18 PM    <DIR>          ..
09/30/2020  06:17 PM             1,382 20200925 1240pm voiceover recording.aup
10/06/2020  09:20 AM    <DIR>          20200925 1240pm voiceover recording_data
10/06/2020  01:18 PM    <DIR>          test
10/06/2020  09:20 AM    <DIR>          voiceover recording 1
10/06/2020  09:20 AM    <DIR>          voiceover recording 2
10/06/2020  09:20 AM    <DIR>          voiceover recording 3
               1 File(s)          1,382 bytes
               7 Dir(s)  67,260,600,320 bytes free
Launched command finished with exit code: 0.
Shadowing successfully completed.

C:\Users\Allen\Documents\
\AAM Auto Backup>
```

3.1.1.3.2.2. 2nd ShadowSpawn execution

```
C:\Users\Allen\Documents\
\AAM Auto Backup>shadowspawn C:\temp\backup_source R: basic_shadowspawn_test2.ba
t
ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing C:\temp\backup_source at R:
Aborting backup.
There was a COM failure 0x80042316 - ShadowSpawn.cpp (281)

C:\Users\Allen\Documents\
\AAM Auto Backup>
```

3.1.1.3.3. conclusion

[10/13/2020 12:34 PM CST]
only one shadow copy mount of a target location can be created at a time.

I'll have to use the (atomic) file lock to manage running only one instance of ShadowSpawn at a time. this will make my job of coding AAM Auto Backup much easier, since there is no race condition to consider at all, when creating incremental backups, and hence the AWS S3 Glacier Deep Archive restoration coding will be very simple and straightforward.

3.1.2. AWS S3 Glacier Deep Archive upload response event test

[10/13/2020 12:24 PM CST]

3.1.2.1. test overview

Check whether AWS S3 being uploaded to Deep Archive via AWS CLI program shows up or get listed while being uploaded.

3.1.2.2. test procedure

copy a file that is one or two gigabytes to C:\Temp\test_upload\test.mp4.

In a web browser, log onto the backup-only AWS account.

In the web browser, display the AWS S3 console.

In a terminal, execute the following command. While the local test file is being uploaded to AWS S3 Glacier Deep Archive, refresh the S3 bucket contents in the web browser. Check and document whether the file being uploaded shows up before the upload is complete.

```
cd C:\Temp\test_upload\

aws --profile cloud-backup-uploader s3 cp test.mp4 s3://aam-data-
backup/ --storage-class DEEP_ARCHIVE
```

~~In another terminal, execute the following command to list the objects in the AWS S3 bucket while the upload is being done.~~ (actually, I won't do this, since I don't have the backup-only AWS account admin profile in an installed AWS CLI program. I don't feel the need to test this. [10/13/2020 1:05 PM CST])

Delete the test file after the test is complete.

3.1.2.3. test result

[10/13/2020 1:00 PM CST]

until the upload completion, the file does not show up in the AWS S3 web console.

3.1.2.4. conclusion

[10/13/2020 1:05 PM CST]

since AWS S3 does not list an object until it is fully uploaded, a backup-file object that is not yet completed upload will not be used in data restoration.

search "aws cli how to list bucket objects" online.
https://docs.aws.amazon.com/cli/latest/reference/s3/ls.html
https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3api/list-objects-v2.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-usage-pagination.html

the AWS CLI program returns only up to 1,000 objects at the maximum.

by using the list-objects-v2 sub-command with the --prefix and --max-items parameters, I can avoid using the AWS Python API (boto). AAM Auto Backup can call AWS CLI iteratively to first find the latest year of the backup AWS S3 objects, then the latest month, in the descending order, starting from the date of the restoration execution.

I think using the AWS Python API (boto) is an option now: I can simply copy and modify the Python code in the AAM Podcast-stats Reporter.

If I use the AWS Python API (boto), I might be able to skip setting up and deleting the AWS CLI program profile, because it may be possible to provide the AAM IAM user credential via the AWS API.

I first need to decide whether to use the AWS Python API (boto) or not in AAM Auto Backup: the decision will affect whether I need to do adding and deleting an AWS CLI program profile for the data restoration.

first, I need to find and read the login method in the AWS Python API. search "AWS Python API user manual" online.
https://docs.aws.amazon.com/pythonsdk/?id=docs_gateway
https://boto3.amazonaws.com/v1/documentation/api/latest/guide/index.html

(NOTE [10/13/2020 1:32 PM CST] it is absolutely critical to get the one-command data restoration right! so, I will get that absolutely right!!!! **I'll keep on executing and improving all my plans, no matter what, no matter what, no matter what, no matter what, no matter what, no matter what, NO MATTER WHAT!!!!!!!**)

Conclusion [10/13/2020 3:53 PM CST]
I can completely skip using the AWS CLI program, since the AWS Python API offers the Configuration object. In all likely chance, I won't use the AWS CLI program in data restoration. I might use it in data backup still, since it is very convenient. On second thought, I might not use the AWS CLI program at all, not even for backup, to make the coding consistent.
https://boto3.amazonaws.com/v1/documentation/api/latest/guide/credentials.html

3.1.3. Simultaneous ShadowSpawn execution test on two different directories

[10/13/2020 5:06 PM CST]

3.1.3.1. test overview

3.1.3.2. test procedure

[10/13/2020 5:06 PM CST]

open two admin cmd.exe windows.

execute each of the following ShadowSpawn commands, in each terminal, back to back, with no pause between the two executions. cd to "C:\Users\Allen\Documents\AAM Auto Backup" first in each admin terminal.

```
shadowspawn C:\Users\Allen\Documents Q:
basic_shadowspawn_test.bat

shadowspawn C:\temp R: basic_shadowspawn_test2.bat
```

basic_shadowspawn_test.bat contains the following.

```
cd /D q:
dir
```

basic_shadowspawn_test2.bat contains the following.

```
cd /D r:
dir
```

4. NOTES, original process outline, outdated and deprecated [10/17/2020 11:50 AM CST]

implementing the following software process is probably the best (and quickest) way to implement AAM Auto Backup.

1. Scan the content of the input text file that contains the last backup date and time—and store the last backup date and time in memory.
2. Scan the content of the input text file that specify the folders and files to backup—and build an in-memory list of the folders and files to backup.
3. (if needed, build a list of files to copy by comparing the created, modified, and/or accessed dates and times of each file to backup, with the last backup date and time.)
4. Backup to the USB hard drive first using hobocopy or shadowspawn. Copy (with overwrite) only the non-existing files and the files with the modification date and time newer than the last backup date and time; create every needed folder.

5. Backup to AWS S3 Deep Glacier.
    5.1. If full backup, use 7-zip to create a single full backup zip or 7-zip file with content encryption (no file name encryption, since the file name must have date and time info in it). create the backup file using the copy on the USB hard drive. upload the full backup zip or 7-zip file to AWS S3 Deep Glacier by executing an AWS CLI command.
    5.2. If incremental backup, copy all the files to backup using hobocopy or shadowspawn to a temporary folder with all the sub-folders. create a single incremental backup zip or 7-zip file with content encryption (no file name encryption, since the file name must have date and time info in it). create the backup file using the copy in the temporary folder. upload the full backup zip or 7-zip file to AWS S3 Deep Glacier by executing an AWS CLI command. delete the temporary folder.
6. use the AWS CLI program or AWS Python module (boto) to email a backup report with a copy of the ShadowSpawn CLI output. [10/5/2020 12:40 PM CST] (great, so damn great!!!!!)

([10/4/2020 6:47 PM CST] ok, I think the above will do. now, I need to test hobocopy and shadowspawn.)

V. Software Process Specs

1. Data-backup Process Specs

1.1. Regular-backup Process Spec

1.1.1. [11/20/2020 4:47 PM CST]

[10/20/2020 2:39 PM CST] the following adheres to the "operation mode overview" sub-section content in the "Regular-backup Operation Mode" section, and also to the [10/19/2020 11:19 AM CST] updates at the end of this document.

(TO DO [11/12/2020 8:41 PM CST] go through the current _ExecuteBackup(). add whatever is needed in the below software process spec.)

1. [Program execution preparation (initialization)]
    1.1. [Local backup log file creation, opening, and initialization] [10/21/2020 8:40 AM CST] because the ShadowSpawn output text file can be large (especially when doing full backup), it should be read by line, then written to the local backup log file by line. this means the local backup log file must be open at start, written to throughout the program execution, and closed at end. use a unique local backup log file name: Make sure that the local backup log file name does not exist already—postfix a unique counter number in the file base name if the file already exists. try opening the local backup log file with the unique name; if error, simply exit the program. after opening the local backup log

file, add the user-executed AAM Auto Backup command parameters to the local backup log file.

1.2. [File lock acquisition] Try acquiring the file lock with timeout.
    1.2.1. If the file lock is already acquired by another AAM Auto Backup program instance, keep on waiting using pause with sleep().
    1.2.2. If the successful file lock acquisition, record the backup execution starting date and time in a datetime data variable, report the backup execution starting time and date in terminal, and record them in the local backup log.
    1.2.3. If no file lock acquisition success within the maximum wait time, display the error in the terminal, append the error to the local backup log, email the backup failure report if the user specified the email report on, close the local backup log, and exit the program.

2. [AAM Auto Backup instruction plain text file processing] Process the AAM Auto Backup instruction plain text file. (NOTE [10/21/2020 4:41 PM CST] get the folders to exclude absolutely right; Robocopy must exclude every folder in the shadow-copy mounted directory, that should not be copied; Robocopy must not exclude folders in copying that must not be excluded.)
    2.1. Silently ignore the directories and non-wildcard files that do not exist, but include them in the local backup log file as invalid, nonexistent directories and non-wildcard files.
    2.2. build the list of the directories to shadow mount in memory, and to keep in memory the lists of folders and files to exclude per shadow-copy mounted directory.
    2.3. save in the local backup log the directories to backup, the directories to shadow mount, and folders and files to exclude.
    2.4. if an AAM Auto Backup instruction plain text file reading error, save the error in the local backup file and the email report (if applicable), email the error, close the local backup file, and terminate the program.

3. [AWS S3 connection testing and latest AWS S3 backup dates and times acquisition] If AWS S3 backup is on, perform the following actions. (NOTE why to do this after the AAM Auto Backup instruction plain text file processing [10/22/2020 7:33 PM CST] multiple backup sources or shadow-copy mounted directories. how to handle that properly? in all likely chance, check for the upload-completeness of full backup for every backup source or shadow-copy mounted directory. how? to do that, I need to process the AAM Auto Backup instruction plain text file first.)
    3.1. Query the backup AWS S3 bucket to get the latest backup-set AWS S3 bucket object name list with a completed-upload full backup (i.e. both full and incremental backup AWS S3 bucket objects in a backup period that can be used to completely restore the backed up data during that period). (UPDATE [10/22/2020 8:04 PM CST] make sure to properly handle more than one full backup in the backup AWS S3 bucket in the same month. also make sure to use the full-backup AWS S3 object set with all the backup sources completely uploaded.) (TO ADD [10/26/2020 7:28 PM CST] handle the backup-source change[s] in the AAM Auto Backup instruction plain text file.)
    3.2. If the AWS S3 bucket query is successful, save the latest completed-upload full and incremental backup dates and times in data variable(s) to pass them to the ShadowSpawn execution as command arguments later. Report the status in terminal, and append the dates and times to the local backup log file.

3.3. If no internet or AWS S3 connection, turn off the AWS S3 backup (mark a flag or a data variable for skipping the AWS S3 backup), report the error in terminal, record the error in the local backup log and the email report. If no local backup, exit the program after emailing the report (if applicable) and closing the local backup log file.

3.4. Perform automatic force backup processing, as needed; turn on force full backup, when applicable. If force full backup is required, and the backup operation type is incremental-only, display and save the error, email the backup report, and quit the program. [11/20/2020 5:06 PM CST]

4. [AWS S3 backup files directory creation] If AWS S3 backup, (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_source_location_nameYYYYMMDD_HHMMSS.SSSSS/~~backup_set_unique_prefix YYYYMMDD_HHMMSS.SSSSS~~, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.) (FINAL DECISION [10/22/2020 11:25 AM CST] I'll use c:\temp\ to minimize the path length. Windows allows only 260 max path length, not too much. if this were a commercial program, I would create the temporary folder in c:\users\(user name)\ on Windows or ~/ on Unix OSes.) if AWS S3 backup files directory creation error, output the error in terminal, include the error in the local backup log file and email report (if applicable), send the email report (if applicable), close the local backup log file, then exit the program.

5. [~~(Backup-source parent)/~~(Shadow-mount) directories processing] For each directory to shadow-copy mount for backup, perform the following.

5.1. using the lists of folders and files to exclude in memory, create a temporary plain text file (named shadow_copy_instruction.txt) in the AAM Auto Backup program directory that includes the lists of the folders and files to include and exclude for this shadow-copy mounted directory. if a temporary plain text file creation error, output the error in terminal, save the error in the local backup file and the email report (if applicable), email the error (if applicable), close the local backup file, and terminate the program.

5.2. delete the shadow copy command output file, and handle any deletion error. if a file deletion error, output the error in terminal, save the error in the local backup file and the email report (if applicable), email the error (if applicable), close the local backup file, and terminate the program.

5.3. Execute the ShadowSpawn command to execute AAM Auto Backup in the ShadowSpawn operation mode, with the output redirected to a temporary output file with a file name unique to this shadow-copy mounted directory (or programmatically read the output then save? FINAL DECISION [10/21/2020 1:22 PM CST] output to a file directly to use less memory) (FINAL DECISION [11/2/2020 3:51 PM CST] just use shadow_copy_command_output.txt for every ShadowSpawn.exe execution, since there is no reason to distinguish the terminal output text files by shadow-copy mounted directory. must delete this file first—and handle any deletion error). if a ShadowSpawn execution error, save the error in the local backup file and the email report (if applicable), email the report (if applicable), close the local backup file, and terminate the program. The AAM Auto Backup in the ShadowSpawn operation mode executes the following.

5.3.1. Process shadow_copy_instruction.txt to build a set of needed data variables.

5.3.2. If local backup is on, generate the Robocopy command in memory, using the temporary plain text file that includes the lists of folders and files to exclude for this shadow-copy mounted directory (i.e. shadow_copy_instruction.txt). if a temporary plain text file reading error, print() the error, and skip all the remaining steps.

5.3.3. If AWS S3 backup is on, scan the shadow copy mount to build the 7z.exe list file, using the temporary plain text file that includes the lists of folders and files to include and exclude for this shadow-copy mounted directory, and the latest completed-upload full and incremental backup dates and times. (also build the 7z.exe command in memory. [11/4/2020 2:40 PM CST]) (instead of using os.listdir() and os.path.isdir(), use the Python walk function to recursively scan each directory to backup in the shadow copy drive. [10/17/2020 12:14 PM CST]) ~~(If there is no file to incrementally backup, because there is no change or new file, report the no incremental backup status in terminal, and append the no file to incrementally backup to the log and email report in memory. [10/17/2020 4:09 PM CST] UPDATE [10/21/2020 8:10 AM CST] no access to the user-executed AAM Auto Backup data variables.)~~([10/20/2020 4:15 PM CST] use a class data variable and print() to indicate no files to incrementally backup. since the ShadowSpawn.exe output will be included in the log, this will be reported in the local backup log and the email report.) ([10/22/2020 12:48 PM CST] open the 7z.exe list file, write each line, then close the file, instead of generating the whole list in memory, in order to minimize the memory use.) if an I/O error, print() the error, and skip all the remaining steps (by existing the program using sys.exit()). ([11/20/2020 5:21 PM CST] if force full backup, do the full backup.)

5.3.4. auto-generate (overwrite or create) the regular-backup batch file with the cd command, Robocopy command (if applicable), and 7z.exe command (if applicable), in the AAM Auto Backup program directory. (dynamically create the batch file with the dynamically generated Robocopy command. [10/7/2020 3:16 PM CST]) (document creating the backup ZIP files in c:\temp to use less characters in the backup ZIP file full path. since the backup ZIP file names are long, it is prudent to use a short directory name. if c:\temp doesn't exist, create it in code. [10/20/2020 3:44 PM CST]) (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.) (FINAL DECISION [10/22/2020 11:43 AM CST] use c:\temp\ to minimize the path length.) if an I/O error, print() the error, and skip all the remaining steps.

5.3.5. execute the dynamically generated regular-backup batch file for doing the shadow backup. if an error, print() the error, and skip all the remaining steps in the ShadowSpawn AAM Auto Backup operation.

5.3.6. perform the local backup-destination validation check after each Robocopy command execution. in the shadow-copy mounted drive, recursively scan each directory included in the backup, and check that every non-excluded file exists in the backup destination with the proper (last modification, not last accessed,) date and time. if an error, use the print() function to output the error to the terminal; do not abort the validation operation when an error occurs, so that all the missing files are reported and saved in the ShadowSpawn output file. (this must be done using the shadow-copy mounted drive, not the original backup source directories; otherwise, an error can happen due to files that get created, modified, or deleted in the backup source while the shadow copy is in progress. [10/21/2020 8:24 AM CST])

5.3.7. remove the 7z.exe list file. [11/4/2020 2:24 PM CST]

5.3.8. print() regular-backup batch file execution success message (to use later for execution success validation).

5.4. Process ShadowSpawn command output text file.

5.4.1. Append the ShadowSpawn command output to the local backup log.

5.4.2. if no ShadowSpawn command output text file, or no regular-backup batch file execution success message, no Robocopy success message, or no 7z.exe success message in the ShadowSpawn command output text file, output the error in terminal, save the error in the local backup log file, email the error (if applicable), close the local backup log file, and exit the program.

5.4.3. If the ShadowSpawn command output contains no files to incrementally backup status, append that to the in-memory email report.

5.4.4. If the ShadowSpawn command output contains one or more local backup-destination validation failures, flag the validation failure report in email, and mark the backup operation as a failure to indicate in the report email title.

5.4.5. If any other error, flag the error to indicate in the email report title, flag no AWS S3 backup files upload, then break out of the ShadowSpawn execution loop to execute the remaining process and exit the program.

6. [AWS S3 backup files upload and deletion] if applicable, upload the encrypted and compressed backup files to AWS, and delete them. if AWS upload error, output the error in terminal, append the error in the local backup log, flag no email sending, abort the AWS S3 backup files upload, and delete the AWS S3 backup files. (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) [10/21/2020 9:10 AM CST] if AWS S3 backup files directory deletion error, output the error in terminal, include the error in the local backup log file and email report (if applicable).
7. [Temporary output files deletion] Delete the temporary output files. [10/21/2020 9:10 AM CST] if temporary output files deletion error, display the error in terminal, include the error in the local backup log file and email report (if applicable).
8. [Email report sending] If applicable, send the regular-backup email report (that does not include the ShadowSpawn.exe outputs). If email sending failure, save the error in the local backup log. (think and decide on not including the ShadowSpawn execution log in the email reports. no good will ever come out of emailing all the ShadowSpawn.exe, Robocopy and 7z.exe outputs in unencrypted emails. also, for the full backups, the Robocopy and maybe 7z.exe outputs will be too big to email—so emailing those outputs should be avoided. I'm very heavily leaning toward including the ShadowSpawn execution log in the AAM Auto Backup log only. think and decide on this, document the decision, and implement the decision!!!! [10/19/2020 8:30 AM CST])
9. [Program execution cleanup]

    9.1. [Local backup log file closing]
    9.2. [File lock release] Release the file lock.

(program exit)

1.1.2. DEPRECATED [11/20/2020 4:47 PM CST]

[10/26/2020 7:28 PM CST]

[10/20/2020 2:39 PM CST] the following adheres to the "operation mode overview" sub-section content in the "Regular-backup Operation Mode" section, and also to the [10/19/2020 11:19 AM CST] updates at the end of this document.

(TO DO [11/12/2020 8:41 PM CST] go through the current _ExecuteBackup(). add whatever is needed in the below software process spec.)

10. [Program execution preparation (initialization)]
    10.1. [Local backup log file creation, opening, and initialization] [10/21/2020 8:40 AM CST] because the ShadowSpawn output text file can be large (especially when doing full backup), it should be read by line, then written to the local backup log file by line. this means the local backup log file must be open at start, written to throughout the program execution, and closed at end. use a unique local backup log file name: Make sure that the local backup log file name does not exist already—postfix a unique counter number in the file base name if the file already exists. try opening the local backup log file with the unique name; if error, simply exit the program. after opening the local backup log file, add the user-executed AAM Auto Backup command parameters to the local backup log file.
    10.2. [File lock acquisition] Try acquiring the file lock with timeout.
        10.2.1. If the file lock is already acquired by another AAM Auto Backup program instance, keep on waiting using pause with sleep().
        10.2.2. If the successful file lock acquisition, record the backup execution starting date and time in a datetime data variable, report the backup execution starting time and date in terminal, and record them in the local backup log.
        10.2.3. If no file lock acquisition success within the maximum wait time, display the error in the terminal, append the error to the local backup log, email the backup failure report if the user specified the email report on, close the local backup log, and exit the program.
11. [AAM Auto Backup instruction plain text file processing] Process the AAM Auto Backup instruction plain text file. (NOTE [10/21/2020 4:41 PM CST] get the folders to exclude absolutely right; Robocopy must exclude every folder in the shadow-copy mounted directory, that should not be copied; Robocopy must not exclude folders in copying that must not be excluded.)
    11.1. Silently ignore the directories and non-wildcard files that do not exist, but include them in the local backup log file as invalid, nonexistent directories and non-wildcard files.
    11.2. build the list of the directories to shadow mount in memory, and to keep in memory the lists of folders and files to exclude per shadow-copy mounted directory.

11.3. save in the local backup log the directories to backup, the directories to shadow mount, and folders and files to exclude.

11.4. if an AAM Auto Backup instruction plain text file reading error, save the error in the local backup file and the email report (if applicable), email the error, close the local backup file, and terminate the program.

12. [AWS S3 connection testing and latest AWS S3 backup dates and times acquisition] If AWS S3 backup is on, perform the following actions. (NOTE why to do this after the AAM Auto Backup instruction plain text file processing [10/22/2020 7:33 PM CST] multiple backup sources or shadow-copy mounted directories. how to handle that properly? in all likely chance, check for the upload-completeness of full backup for every backup source or shadow-copy mounted directory. how? to do that, I need to process the AAM Auto Backup instruction plain text file first.)

12.1. Query the backup AWS S3 bucket to get the latest backup-set AWS S3 bucket object name list with a completed-upload full backup (i.e. both full and incremental backup AWS S3 bucket objects in a backup period that can be used to completely restore the backed up data during that period). (UPDATE [10/22/2020 8:04 PM CST] make sure to properly handle more than one full backup in the backup AWS S3 bucket in the same month. also make sure to use the full-backup AWS S3 object set with all the backup sources completely uploaded.) (TO ADD [10/26/2020 7:28 PM CST] handle the backup-source change[s] in the AAM Auto Backup instruction plain text file.)

12.2. If the AWS S3 bucket query is successful, save the latest completed-upload full and incremental backup dates and times in data variable(s) to pass them to the ShadowSpawn execution as command arguments later. Report the status in terminal, and append the dates and times to the local backup log file.

12.3. If no internet or AWS S3 connection, turn off the AWS S3 backup (mark a flag or a data variable for skipping the AWS S3 backup), report the error in terminal, record the error in the local backup log and the email report. If no local backup, exit the program after emailing the report (if applicable) and closing the local backup log file.

13. [AWS S3 backup files directory creation] If AWS S3 backup, (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_source_location_nameYYYYMMDD_HHMMSS.SSSSS/~~backup_set_unique_prefix YYYYMMDD_HHMMSS.SSSSS~~, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.) (FINAL DECISION [10/22/2020 11:25 AM CST] I'll use c:\temp\ to minimize the path length. Windows allows only 260 max path length, not too much. if this were a commercial program, I would create the temporary folder in c:\users\(user name)\ on Windows or ~/ on Unix OSes.) if AWS S3 backup files directory creation error, output the error in terminal, include the error in the local backup log file and email report (if applicable), send the email report (if applicable), close the local backup log file, then exit the program.

14. [~~(Backup-source parent)/~~(Shadow-mount) directories processing] For each directory to shadow-copy mount for backup, perform the following.

14.1. using the lists of folders and files to exclude in memory, create a temporary plain text file (named shadow_copy_instruction.txt) in the AAM Auto Backup program directory that includes the lists of the folders and files to include and exclude for this shadow-copy mounted directory. if a temporary plain text file creation error, output the error in terminal, save the error in the local backup file and the email report (if applicable), email the error (if applicable), close the local backup file, and terminate the program.

14.2. delete the shadow copy command output file, and handle any deletion error. if a file deletion error, output the error in terminal, save the error in the local backup file and the email report (if applicable), email the error (if applicable), close the local backup file, and terminate the program.

14.3. Execute the ShadowSpawn command to execute AAM Auto Backup in the ShadowSpawn operation mode, with the output redirected to a temporary output file with a file name unique to this shadow-copy mounted directory (or programmatically read the output then save? FINAL DECISION [10/21/2020 1:22 PM CST] output to a file directly to use less memory) (FINAL DECISION [11/2/2020 3:51 PM CST] just use shadow_copy_command_output.txt for every ShadowSpawn.exe execution, since there is no reason to distinguish the terminal output text files by shadow-copy mounted directory. must delete this file first—and handle any deletion error). if a ShadowSpawn execution error, save the error in the local backup file and the email report (if applicable), email the report (if applicable), close the local backup file, and terminate the program. The AAM Auto Backup in the ShadowSpawn operation mode executes the following.

14.3.1. Process shadow_copy_instruction.txt to build a set of needed data variables.

14.3.2. If local backup is on, generate the Robocopy command in memory, using the temporary plain text file that includes the lists of folders and files to exclude for this shadow-copy mounted directory (i.e. shadow_copy_instruction.txt). if a temporary plain text file reading error, print() the error, and skip all the remaining steps.

14.3.3. If AWS S3 backup is on, scan the shadow copy mount to build the 7z.exe list file, using the temporary plain text file that includes the lists of folders and files to include and exclude for this shadow-copy mounted directory, and the latest completed-upload full and incremental backup dates and times. (also build the 7z.exe command in memory. [11/4/2020 2:40 PM CST]) (instead of using os.listdir() and os.path.isdir(), use the Python walk function to recursively scan each directory to backup in the shadow copy drive. [10/17/2020 12:14 PM CST]) ~~(If there is no file to incrementally backup, because there is no change or new file, report the no incremental backup status in terminal, and append the no file to incrementally backup to the log and email report in memory. [10/17/2020 4:09 PM CST] UPDATE [10/21/2020 8:10 AM CST] no access to the user-executed AAM Auto Backup data variables.)~~([10/20/2020 4:15 PM CST] use a class data variable and print() to indicate no files to incrementally backup. since the ShadowSpawn.exe output will be included in the log, this will be reported in the local backup log and the email report.) ([10/22/2020 12:48 PM CST] open the 7z.exe list file, write each line, then close the file, instead of generating the whole list in memory, in order to minimize the memory use.) if an I/O error, print() the error, and skip all the remaining steps (by existing the program using sys.exit()).

14.3.4. auto-generate (overwrite or create) the regular-backup batch file with the cd command, Robocopy command (if applicable), and 7z.exe command (if applicable), in the AAM Auto Backup program directory. (dynamically create the batch file with the dynamically generated Robocopy command. [10/7/2020 3:16 PM CST]) (document creating the backup ZIP files in c:\temp to use less characters in the backup ZIP file full path. since the backup ZIP file names are long, it is prudent to use a short directory name. if c:\temp doesn't exist, create it in code. [10/20/2020 3:44 PM CST]) (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.) (FINAL DECISION [10/22/2020 11:43 AM CST] use c:\temp\ to minimize the path length.) if an I/O error, print() the error, and skip all the remaining steps.

14.3.5. execute the dynamically generated regular-backup batch file for doing the shadow backup. if an error, print() the error, and skip all the remaining steps in the ShadowSpawn AAM Auto Backup operation.

14.3.6. perform the local backup-destination validation check after each Robocopy command execution. in the shadow-copy mounted drive, recursively scan each directory included in the backup, and check that every non-excluded file exists in the backup destination with the proper (last modification, not last accessed,) date and time. if an error, use the print() function to output the error to the terminal; do not abort the validation operation when an error occurs, so that all the missing files are reported and saved in the ShadowSpawn output file. (this must be done using the shadow-copy mounted drive, not the original backup source directories; otherwise, an error can happen due to files that get created, modified, or deleted in the backup source while the shadow copy is in progress. [10/21/2020 8:24 AM CST])

14.3.7. remove the 7z.exe list file. [11/4/2020 2:24 PM CST]

14.3.8. print() regular-backup batch file execution success message (to use later for execution success validation).

14.4. Process ShadowSpawn command output text file.

14.4.1. Append the ShadowSpawn command output to the local backup log.

14.4.2. if no ShadowSpawn command output text file, or no regular-backup batch file execution success message, no Robocopy success message, or no 7z.exe success message in the ShadowSpawn command output text file, output the error in terminal, save the error in the local backup log file, email the error (if applicable), close the local backup log file, and exit the program.

14.4.3. If the ShadowSpawn command output contains no files to incrementally backup status, append that to the in-memory email report.

14.4.4. If the ShadowSpawn command output contains one or more local backup-destination validation failures, flag the validation failure report in email, and mark the backup operation as a failure to indicate in the report email title.

14.4.5. If any other error, flag the error to indicate in the email report title, flag no AWS S3 backup files upload, then break out of the ShadowSpawn execution loop to execute the remaining process and exit the program.

15. [AWS S3 backup files upload and deletion] if applicable, upload the encrypted and compressed backup files to AWS, and delete them. if AWS upload error, output the error in

terminal, append the error in the local backup log, flag no email sending, abort the AWS S3 backup files upload, and delete the AWS S3 backup files. (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) [10/21/2020 9:10 AM CST] if AWS S3 backup files directory deletion error, output the error in terminal, include the error in the local backup log file and email report (if applicable).

16. [Temporary output files deletion] Delete the temporary output files. [10/21/2020 9:10 AM CST] if temporary output files deletion error, display the error in terminal, include the error in the local backup log file and email report (if applicable).
17. [Email report sending] If applicable, send the regular-backup email report (that does not include the ShadowSpawn.exe outputs). If email sending failure, save the error in the local backup log. (think and decide on not including the ShadowSpawn execution log in the email reports. no good will ever come out of emailing all the ShadowSpawn.exe, Robocopy and 7z.exe outputs in unencrypted emails. also, for the full backups, the Robocopy and maybe 7z.exe outputs will be too big to email—so emailing those outputs should be avoided. I'm very heavily leaning toward including the ShadowSpawn execution log in the AAM Auto Backup log only. think and decide on this, document the decision, and implement the decision!!!! [10/19/2020 8:30 AM CST])
18. [Program execution cleanup]
    18.1. [Local backup log file closing]
    18.2. [File lock release] Release the file lock.

(program exit)

1.1.3. DEPRECATED [10/26/2020 7:28 PM CST]

[10/21/2020 8:44 AM CST]

[10/20/2020 2:39 PM CST] the following adheres to the "operation mode overview" sub-section content in the "Regular-backup Operation Mode" section, and also to the [10/19/2020 11:19 AM CST] updates at the end of this document.

19. [Program execution preparation (initialization)]
    19.1. [Local backup log file creation, opening, and initialization] [10/21/2020 8:40 AM CST] because the ShadowSpawn output text file can be large (especially when doing full backup), it should be read by line, then written to the local backup log file by line. this means the local backup log file must be open at start, written to throughout the program execution, and closed at end. use a unique local backup log file name: Make sure that the local backup log file name does not exist already—postfix a unique counter number in the file base name if the file already exists. try opening the local backup log file with the unique name; if error, simply exit the program. after opening the local backup log file, add the user-executed AAM Auto Backup command parameters to the local backup log file.
    19.2. [File lock acquisition] Try acquiring the file lock with timeout.

19.2.1. If the file lock is already acquired by another AAM Auto Backup program instance, keep on waiting using pause with sleep().

19.2.2. If the successful file lock acquisition, record the backup execution starting date and time in a datetime data variable, report the backup execution starting time and date in terminal, and record them in the local backup log.

19.2.3. If no file lock acquisition success within the maximum wait time, display the error in the terminal, append the error to the local backup log, email the backup failure report if the user specified the email report on, close the local backup log, and exit the program.

20. [AAM Auto Backup instruction plain text file processing] Process the AAM Auto Backup instruction plain text file. (NOTE [10/21/2020 4:41 PM CST] get the folders to exclude absolutely right; Robocopy must exclude every folder in the shadow-copy mounted directory, that should not be copied; Robocopy must not exclude folders in copying that must not be excluded.)

20.1. Silently ignore the directories and non-wildcard files that do not exist, but include them in the local backup log file as invalid, nonexistent directories and non-wildcard files.

20.2. build the list of the directories to shadow mount in memory, and to keep in memory the lists of folders and files to exclude per shadow-copy mounted directory.

20.3. save in the local backup log the directories to backup, the directories to shadow mount, and folders and files to exclude.

20.4. if an AAM Auto Backup instruction plain text file reading error, save the error in the local backup file and the email report (if applicable), email the error, close the local backup file, and terminate the program.

21. [AWS S3 connection testing and latest AWS S3 backup dates and times acquisition] If AWS S3 backup is on, perform the following actions. (NOTE why to do this after the AAM Auto Backup instruction plain text file processing [10/22/2020 7:33 PM CST] multiple backup sources or shadow-copy mounted directories. how to handle that properly? in all likely chance, check for the upload-completeness of full backup for every backup source or shadow-copy mounted directory. how? to do that, I need to process the AAM Auto Backup instruction plain text file first.)

21.1. Query the backup AWS S3 bucket to get the latest backup-set AWS S3 bucket object name list with a completed-upload full backup (i.e. both full and incremental backup AWS S3 bucket objects in a backup period that can be used to completely restore the backed up data during that period). (UPDATE [10/22/2020 8:04 PM CST] make sure to properly handle more than one full backup in the backup AWS S3 bucket in the same month. also make sure to use the full-backup AWS S3 object set with all the backup sources completely uploaded.)

21.2. If the AWS S3 bucket query is successful, save the latest completed-upload full and incremental backup dates and times in data variable(s) to pass them to the ShadowSpawn execution as command arguments later. Report the status in terminal, and append the dates and times to the local backup log file.

21.3. If no internet or AWS S3 connection, turn off the AWS S3 backup (mark a flag or a data variable for skipping the AWS S3 backup), report the error in terminal, record the error in the local backup log and the email report. If no local backup, exit the program after emailing the report (if applicable) and closing the local backup log file.

22. [AWS S3 backup files directory creation] If AWS S3 backup, (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.) (FINAL DECISION [10/22/2020 11:25 AM CST] I'll use c:\temp\ to minimize the path length. Windows allows only 260 max path length, not too much. if this were a commercial program, I would create the temporary folder in c:\users\(user name)\ on Windows or ~/ on Unix OSes.) if AWS S3 backup files directory creation error, output the error in terminal, include the error in the local backup log file and email report (if applicable), send the email report (if applicable), close the local backup log file, then exit the program.
23. [~~(Backup-source parent)/~~(Shadow-mount) directories processing] For each directory to shadow-copy mount for backup, perform the following.
    23.1. using the lists of folders and files to exclude in memory, create a temporary plain text file (named shadow_copy_instruction.txt) in the AAM Auto Backup program directory that includes the lists of the folders and files to include and exclude for this shadow-copy mounted directory. if a temporary plain text file creation error, output the error in terminal, save the error in the local backup file and the email report (if applicable), email the error (if applicable), close the local backup file, and terminate the program.
    23.2. Execute the ShadowSpawn command to execute AAM Auto Backup in the ShadowSpawn operation mode, with the output redirected to a temporary output file with a file name unique to this shadow-copy mounted directory (or programmatically read the output then save? FINAL DECISION [10/21/2020 1:22 PM CST] output to a file directly to use less memory). if a ShadowSpawn execution error, save the error in the local backup file and the email report (if applicable), email the report (if applicable), close the local backup file, and terminate the program. The AAM Auto Backup in the ShadowSpawn operation mode executes the following.
        23.2.1. If local backup is on, generate the Robocopy command in memory, using the temporary plain text file that includes the lists of folders and files to exclude for this shadow-copy mounted directory (i.e. shadow_copy_instruction.txt). if a temporary plain text file reading error, print() the error, and skip all the remaining steps.
        23.2.2. If AWS S3 backup is on, scan the shadow copy mount to build the 7z.exe list file, using the temporary plain text file that includes the lists of folders and files to include and exclude for this shadow-copy mounted directory, and the latest completed-upload full and incremental backup dates and times. (instead of using os.listdir() and os.path.isdir(), use the Python walk function to recursively scan each directory to backup in the shadow copy drive. [10/17/2020 12:14 PM CST]) ~~(If there is no file to incrementally backup, because there is no change or new file, report the no incremental backup status in terminal, and append the no file to incrementally backup to the log and email report in memory. [10/17/2020 4:09 PM CST] UPDATE [10/21/2020 8:10 AM CST] no access to the user-executed AAM Auto Backup data variables.)~~([10/20/2020 4:15 PM CST] use a class data variable and print() to indicate no files to incrementally backup. since the ShadowSpawn.exe output will be included in the log, this will be reported in the

local backup log and the email report.) ([10/22/2020 12:48 PM CST] open the 7z.exe list file, write each line, then close the file, instead of generating the whole list in memory, in order to minimize the memory use.) if an I/O error, print() the error, and skip all the remaining steps (by existing the program using sys.exit()).

23.2.3. auto-generate (overwrite or create) the regular-backup batch file with the cd command, Robocopy command (if applicable), and 7z.exe command (if applicable), in the AAM Auto Backup program directory. (dynamically create the batch file with the dynamically generated Robocopy command. [10/7/2020 3:16 PM CST]) (document creating the backup ZIP files in c:\temp to use less characters in the backup ZIP file full path. since the backup ZIP file names are long, it is prudent to use a short directory name. if c:\temp doesn't exist, create it in code. [10/20/2020 3:44 PM CST]) (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.) (FINAL DECISION [10/22/2020 11:43 AM CST] use c:\temp\ to minimize the path length.) if an I/O error, print() the error, and skip all the remaining steps.

23.2.4. execute the dynamically generated regular-backup batch file for doing the shadow backup. if an error, print() the error, and skip all the remaining steps in the ShadowSpawn AAM Auto Backup operation.

23.2.5. print() regular-backup batch file execution success message (to use later for execution success validation).

23.2.6. perform the local backup-destination validation check after each Robocopy command execution. in the shadow-copy mounted drive, recursively scan each directory included in the backup, and check that every non-excluded file exists in the backup destination with the proper (last modification, not last accessed,) date and time. if an error, use the print() function to output the error to the terminal; do not abort the validation operation when an error occurs, so that all the missing files are reported and saved in the ShadowSpawn output file. (this must be done using the shadow-copy mounted drive, not the original backup source directories; otherwise, an error can happen due to files that get created, modified, or deleted in the backup source while the shadow copy is in progress. [10/21/2020 8:24 AM CST])

23.3. Process ShadowSpawn command output text file.

23.3.1. Append the ShadowSpawn command output to the local backup log.

23.3.2. if no ShadowSpawn command output text file, or no regular-backup batch file execution success message, no Robocopy success message, or no 7z.exe success message in the ShadowSpawn command output text file, output the error in terminal, save the error in the local backup log file, email the error (if applicable), close the local backup log file, and exit the program.

23.3.3. If the ShadowSpawn command output contains no files to incrementally backup status, append that to the in-memory email report.

23.3.4. If the ShadowSpawn command output contains one or more local backup-destination validation failures, flag the validation failure report in email, and mark the backup operation as a failure to indicate in the report email title.

23.3.5. If any other error, flag the error to indicate in the email report title, flag no AWS S3 backup files upload, then break out of the ShadowSpawn execution loop to execute the remaining process and exit the program.

24. [AWS S3 backup files upload and deletion] if applicable, upload the encrypted and compressed backup files to AWS, and delete them. if AWS upload error, output the error in terminal, append the error in the local backup log, flag no email sending, abort the AWS S3 backup files upload, and delete the AWS S3 backup files. (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) [10/21/2020 9:10 AM CST] if AWS S3 backup files directory deletion error, output the error in terminal, include the error in the local backup log file and email report (if applicable).
25. [Temporary output files deletion] Delete the temporary output files. [10/21/2020 9:10 AM CST] if temporary output files deletion error, display the error in terminal, include the error in the local backup log file and email report (if applicable).
26. [Email report sending] If applicable, send the regular-backup email report (that does not include the ShadowSpawn.exe outputs). If email sending failure, save the error in the local backup log. (think and decide on not including the ShadowSpawn execution log in the email reports. no good will ever come out of emailing all the ShadowSpawn.exe, Robocopy and 7z.exe outputs in unencrypted emails. also, for the full backups, the Robocopy and maybe 7z.exe outputs will be too big to email—so emailing those outputs should be avoided. I'm very heavily leaning toward including the ShadowSpawn execution log in the AAM Auto Backup log only. think and decide on this, document the decision, and implement the decision!!!! [10/19/2020 8:30 AM CST])
27. [Program execution cleanup]
    27.1. [Local backup log file closing]
    27.2. [File lock release] Release the file lock.

(program exit)

1.1.4. DEPRECATED [10/21/2020 8:44 AM CST]

[10/20/2020 2:38 PM CST]

[10/20/2020 2:39 PM CST] the following adheres to the "operation mode overview" sub-section content in the "Regular-backup Operation Mode" section, and also to the [10/19/2020 11:19 AM CST] updates at the end of this document.

TO CONSIDER [10/21/2020 8:40 AM CST] because the ShadowSpawn output text file can be large (especially when doing full backup), it should be read by line, then written to the local backup log file by line. this means the local backup log file must be open at start, written to throughout the program execution, and closed at end.

28. [File lock acquisition] Try acquiring the file lock with timeout. If the file lock is already acquired by another AAM Auto Backup program instance, keep on waiting using pause with sleep(). If no file lock acquisition success within the maximum wait time, display the error in the terminal, email the backup failure report if the user specified the email report on, save the local backup log, and exit the program. After the successful file lock acquisition, report the backup execution starting time and date in terminal, and record them in the local backup log and the email report (if applicable).
29. [AWS S3 connection testing and latest AWS S3 backup dates and times acquisition] If AWS S3 backup is on, perform the following actions.
    29.1. Query the backup AWS S3 bucket to get the list of the latest AWS S3 bucket object name set with a completed-upload full backup.
    29.2. If the AWS S3 bucket query is successful, save the latest completed-upload full and incremental backup dates and times in data variable(s) to pass them to the ShadowSpawn execution as command arguments. Report the status in terminal, and append the dates and times to the local backup log file.
    29.3. If no internet or AWS S3 connection, turn off the AWS S3 backup (mark a flag or a data variable for skipping the AWS S3 backup), report the error in terminal, record the error in the local backup log and the email report. If no local backup, exit the program after emailing the report (if applicable) and saving the local backup log.
30. [AAM Auto Backup instruction plain text file processing] Process the AAM Auto Backup instruction plain text file, to build the list of the directories to shadow mount, and keep in memory the lists of folders and files to exclude per shadow-copy mounted directory.
31. [AWS S3 backup files directory creation] If AWS S3 backup, (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.)
32. [~~(Backup-source parent)/~~(Shadow-mount) directories processing] For each Robocopy directory to backup, perform the following.
    32.1. using the lists of folders and files to exclude in memory, in the AAM Auto Backup program directory, create a temporary plain text file that includes the lists of the folders and files to include and exclude, for this shadow-copy mounted directory.
    32.2. Execute the ShadowSpawn command to execute AAM Auto Backup in the ShadowSpawn operation mode, with the output redirected to a temporary output file with a file name unique to this shadow-copy mounted directory (or programmatically read the output then save?). The AAM Auto Backup in the ShadowSpawn operation mode executes the following.
        32.2.1. If local backup is on, generate the Robocopy command in memory, using the temporary plain text file that includes the lists of folders and files to exclude for this shadow-copy mounted directory.
        32.2.2. If AWS S3 backup is on, scan the shadow copy mount to build the 7z.exe list file, using the temporary plain text file that includes the lists of folders and files to include and exclude for this shadow-copy mounted directory, and the latest completed-upload full and incremental backup dates and times. (instead of using os.listdir() and os.path.isdir(), use the Python walk function to recursively scan each

directory to backup in the shadow copy drive. [10/17/2020 12:14 PM CST]) (~~If there is no file to incrementally backup, because there is no change or new file, report the no incremental backup status in terminal, and append the no file to incrementally backup to the log and email report in memory.~~ [10/17/2020 4:09 PM CST] UPDATE [10/21/2020 8:10 AM CST] no access to the user-executed AAM Auto Backup data variables.)([10/20/2020 4:15 PM CST] use print() to indicate no files to incrementally backup. since the ShadowSpawn.exe output will be included in the log, this will be reported in the local backup log and the email report.)

32.2.3. auto-generate (overwrite or create) the regular-backup batch file with the cd command, Robocopy command (if applicable), and 7z.exe command (if applicable), in the AAM Auto Backup program directory. (dynamically create the batch file with the dynamically generated Robocopy command. [10/7/2020 3:16 PM CST]) (document creating the backup ZIP files in c:\temp to use less characters in the backup ZIP file full path. since the backup ZIP file names are long, it is prudent to use a short directory name. if c:\temp doesn't exist, create it in code. [10/20/2020 3:44 PM CST]) (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.) (UPDATE [10/21/2020 8:17 AM CST] consider creating the temporary AWS S3 backup files directory in the AAM Auto Backup directory.)

32.2.4. execute the dynamically generated regular-backup batch file for doing the shadow backup.

32.2.5. perform the local backup-destination validation check after each Robocopy command execution. in the shadow-copy mounted drive, recursively scan each directory included in the backup, and check that every non-excluded file exists in the backup destination. if error, use the print() function to output the error to the terminal. (this must be done using the shadow-copy mounted drive; otherwise, an error can happen due to files that get included or deleted while the shadow copy is in progress. [10/21/2020 8:24 AM CST])

32.3. Process ShadowSpawn command output text file. Append the ShadowSpawn command output to the local backup log. If the ShadowSpawn command output contains no files to incrementally backup status, append that to the in-memory email report. If the ShadowSpawn command output contains a local backup-destination validation failure, append that to the in-memory email report.

33. [AWS S3 backup files upload and deletion] if applicable, upload the encrypted and compressed backup files to AWS, and delete them. if AWS upload error, append the error in the in-memory local backup log, flag no email sending, abort the AWS S3 backup files upload, and delete the AWS S3 backup files. (UPDATE [10/20/2020 4:53 PM CST] for easy and safe AWS S3 backup files management, create a directory in c:\temp with the name in the format of backup_set_unique_prefixYYYYMMDD_HHMMSS.SSSSS, create all AWS S3 backup files in that directory, and delete the entire directory after the use.)
34. [Temporary output files deletion] Delete the temporary output files.
35. [Email report sending] If applicable, send the regular-backup email report (that does not include the ShadowSpawn.exe outputs). If email sending failure, save the error in a data variable for including it in the local backup log. (think and decide on not including the

ShadowSpawn execution log in the email reports. no good will ever come out of emailing all the ShadowSpawn.exe, Robocopy and 7z.exe outputs in unencrypted emails. also, for the full backups, the Robocopy and maybe 7z.exe outputs will be too big to email—so emailing those outputs should be avoided. I'm very heavily leaning toward including the ShadowSpawn execution log in the AAM Auto Backup log only. think and decide on this, document the decision, and implement the decision!!!! [10/19/2020 8:30 AM CST])

36. [Local backup log file creation] Create the local backup log file using the temporary output files. (decide on ~~creating/~~saving the log after sending email. also using self._bEmailSendingFailure in _SendEmailReport(). to include email sending error in the log file. in all likely chance, I'll do this. [10/18/2020 9:24 PM CST])
37. [File lock release] Release the file lock.

(program exit)

1.1.5. DEPRECATED [10/20/2020 2:38 PM CST]

[10/16/2020 10:14 PM CST]

38. Try acquiring the file lock with timeout. If the file lock is already acquired by another AAM Auto Backup program instance, keep on waiting using pause with sleep(). If no file lock acquisition success within the maximum wait time, display the error in the terminal, email the backup failure report if the user specified the email report on, and exit the program.
39. ~~To~~ check for the Internet ~~and AWS~~ connection~~, perform a simple AWS operation~~. Internet~~/AWS~~ connection ~~or usage~~ error, mark a flag for skipping the backup to AWS S3, and mark a flag for including the internet~~/AWS~~ connection ~~or usage~~ error in the log file. (TO DO [10/17/2020 3:53 PM CST] incorporate proper no internet connection handling.) (UPDATE [10/18/2020 6:17 PM CST] list the backup AWS S3 bucket objects.) (UPDATE [10/18/2020 6:30 PM CST] querying AWS costs money, and it is unnecessary, so it should be avoided. search "Python how to check internet connection" online. if upload to AWS S3 fails, then that can be handled properly later on. no reason to query AWS here.) (TO DECIDE [10/18/2020 6:44 PM CST] I'm thinking about skipping this step entirely, because there doesn't seem to be a reason to do this.)
40. Read last AWS S3 backup dates and times plain text file content, and save it in data variable(s) to pass them in the ShadowShawn execution.
41. Process the AAM Auto Backup instruction plain text file, to build the list of the directories to shadow mount, and keep in memory the lists of folders and files to exclude per shadow-copy mounted directory.
42. If applicable, check the last AWS S3 backup dates and times plain text file existence, and acquire its content if it exists. If the file doesn't exist, and AWS backup is on, report the non-existence in the terminal, flag for reporting the non-existence in the log and the email report, and query the backup AWS S3 bucket as needed. (UPDATE [10/18/2020 6:34 PM CST] if AWS query error, handle that properly.)
43. For each Robocopy directory to backup, perform the following.
    43.1. using the lists of folders and files to exclude in memory, in the AAM Auto Backup program directory, create a temporary plain text file that includes the lists of the folders and files to exclude, for this shadow-copy mounted directory.
    43.2. to execute AAM Auto Backup in the ShadowSpawn operation mode that executes the following, execute the ShadowSpawn command, with the output redirected to a

temporary output file with a file name unique to this shadow-copy mounted directory (or programmatically read the output then save?).

43.2.1. if applicable, generate the Robocopy command in memory, using the temporary plain text file that includes the lists of folders and files to exclude for this shadow-copy mounted directory.

43.2.2. if applicable, scan the shadow copy mount to build the 7z.exe list file, using the temporary plain text file that includes the lists of folders and files to exclude for this shadow-copy mounted directory. (instead of using os.listdir() and os.path.isdir(), use the Python walk function to recursively scan each directory to backup in the shadow copy drive. [10/17/2020 12:14 PM CST]) (If there is no file to incrementally backup, because there is no change or new file, report the no incremental backup status in terminal, and flag for reporting the no file to incrementally backup later in the log and email report. [10/17/2020 4:09 PM CST])

43.2.3. auto-generate (overwrite or create) the regular-backup batch file with the cd command, Robocopy command (if applicable), and 7z.exe command (if applicable), in the AAM Auto Backup program directory. (dynamically create the batch file with the dynamically generated Robocopy command. [10/7/2020 3:16 PM CST])

43.2.4. execute the regular-backup batch file. (execute the dynamically generated batch file for doing the shadow backup. [10/7/2020 3:16 PM CST])

43.2.5. perform the local backup-destination validation check after each Robocopy command execution. in the shadow-copy mounted drive, recursively scan each directory included in the backup, and check that every non-excluded file exists in the backup destination. if error, use the print() function to output the error to the terminal.

44. if applicable, upload the encrypted and compressed backup files to AWS, and delete them. (UPDATE [10/18/2020 6:34 PM CST] if AWS upload error, handle that properly.)
45. Create the log file using the temporary output files.
46. Delete the temporary output files.
47. If applicable, send the regular-backup email report.
48. Create or update the last AWS S3 backup dates and times plain text file content.
49. Exit the program.

~~TO DO [10/17/2020 12:03 PM CST] update the following using the "operation mode overview" sub-section content in the "Regular-backup Operation Mode" section. (DONE. [10/17/2020 4:11 PM CST])~~
[10/17/2020 4:11 PM CST] the above adheres to the "operation mode overview" sub-section content in the "Regular-backup Operation Mode" section.

1.2. Incremental-only Backup Process Spec

1.2.1. [10/20/2020 4:25 PM CST]

The incremental-only backup process is similar to the regular backup process, with a few important differences.

The incremental-only backup process never performs the full backup; it performs incremental backups only, to a local data storage (if commanded by the user), and to the backup AWS S3 bucket (if commanded by the user).

The incremental-only backup uses the latest completely-uploaded full backup in the backup AWS S3 bucket.

At the end of each incremental-only backup operation, AAM Auto Backup does not email a backup report via AWS SES, and it waits for a keyboard user input before exiting the program to ensure that the user sees all the backup messages in the terminal.

The incremental-only backup process adheres to the "operation mode overview" sub-section content in the "Incremental-only Backup Operation Mode" section.

1.2.2. DEPRECATED [10/20/2020 4:24 PM CST]

[10/17/2020 4:21 PM CST]
The incremental-only backup process is similar to the regular backup process, with a few important differences.

The incremental-only backup process performs incremental backups only, to a local data storage (if commanded by the user), and to the backup AWS S3 bucket (if commanded by the user).

When incrementally backing up data to the backup AWS S3 bucket, the incremental-only backup uses the date and time of the latest completed-upload full data backup in the backup AWS S3 bucket, in the name of the incremental backup files that are uploaded to the backup AWS S3 bucket. The incremental-only backup process uses the last AWS S3 backup dates and times plain text file content, to decide the completely-uploaded latest full backup date and time in the backup AWS S3 bucket, and the files to incrementally backup to AWS S3. If the last AWS S3 backup dates and times plain text file is missing, the incremental-only backup process queries AWS S3 for the needed information.

At the end of each incremental-only backup operation, AAM Auto Backup does not email a backup report via AWS SES, and it waits for a keyboard user input before exiting the program to ensure that the user sees all the backup messages in the terminal.

The incremental-only backup process adheres to the "operation mode overview" sub-section content in the "Incremental-only Backup Operation Mode" section.

1.3. ShadowSpawn-executed Backup Process Spec

ShadowSpawn-~~executed AWS S3~~ Backup Process Spec
ShadowSpawn-~~execution~~ Backup Process Spec
ShadowSpawn-executed Backup Process Spec

[10/16/2020 10:14 PM CST]
Execute the ShadowSpawn-executed portion of each AAM Auto Backup backup operation, then exit the program.

2. Data-restoration Process Specs

2.1. One-step AWS S3 Data Restoration Process Spec

[10/16/2020 10:14 PM CST]
perform all the processes in the multi-step AWS S3 data restoration.

2.2. Multi-step AWS S3 Data Restoration Process Specs

[10/16/2020 10:14 PM CST] all of the following with a proper no Internet connection handling.

1. Restore to AWS S3 from AWS S3 Glacier Deep Archive
2. Check AWS S3 restoration status
3. Download to local computer from AWS S3
4. Decompress all the downloaded full and incremental backup files on the local computer

VI. Implementation Techniques

1. how to do Windows Shadow Backup or Copy

[10/2/2020 6:09 AM CST]
search "Windows shadow copy command" online.
https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/vssadmin

[10/3/2020 11:23 AM CST] the Windows 7 Home shadow copy CLI program cannot be used for shadow copying. use a Python module for Windows shadow copying.
Search "python module windows shadow copy" online.
https://www.sans.org/blog/using-volume-shadow-copies-from-python/
https://social.technet.microsoft.com/Forums/scriptcenter/en-US/7e28026a-63a5-4129-a956-e3332a1de236/volume-shadow-copy-management-in-python?forum=ITCG

Hi Scott,
If it is not too late, you can use https://sourceforge.net/projects/pyvss/ :

PyVSS is a python module which allows you to copy files using Volume Shadow Copy Service (http://en.wikipedia.org/wiki/Shadow_Copy). It is in an early development phase, yet is stable, simple and straight-forward to use.
At this moment is only able to copy files using VSS.
marius

https://en.wikipedia.org/wiki/Shadow_Copy

https://stackoverflow.com/questions/13624198/consistent-backups-in-python

1.1. Shadowspawn.exe execution issues handling (administrator mode requirement in Windows, etc)

[12/19/2020 1:20 PM CST]

When a Python CLI program that executes ShadowSpawn.exe via os.system() is executed in an administrator terminal, no separate terminal popup, and the proper output file generation.

When a Python CLI program that executes ShadowSpawn.exe via os.system() is executed by Windows Task Scheduler with the "Run with highest privileges" setting on, no separate terminal popup, and the proper output file generation.

When ShadowSpawn.exe via os.system() is executed by Visual Studio code that is launched with the administrator privilege, no separate terminal popup, and the proper output file generation.

VERDICT: os.system() can and will be used for my need.

1.1.1. debugging technique

[12/19/2020 1:29 PM CST]
Launch Visual Studio Code with the administrator privilege. see above for why.

1.1.2. ORIGINAL NOTES [12/19/2020 1:20 PM CST]

~~1.1.1.1.4. Look more into os.system(). (DONE. [12/19/2020 11:52 AM CST])~~
- ~~1.1.1.1.4.1. Read its manual entry. (DONE. [12/19/2020 11:09 AM CST])~~
- ~~1.1.1.1.4.2. Run it in IDLE. (DONE. [12/19/2020 11:25 AM CST])~~
- ~~1.1.1.1.4.3. Run it in a test Python CLI program with nothing but Shadowspawn.exe executed by os.system(), with the output~~

~~redirection to a file, with the administrator privilege. [12/19/2020 9:16 AM CST] (DONE. [12/19/2020 11:52 AM CST])~~

~~1.1.1.1.5. shadow_copy_command_output.txt contains no output, when AAM Auto Backup is executed from Visual Studio Code, and shadowspawn.exe is executed in a new terminal that gets popped up. try executing AAM Auto Backup from a terminal, and with no c:\users\allen directories in shadow_copy_instruction.txt, and see if the same thing happens. [12/18/2020 8:54 PM CST] (NO NEED [12/19/2020 11:54 AM CST] in administrator terminal, ShadowSpawn.exe executes without popping a new terminal, and the output file gets generated properly.)~~

~~1.1.1.1.6. launch Visual Studio Code with administrator privilege. see if that eliminates the ShadowSpawn.exe terminal popup and produces the output file properly. [12/19/2020 11:55 AM CST] (DONE [12/19/2020 12:21 PM CST] ShadowSpawn.exe executes without popping a new terminal, and the output file gets generated properly.)~~

~~1.1.1.1.7. run (AAM Auto Backup)/(recursive_program_execution_test3.py) via Windows Task Scheduler as administrator, see if the terminal window pops up with outputs. assess the situation, and decide what to do; if the terminal window pops up with outputs, os.system() is not usable, since displaying the outputs makes the command execution too slow, and I must use the one that does not display any terminal window. [12/18/2020 8:05 PM CST] (DONE. [12/19/2020 1:19 PM CST])~~

1.1.1.1.8. create and write out a section titled "Shadowspawn.exe execution issues handling (administrator mode requirement in Windows, etc)" in the AAM Auto Backup document. [12/19/2020 9:12 AM CST]

~~1.1.1.1.9. If needed, update the AAM Auto Backup source code to execute Shadowspawn.exe as a subprocess with administrator privileges. [12/19/2020 9:19 AM CST] (NO NEED. [12/19/2020 1:20 PM CST])~~

1.2. references

[10/4/2020 3:39 PM CST]
search "windows shadow copy reference" online.
https://docs.microsoft.com/en-us/windows-server/storage/file-server/volume-shadow-copy-service
https://docs.microsoft.com/en-us/windows/win32/vss/volume-shadow-copy-reference

https://docs.microsoft.com/en-us/windows/win32/vss/volume-shadow-copy-service-overview?redirectedfrom=MSDN

https://en.m.wikipedia.org/wiki/Backup_and_Restore
(Look into using Windows Backup and Restore. It might not be necessary to use Windows shadow copy directly.)
https://en.m.wikipedia.org/wiki/WBAdmin

(Maybe using WBAdmin will do. No complication whatsoever needed in directly using Windows shadow copy.)
https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/wbadmin
Test creating full and incremental backup zip files using WBAdmin. For the mirror back up to the USB hard drive, maybe a single incremental backup zip file for all the folders (and files) to be backed up can be created using WBAdmin, and then decompressed with automatic overwrite onto the mirror backup folder in the USB hard drive. It could be as simple as that.
https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc742083(v=ws.10)

Search "wbadmin how to full backup a folder to a zip file" online.

Search "python run a command as administrator" online

https://social.technet.microsoft.com/Forums/en-US/3dee8ffa-461f-4ab0-9521-8a8e976df678/how-can-i-force-full-backups?forum=windowsbackup

https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/wbadmin-start-backup

try the following commands in an administrator cmd terminal

```
wbadmin start backup -backupTarget:C:\Temp\backup\ -include:C:\Temp\cleaned\ -vssFull
```

```
wbadmin start backup -backupTarget:C:\Temp\backup\ -include:C:\Temp\cleaned\ -vsscopy
```

```
C:\Windows\system32>wbadmin start backup -backupTarget:C:\Temp\backup\ -include:
C:\Temp\cleaned\ -vssFull
wbadmin 1.0 - Backup command-line tool
(C) Copyright 2004 Microsoft Corp.

Retrieving volume information...
A partial backup of volumes is not supported on this version of Windows.
```

[10/4/2020 5:40 PM CST] So, I do need to access Windows shadow copy from Python?

```
C:\Windows\system32>wbadmin start backup -backupTarget:C:\Temp\backup\ -include:
C:\Temp\cleaned\ -vsscopy
wbadmin 1.0 - Backup command-line tool
(C) Copyright 2004 Microsoft Corp.

Retrieving volume information...
```

A partial backup of volumes is not supported on this version of Windows.

[10/4/2020 5:43 PM CST] search online "does windows xcopy support windows shadow copy?"
https://en.wikipedia.org/wiki/XCOPY
No open files
XCOPY will not copy open files. Any process may open files for exclusive read access by withholding the FILE_SHARE_READ https://msdn.microsoft.com/en-us/library/aa363858.aspx The Windows Volume Shadow Copy service is used for such situations, but XCOPY does not use it. Therefore, XCOPY is not useful for backing up live operating system volumes.

[10/4/2020 5:43 PM CST] it looks like I will have to use Windows shadow copy from Python. No way out.

[10/4/2020 6:00 PM CST]
Consider using Xcopy or 7zip only, skipping backing up open files.

Search "does 7-zip support Windows shadow copy?" online
https://superuser.com/questions/735559/zip-files-and-volume-shadow-copy
https://sourceforge.net/p/sevenzip/discussion/45797/thread/baaff282/

https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc772172(v=ws.10)?redirectedfrom=MSDN

[10/4/2020 6:03 PM CST]
search "Windows 7 file copy command with Windows shadow copy" online.
https://sourcedaddy.com/windows-7/how-to-manage-shadow-copies.html
https://sourcedaddy.com/windows-7/shadow-copy-feature.html
https://docs.microsoft.com/en-us/windows-server/storage/file-server/volume-shadow-copy-service
https://docs.microsoft.com/en-us/windows/win32/vss/vshadow-tool-examples
https://superuser.com/questions/394296/is-there-a-tool-to-copy-a-folder-using-shadow-copy
Try Hobocopy from Wangdera Tools and Utilities.
https://candera.github.io/hobocopy/
https://github.com/candera/shadowspawn

https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/robocopy

https://en.wikipedia.org/wiki/Robocopy
Robocopy does not copy open files. Any process may open files for exclusive read access by withholding the FILE_SHARE_READ[6] flag during opening. Even Robocopy's Backup mode will not touch those files. (Backup mode instead runs Robocopy as a "Backup Operator". This allows Robocopy to override permissions settings, specifically, NTFS ACLs).[7][8] Normally

Volume Shadow Copy Service is used for such situations, but Robocopy does not use it. Consequently, Robocopy is not suitable for backing up live operating system volumes. However, a separate utility such as ShadowSpawn[9] (Free, Open Sourced, and MIT Licensed) or GSCopyPro[10] ($30 - $50) or DiskShadow.exe[11] (included with Windows Server 2008), can be used beforehand to create a shadow copy of a given volume, which Robocopy can then back up.

For the sake of completeness as I stumbled upon this. There is also a WBAdmin tool that can backup folders on Windows 7 using VSS. See TechNet for details. It works out of the box and no 3rd party tool needs to be installed.
Unfortunately, you can only backup individual files & folders on the Server Windows, the desktop versions give an error: "A partial backup of volumes is not supported on this version of Windows." Which is bizarre because the GUI can do it, but I guess that's Microsoft. – SilverbackNet Jun 16 '15 at 8:21

1.3. potentially easier solution (no Windows shadow copy)

[10/4/2020 6:06 PM CST]
use xcopy.

if xcopy displays the files that could not be copied due to being open, wait for 10 minutes and try again. keep waiting for up to 3 hours. if the waiting expires, just process the copied files only.

1.4. very likely final solution: using hobocopy or shadowspawn

[10/4/2020 6:18 PM CST]
https://superuser.com/questions/394296/is-there-a-tool-to-copy-a-folder-using-shadow-copy
Try Hobocopy from Wangdera Tools and Utilities.
https://candera.github.io/hobocopy/
https://github.com/candera/hobocopy/downloads
https://github.com/candera/shadowspawn
Probably the most common way to use ShadowSpawn is to use Robocopy make a copy of files that are currently in use.
ShadowSpawn is derived from the same source code as HoboCopy and is intended to replace it. The evolution was driven by the fact that although the shadow copy part of HoboCopy works well enough, the copying part was nowhere near as robust as tools like RoboCopy. By providing a tool that just takes care of the shadow copy, ShadowSpawn allows users to work with locked and in-use files using any other tool, not just the limited copy features provided by HoboCopy.

use hobocopy or shadowspawn (very likely with robocopy). keep the Windows shadow copy CLI program in the AAM Auto Backup program (source-code file) directory.

writing code to access an OS API (Windows shadow copy API, in this case) is too much of unnecessary work to do myself: I should avoid it if I can.

1.4.1. how to use ShadowSpawn

Installing ShadowSpawn
Most users can simply unzip the appropriate zip file from the download page. ShadowSpawn.exe can then just be run - there is no installer. However, ShadowSpawn uses the Visual C++ runtime, which may not be present on some machines. If ShadowSpawn does not work for you, run the vcredist executable available from the same download page.

Running ShadowSpawn
ShadowSpawn is a command-line tool: there is no GUI.

ShadowSpawn take three arguments:

The directory that contains the files you want to snapshot.
An available drive letter where the snapshot will become visible.
A command to run.

Let's say that you wanted to use robocopy to copy files from the C:\foo directory to the C:\bar directory. You could do that with the following command:

```
shadowspawn C:\foo Q: robocopy Q:\ C:\bar /s
```

That would cause shadowspawn to
Make a shadow copy of the C: drive.
Mount the shadowed version of the C:\foo directory at Q:.
Launch robocopy Q:\ C:\bar /s
Wait for Robocopy to finish.
Clean up the shadow copy and remove it from Q:

You can use any drive letter you want (it doesn't have to be Q:), but it does have to be a drive letter that's not currently being used for anything else.

You can run any command you want. So if you just wanted to use notepad to look at a shadow copy of C:\foo\blah.txt, you'd run

```
shadowspawn C:\foo Q: notepad Q:\blah.txt
```

Just remember that shadowspawn will remove the Q: drive as soon as the command you specify exits.

1.4.1.1. ShadowSpawn test command

[10/5/2020 10:46 AM CST]

shadowspawn C:\Temp\cleaned Q: robocopy Q:\ C:\Temp\backup

([10/5/2020 11:08 AM CST] ok! the above command works flawlessly! I can use ShadowSpawn for the fully-automated daily backup! no Windows Shadow Copy API use required!!!!!!)

1.4.1.2. likely use of ShadowSpawn or Windows Shadow Copy

[10/5/2020 10:56 AM CST]
an entire directory must be mounted for shadow copying.

so, instead of running ShadowSpawn multiple times for individual directories, I'll probably run ShadowSpawn on C:\Users\Allen\Documents, to run the ShadownSpawn CLI program just once for the backup.

1.5. how to execute or run a command as administrator in Python (for running ShadowSpawn)

CONCLUSION [10/5/2020 1:30 PM CST]
executing the following command in AAM_auto_backup.py works, when AAM_auto_backup.py is executed by Windows Task Scheduler as a Windows Task Scheduler task with the highest privilege.

os.system('"C:\\Users\\Allen\\Documents\\AAM Auto Backup\\ShadowSpawn.exe" C:\\Temp\\cleaned Q: robocopy Q:\\ C:\\Temp\\backup > c:\temp\test_log.txt')

instead of c:\temp\test_log.txt, use a folder and a log file name that is relevant to the current backup date and time.

use the above method, since Python code for granting Windows admin access is not required at all. **less coding is the best coding always!**

1.6. Robocopy references

[10/4/2020 10:59 PM CST]
search "Robocopy reference" online.
https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/robocopy

search "what is Robocopy job file" online.
The Ultimate Guide to Robocopy - Adam the Automator

adamtheautomator.com › robocopy-the-ultimate
Jump to Robocopy Jobs - Robocopy job files are text files containing one option per line. You'll typically use robocopy to create these job files.
https://adamtheautomator.com/robocopy-the-ultimate/
https://adamtheautomator.com/robocopy-the-ultimate/#Robocopy_Jobs

[10/5/2020 1:38 PM CST] how to skip copying certain files in each directory (e.g. MS Word temp files).

1.6.1. Robocopy notes

[10/5/2020 6:31 PM CST]

The Robocopy command to use
robocopy c:\reports '\\marketing\videos' yearly-report.mov /mt /z

/e   Copies subdirectories. This option automatically includes empty directories.

/log option
(A potential implementation technique. Run Robocopy to copy only new files. Process the Robocopy log file to get the list of the files copied. For the AWS S3 Deep Glacier incremental backup, copy the copied files to a separate directory, package the contents of the directory into a compressed file. Better yet, if the 7-Zip CLI program can use a text file with a list of files to compress, use that.)

/maxlad:<n>   Specifies the maximum last access date (excludes files unused since n).
n specifies a date in the format YYYYMMDD.
(The above Robocopy option can be used for incremental backup, since using the above option will copy only the newer files. Test on the Allen Documents directory.)

/l   Specifies that files are to be listed only (and not copied, deleted, or time stamped).
/ts   Includes source file time stamps in the output.
/fp   Includes the full path names of the files in the output.
/np   Specifies that the progress of the copying operation (the number of files or directories copied so far) will not be displayed.
/eta   Shows the estimated time of arrival (ETA) of the copied files.
/log:<logfile>   Writes the status output to the log file (overwrites the existing log file).
/log+:<logfile>   Writes the status output to the log file (appends the output to the existing log file).
/unicode   Displays the status output as Unicode text.
/unilog:<logfile>   Writes the status output to the log file as Unicode text (overwrites the existing log file).
/unilog+:<logfile>   Writes the status output to the log file as Unicode text (appends the output to the existing log file).

/tee   Writes the status output to the console window, as well as to the log file.
Job options
JOB OPTIONS
Option   Description
/job:<jobname>   Specifies that parameters are to be derived from the named job file.
/save:<jobname>   Specifies that parameters are to be saved to the named job file.
/quit   Quits after processing command line (to view parameters).
/nosd   Indicates that no source directory is specified.
/nodd   Indicates that no destination directory is specified.
/if   Includes the specified files.
(Use the above for testing.)

To create the Robocopy command(s) to use, create, document, and execute a variety of Robocopy commands.

1.6.2. test Robocopy commands to try to find the Robocopy command(s) to use in AAM Auto Backup

[10/5/2020 6:32 PM CST]

[create and] use a complex directory structure with sub-directories. use "C:\Temp\backup_source" as the source with the "C:\Temp\AAM_Audio_Masterer testing" contents and newly created plain text files; use "C:\Temp\backup_destination" as the destination.

(the first, very basic Robocopy commands to try)
robocopy C:\Temp\backup_source C:\Temp\backup_destination
robocopy C:\Temp\backup_source C:\Temp\backup_destination /mt /z

(folder copy)
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e
~~robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /b~~

(individual folder copy)
~~robocopy C:\Temp\backup_source C:\Temp\backup_destination \test*.~~
~~robocopy C:\Temp\backup_source C:\Temp\backup_destination test*.~~
~~robocopy C:\Temp\backup_source C:\Temp\backup_destination .\test*.~~
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd C:\Temp\backup_source\test\

(log generation)
~~robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /unilog:'C:\Temp\backup_destination\copy_log.txt'~~

robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /log:"C:\Temp\backup_destination\copy_log.txt"

robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /unilog:"C:\Temp\backup_destination\copy_log.txt"

(copy files after a certain data and time only)
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /log:"C:\Temp\backup_destination\copy_log.txt" /maxage:20201006

/maxage:<n>   Specifies the maximum file age (to exclude files older than n days or date).

robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /log:"C:\Temp\backup_destination\copy_log.txt" /maxlad:20201006

/maxlad:<n>   Specifies the maximum last access date (excludes files unused since n).

(job file creation)

(job file use)

(ok, I think the above will do for now!)

1.6.2.1. /b parameter output

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /b

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 06 09:49:50 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

  Options : *.* /S /E /COPY:DAT /B /R:1000000 /W:30
```

```
-------------------------------------------------------------------------------

ERROR : You do not have the Backup and Restore Files user rights.
*****  You need these to perform Backup copies (/B or /ZB).

ERROR : Robocopy ran out of memory, exiting.
ERROR : Invalid Parameter #%d : "%s"

ERROR : Invalid Job File, Line #%d :"%s"


  Started : %hs

   Source %c

     Dest %c
       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

1.6.2.2. /unilog:'C:\Temp\backup_destination\copy_log.txt' parameter output

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /uni
log:'C:\Temp\backup_destination\copy_log.txt'

2020/10/06 09:58:47 ERROR 123 (0x0000007B) Opening Log File C:\Users\Allen\'C:\T
emp\backup_destination\copy_log.txt'
The filename, directory name, or volume label syntax is incorrect.

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 06 09:58:47 2020
```

```
   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #4 : "/unilog:'C:\Temp\backup_destination\copy_log.txt
'"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

1.6.2.3. \test*. parameter output

[10/6/2020 1:19 PM CST]

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination \test\*
.

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 06 13:18:53 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /COPY:DAT /R:1000000 /W:30
```

```
-------------------------------------------------------------------------------

ERROR : Invalid Parameter #3 : "\test\*."

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

1.6.2.4. test*. parameter output

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination test\*.


-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 06 13:20:27 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /COPY:DAT /R:1000000 /W:30

-------------------------------------------------------------------------------

ERROR : Invalid Parameter #3 : "test\*."

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
```

```
              /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

1.6.2.5. .\test*. parameter output

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination .\test\
*.

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 06 13:21:38 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #3 : ".\test\*."

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

1.6.3. meanings of Windows created modified accessed file date and time

[10/6/2020 10:19 AM CST]
search "Windows created modified accessed file date and time meanings" online.

https://support.accessdata.com/hc/en-us/articles/204353955-File-Created-File-Accessed-File-Modified

File Created: This is the date the file was "created" on the volume. This does not change when working normally with a file, e.g. opening, closing, saving, or modifying the file.

File Accessed: This is the date the file was last accessed. An access can be a move, an open, or any other simple access. It can also be tripped by Anti-virus scanners, or Windows system processes. Therefore caution has to be used when stating a "file was last accessed by user XXX" if there is only the "File Access" date in NTFS to work from.

File Modified: This date as shown by Windows there has been a change to the file itself. E.g a notepad document is has more date added to it, would trip the date it was modified.

1.7. Robocopy job file content to use for the daily automated backup

The Robocopy command(s?) to run to create the job file.
/save:<jobname>

1.8. Robocopy commands to use for the fully-automated daily backup

[10/6/2020 10:44 AM CST]

A MUST [10/6/2020 1:51 PM CST] manually test all the commands below, before actually including and using them in the AAM Auto Backup source code.

A DECISION [10/6/2020 10:46 AM CST] leave the old files that are deleted in the backup source (for the archiving purposes).

~~[10/6/2020 10:56 AM CST] use a Robocopy job file for specifying the folders and files to copy.~~
FINAL DECISION [10/6/2020 1:22 PM CST] no need to use Robocopy job file(s). the Robocopy command text will be generated in the AAM Auto Backup Python code.

FINAL DECISION [10/6/2020 1:23 PM CST] **since Robocopy copies the entire directory contents, and can only exclude (not select) files and sub-directories, AAM Auto Backup will create proper Robocopy command(s) for that.**

1.8.1. full backup Robocopy commands

1.8.1.1. full backup to the USB hard drive

```
robocopy "C:\Users\Allen\Documents\" "(backup destination root
folder full path, ending with \)" /e /unilog:"(backup destination
root folder full path)\backup_log_(auto-generated backup date and
time).txt" /xd <directory>[ ...] /xf <filename>[ ...]
```

NOTES
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /unilog:"C:\Temp\backup_destination\copy_log.txt" /xd <directory>[...] /xf <filename>[...]

robocopy C:\Temp\backup_source C:\Temp\backup_destination /e
the above copies only the changed file. maybe the above can be used for incremental backup to the USB hard drive too.

(TO DECIDE [10/6/2020 3:51 PM CST] is there a need for a log file? I'm starting to think no. no need to create and access the log file.)
(FINAL DECISION [10/6/2020 3:52 PM CST] no log file needed!)
(UPDATED FINAL DECISION [10/6/2020 3:54 PM CST] produce the log file for debugging purposes!)

1.8.1.2. full backup to AWS S3 Glacier Deep Archive

[10/6/2020 4:03 PM CST]

No Robocopy use in full backup to AWS S3 Glacier Deep Archive.

ShadowSpawn executes a 7-Zip CLI command.

The 7-Zip CLI command compresses the Windows shadow copy of the entire backup source directory, with the excluded sub-directories and files.

The above approach means that for each backup source directory, a set of Cloud backup files are generated, both full and incremental.

Better yet, a single ShadowSpawn CLI command should execute a batch file that executes both Robocopy and 7-zip in sequence, back-to-back!

(ok, ok, now I am finally getting this completely right! finally!!!!!! [10/6/2020 4:10 PM CST])

NOTES
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /unilog:"C:\Temp\backup_destination\copy_log.txt" /xd <directory>[...] /xf <filename>[...]

1.8.2. incremental backup Robocopy commands

1.8.2.1. update backup (with overwrite) to the USB hard drive

```
robocopy "C:\Users\Allen\Documents\" "(backup destination root
folder full path, ending with \)" /e /unilog:"(backup destination
root folder full path)\backup_log_(auto-generated backup date and
time).txt" /xd <directory>[ ...] /xf <filename>[ ...]
```

NOTES
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /log:"C:\Temp\backup_destination\copy_log.txt" ~~/maxlad:20201006~~ /xd <directory>[...] /xf <filename>[...]

1.8.2.2. incremental backup to AWS S3 Glacier Deep Archive

[10/6/2020 4:03 PM CST]

If 7-zip supports excluding files by access date, do the following: don't use Robocopy for the incremental backup to AWS S3 Glacier Deep Archive—use 7-zip CLI command only.

No Robocopy use in incremental backup to AWS S3 Glacier Deep Archive.

ShadowSpawn executes a 7-Zip CLI command.

The 7-Zip CLI command compresses the Windows shadow copy of the relevant files and directories in the backup source directory, with the excluded sub-directories and files.

The above approach means that for each backup source directory, a set of Cloud backup files are generated, both full and incremental.

Better yet, a single ShadowSpawn CLI command should execute a batch file that executes both Robocopy and 7-zip in sequence, back-to-back!

(ok, ok, now I am finally getting this completely right! finally!!!!!! [10/6/2020 4:10 PM CST])

NOTE [10/6/2020 4:16 PM CST] one technique that can be used is AAM Auto Backup scanning the entire contents of the shadow copy of each backup source root directory, building a list of files (in UTF-8) to archive using the file access time and the last backup date and time info, then

specifying the list in the 7-Zip CLI command, since the 7-Zip CLI command can take the [@listfile] argument.
FINAL DECISION [10/6/2020 4:36 PM CST] I'll use the above method. the above seems to be the best method.

<u>NOTES</u>
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /log:"C:\Temp\backup_destination\copy_log.txt" /maxlad:20201006 /xd <directory>[...] /xf <filename>[...]

1.8.3. [failure] test for omitting all the non-backup folders in the backup directory in the Robocopy command by calling AAM Auto Backup again in the ShadowCopy command executed by AAM Auto Backup 1

FINAL DECISION [10/7/2020 1:33 PM CST]
I'll create a separate Python program, named recursive_program_execution_test.py, for testing this.
[10/7/2020 2:29 PM CST] place recursive_program_execution_test.py in "C:\Users\Allen\Documents\AAM Auto Backup". in an administrator terminal, cd to "C:\Users\Allen\Documents\AAM Auto Backup", then execute recursive_program_execution_test.py without any argument, to perform the test.

rethink the Robocopy use, and document the rethinking. test the new approach. [10/7/2020 1:26 PM CST]

rethink the Robocopy use, and document the rethinking. In "C:\Users\Allen\Documents", new folders will be created albeit occasionally. address that issue. while I think calling ShadowSpawn multiple times per directory to copy is a very bad idea and practice, dynamically assigning the directories to omit in the Robocopy command doesn't look very clean either. I certainly do not want to copy all the files to back up to a separate temporary and intermediate directory, back up the intermediate directories and files, then deleting them. the solution I think I should implement is ShadowSpawn executing AAM Auto Backup with a certain command format, or executing another Python program, that will check the contents of the shadow copy mount, dynamically generate the (correct) Robocopy command using the contents of the shadow copy mount, then execute Robocopy. I think that is the best way to go. if using AAM Auto Backup for the job, then I must firs test whether AAM Auto Backup can be executed by a ShadowSpawn command that is executed by AAM Auto Backup. I'll test that, and document everything relevant, including this note. [10/7/2020 1:26 PM CST]

1.8.3.1. test result (terminal output)

```
C:\Users\Allen\Documents\
\AAM Auto Backup>recursive_program_execution_test.py
---main execution---

ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing c:\temp\backup_source\ at Q:
Launching command: recursive_program_execution_test.py "ShadowSpawn"
Aborting backup.
Dismounting device: Q:
Deleting snapshot.
There was an error calling CreateProcess. Process: recursive_program_execution_test.py
"ShadowSpawn"  Error: Error 193
```

1.8.4. [failure] test for omitting all the non-backup folders in the backup directory in the Robocopy command by calling AAM Auto Backup again in the ShadowCopy command executed by AAM Auto Backup 2

[10/7/2020 2:58 PM CST]
using recursive_program_execution_test2.py, that calls recursive_program_execution_test.py through ShadowSpawn.

1.8.4.1. test result (terminal output)

```
ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing c:\temp\backup_source\ at Q:
Launching command: recursive_program_execution_test2.py "ShadowSpawn"
Aborting backup.
Dismounting device: Q:
Deleting snapshot.
There was an error calling CreateProcess. Process: recursive_program_execution_test2.py
"ShadowSpawn"  Error: Error 193
```

1.8.5. [success] test for omitting all the non-backup folders in the backup directory in the Robocopy command by calling AAM Auto Backup again in the ShadowCopy command executed by AAM Auto Backup 3

[10/7/2020 3:02 PM CST]
test if calling Python.exe can work in recursive_program_execution_test3.py. too early to give up yet.

1.8.5.1. test result (terminal output)

```
---main execution---

ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing c:\temp\backup_source\ at Q:
Launching command: "C:\Users\Allen\AppData\Local\Programs\Python\Python38\python.exe"
recursive_program_execution_test3.py "ShadowSpawn"

C:\Users\Allen\Documents\AAM Auto Backup>cd /D q:

Q:\>dir
 Volume in drive Q is OS
 Volume Serial Number is 4407-9849

 Directory of Q:\

10/06/2020  01:18 PM    <DIR>          .
10/06/2020  01:18 PM    <DIR>          ..
09/30/2020  06:17 PM             1,382 20200925 1240pm voiceover recording.aup
10/06/2020  09:20 AM    <DIR>          20200925 1240pm voiceover recording_data
10/06/2020  01:18 PM    <DIR>          test
10/06/2020  09:20 AM    <DIR>          voiceover recording 1
10/06/2020  09:20 AM    <DIR>          voiceover recording 2
10/06/2020  09:20 AM    <DIR>          voiceover recording 3
               1 File(s)          1,382 bytes
               7 Dir(s)  71,441,281,024 bytes free
---execution by ShadowSpawn---
['20200925 1240pm voiceover recording.aup', '20200925 1240pm voiceover recording_data',
'test', 'voiceover recording 1', 'voiceover recording 2', 'voiceover recording 3']
the batch file to execute should be auto-generated by AAM Auto Backup here.
Launched command finished with exit code: 0.
Shadowing successfully completed.
```

1.9. ShadowSpawn test commands

[10/6/2020 8:34 PM CST]
If all of the robocopy, 7z, shadowspawn, and AWS CLI commands work properly, the Python coding will be very minimal (as it should be): because I use existing software programs, the job is much easier and faster for me! The only thing I am unsure of is changing the directory in the batch file executed by shadowspawn. I will create and execute a test batch file that changes the path, and saves dir output with the current path, in a plain text file using > in the command.

1.9.1. [success] basic ShadowSpawn test command

1.9.1.1. command to execute for testing

[10/7/2020 11:33 AM CST]

Since ShadowSpawn requires the administrator privilege, launch the administrator CMD terminal to run the following ShadowSpawn test command.

`basic_shadowspawn_test.bat` is in the AAM Auto Backup program directory, so change the directory to the AAM Auto Backup program directory before executing the following ShadowSpawn test command.

```
shadowspawn c:\temp\backup_source\ Q: basic_shadowspawn_test.bat
> basic_shadowspawn_test_log.txt
```

the following in `basic_shadowspawn_test.bat`.

```
cd /D q:
dir
```

1.9.1.2. test result

[10/7/2020 12:01 PM CST]

```
ShadowSpawn (c) 2011 Craig Andera. shadowspawn@wangdera.com

Shadowing c:\temp\backup_source\ at Q:
Launching command: basic_shadowspawn_test.bat

C:\Users\Allen\Documents\AAM Auto Backup>cd /D q:

Q:\>dir
 Volume in drive Q is OS
 Volume Serial Number is 4407-9849

 Directory of Q:\

10/06/2020  01:18 PM    <DIR>          .
10/06/2020  01:18 PM    <DIR>          ..
09/30/2020  06:17 PM             1,382 20200925 1240pm voiceover recording.aup
10/06/2020  09:20 AM    <DIR>          20200925 1240pm voiceover recording_data
10/06/2020  01:18 PM    <DIR>          test
10/06/2020  09:20 AM    <DIR>          voiceover recording 1
10/06/2020  09:20 AM    <DIR>          voiceover recording 2
10/06/2020  09:20 AM    <DIR>          voiceover recording 3
```

```
               1 File(s)          1,382 bytes
               7 Dir(s)  67,446,808,576 bytes free
Launched command finished with exit code: 0.
Shadowing successfully completed.
```

(OK!!! it works!!!!!! IT WORKS!!!!!!!!!!!!!!!!!!! now, I can complete AAM Auto Backup and make it happen for sure, very likely today!!!!!!! [10/7/2020 12:05 PM CST])

1.9.2. [failure] full ShadowSpawn test command 1

1.9.2.1. command to execute for testing

[10/7/2020 12:10 PM CST]

Since ShadowSpawn requires the administrator privilege, launch the administrator CMD terminal to run the following ShadowSpawn test command.

`full_shadowspawn_test.bat` is in the AAM Auto Backup program directory, so change the directory to the AAM Auto Backup program directory before executing the following ShadowSpawn test command.

```
shadowspawn c:\temp\backup_source\ Q: full_shadowspawn_test.bat >
full_shadowspawn_test_log.txt
```

the following in `full_shadowspawn_test.bat`.

```
cd /D q:
robocopy "C:\Temp\backup_source\" "C:\Temp\
backup_destination\" /e /unilog:"C:\Temp\backup_destination\
copy_log.txt"
"c:\program files\7-zip\7z" a "C:\Temp\backup_destination\
full_shadowspawn_test.zip" -v4000000000b -mcu=on -mem=AES256 -
1234abcd @"C:\Users\Allen\Documents\AAM Auto Backup\
full_shadowspawn_test_listfile.txt"
```

1.9.2.2. test result

[10/7/2020 1:04 PM CST]

full_shadowspawn_test.zip.001 is created.

copy_log.txt is created. no file copied in C:\Temp\backup_destination.

the source directory in the Robocopy command is wrong.

```
-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows
-------------------------------------------------------------------------------

  Started : Wed Oct 07 13:02:51 2020

   Source : C:\Temp\backup_source" C:\Temp\backup_destination"\
     Dest -

    Files : *.*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : No Destination Directory Specified.

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !
```

1.9.3. [success] full ShadowSpawn test command 2

1.9.3.1. command to execute for testing

[10/7/2020 12:10 PM CST]

Since ShadowSpawn requires the administrator privilege, launch the administrator CMD terminal to run the following ShadowSpawn test command.

`full_shadowspawn_test.bat` is in the AAM Auto Backup program directory, so change the directory to the AAM Auto Backup program directory before executing the following ShadowSpawn test command.

```
shadowspawn c:\temp\backup_source\ Q: full_shadowspawn_test.bat >
full_shadowspawn_test_log.txt
```

the following in `full_shadowspawn_test.bat`.

```
cd /D q:
robocopy "q:" "C:\Temp\backup_destination" /e /unilog:"C:\Temp\
backup_destination\copy_log.txt"
"c:\program files\7-zip\7z" a "C:\Temp\backup_destination\
full_shadowspawn_test.zip" -v4000000000b -mcu=on -mem=AES256 -
1234abcd @"C:\Users\Allen\Documents\AAM Auto Backup\
full_shadowspawn_test_listfile.txt"
```

1.9.3.2. test result

[10/7/2020 1:13 PM CST] all the files have been copied and created. so, **no \ at the end of source and target Robocopy directories!**

```
-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows
-------------------------------------------------------------------------------

  Started : Wed Oct 07 13:12:11 2020

   Source : Q:\
     Dest : C:\Temp\backup_destination\

    Files : *.*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

	                   1	Q:\
	    *EXTRA File 		     162	~$test.doc
	    *EXTRA File 		       0	copy_log.txt
	      New File  		    1382	20200925 1240pm voiceover recording.aup
  0%
100%
	      New Dir          0	Q:\20200925 1240pm voiceover recording_data\
	      New Dir          0	Q:\20200925 1240pm voiceover recording_data\e08\
	      New Dir          2	Q:\20200925 1240pm voiceover recording_data\e08\d08\
	      New File  		  301124	e08087d3.au
  0%
 43%
 87%
100%
	  New File  		   1.0 m	e0808912.au
  0%
 24%
 49%
 74%
 98%
100%
	New Dir 	1	Q:\test\
	  New File  		      13	new file.txt
  0%
100%
	New Dir 	1	Q:\voiceover recording 1\
	  New File  		    1400	20200925 1240pm voiceover recording (copy
1).aup
  0%
100%
	New Dir 	0	Q:\voiceover recording 1\20200925 1240pm voiceover recording
(copy 1)_data\
	New Dir 	0	Q:\voiceover recording 1\20200925 1240pm voiceover recording
(copy 1)_data\e08\
	New Dir 	2	Q:\voiceover recording 1\20200925 1240pm voiceover recording
(copy 1)_data\e08\d08\
	  New File  		   1.0 m	e08088aa.au
  0%
 24%
 49%
 74%
 98%
100%
	  New File  		  301124	e0808c0f.au
  0%
 43%
 87%
100%
	New Dir 	1	Q:\voiceover recording 2\
	  New File  		    1400	20200925 1240pm voiceover recording (copy
2).aup
  0%
100%
	New Dir 	0	Q:\voiceover recording 2\20200925 1240pm voiceover recording
(copy 2)_data\
	New Dir 	0	Q:\voiceover recording 2\20200925 1240pm voiceover recording
(copy 2)_data\e08\
          New Dir          2        Q:\voiceover recording 2\20200925 1240pm voiceover recording
(copy 2)_data\e08\d08\
            New File                301124         e0808a7a.au
  0%
 43%
 87%
100%
            New File                1.0 m          e0808dd7.au
  0%
 24%
 49%
 74%
 98%
100%
          New Dir          1        Q:\voiceover recording 3\
            New File                  1400         20200925 1240pm voiceover recording (copy
3).aup
  0%
100%
          New Dir          0        Q:\voiceover recording 3\20200925 1240pm voiceover recording
(copy 3)_data\
          New Dir          0        Q:\voiceover recording 3\20200925 1240pm voiceover recording
(copy 3)_data\e08\
          New Dir          2        Q:\voiceover recording 3\20200925 1240pm voiceover recording
(copy 3)_data\e08\d08\
            New File                1.0 m          e08082a3.au
  0%
 24%
 49%
 74%
 98%
100%
            New File                301124         e080899e.au
  0%
 43%
 87%
100%

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :        17        16         1         0         0         0
   Files :        13        13         0         0         0         2
   Bytes :    5.20 m    5.20 m         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00
```

```
Speed :          20658768 Bytes/sec.
Speed :          1182.104 MegaBytes/min.

Ended : Wed Oct 07 13:12:11 2020
```

1.10. ShadowSpawn commands to use for the fully-automated daily backup

1.10.1. [10/7/2020 3:52 PM CST]

<u>Full Backup</u>

```
shadowspawn (the backup root directory full path) (the shadow
copy mount drive name or letter): "(a Python executable full
path)" AAM_auto_backup.py (full-backup parameters, including the
shadow copy mount drive name or letter)
```

<u>Incremental Backup</u>

```
shadowspawn (the backup root directory full path) (the shadow
copy mount drive name or letter): "(a Python executable full
path)" AAM_auto_backup.py (incremental-backup parameters,
including the shadow copy mount drive name or letter)
```

1.10.2. DEPRECATED [10/7/2020 3:52 PM CST]

[10/6/2020 5:24 PM CST]
<u>Full Backup</u>

```
shadowspawn (a backup root directory full path) Q:
AAM_Auto_Backup_full_backup.bat
```

<u>Incremental Backup</u>

```
shadowspawn (a backup root directory full path) Q:
AAM_Auto_Backup_incremental_backup.bat
```

([10/5/2020 11:22 AM CST] damn right!!!! finally, I'm getting this!!!! finally!!!! after a few years of only thinking about it due to the lack of money! finally, I'm doing the fully-automated daily backup with Windows Shadow Copy, to both a USB hard drive and the Cloud, AWS S3 Deep Glacier! I'm precisely on the right track! I'll simply keep on executing and improving all my plans, NO MATTER WHAT, NO MATTER WHAT, NO MATTER WHAT, NO MATTER WHAT, NO MATTER WHAT!!!!!!!!!!)

2. 7-Zip CLI program use method

NOTE [10/6/2020 4:51 PM CST]
~~must navigate to the root directory of the backup, so that only relative file path is stored in each list file and compressed (backup) file.~~
get the relative directory path storage in the compressed backup file absolutely right!

[10/6/2020 5:05 PM CST]
the full path of the backup root directory, with : and \ replaced with -, will be used as the unique backup name. e.g. "c:\users\allen\documents\" will be "c--users-allen-documents-".

the compressed backup file name format: (auto-generated unique backup name)-YYYYMMDD-HHMMSS
(HH in the 24-hour format)

an example
c--users-allen-documents-20201006-030000

2.1. 7-Zip CLI test commands

2.1.1. [success] backup test command

2.1.1.1. test command to execute

[10/7/2020 12:35 PM CST] execute the following command in the "C:\Temp\backup_source" directory in a terminal with or without the administrator privilege.

```
"c:\program files\7-zip\7z" a "C:\Temp\backup_destination\
full_shadowspawn_test.zip" -v4000000000b -mcu=on -mem=AES256 -
1234abcd @"C:\Users\Allen\Documents\AAM Auto Backup\
full_shadowspawn_test_listfile.txt"
```

2.1.1.2. test result

[10/7/2020 12:40 PM CST]

```
7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Scanning the drive:
5 folders, 4 files, 1363493 bytes (1332 KiB)

Creating archive: C:\Temp\backup_destination\full_shadowspawn_test.zip
```

```
Add new data to archive: 5 folders, 4 files, 1363493 bytes (1332 KiB)


Files read from disk: 4
Archive size: 1008111 bytes (985 KiB)
Everything is Ok
```

[10/7/2020 12:47 PM CST] success! the backup zip file has been created!

2.1.2. [failure] restoration test command 1

2.1.2.1. test command to execute

[10/7/2020 12:46 PM CST] execute the following command in any directory in a terminal with or without the administrator privilege. the zip file(s) must be in the directory.

```
"c:\program files\7-zip\7z" x "C:\Temp\backup_destination\
full_shadowspawn_test.zip" -o"C:\Temp\backup_destination\" -y -
1234abcd
```

2.1.2.2. test result

[10/7/2020 12:49 PM CST]

```
7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Scanning the drive for archives:

ERROR: The system cannot find the file specified.
C:\Temp\backup_destination\full_shadowspawn_test.zip



System ERROR:
The system cannot find the file specified.
```

2.1.3. [success] restoration test command 2

2.1.3.1. test command to execute

[10/7/2020 12:46 PM CST] execute the following command in any directory in a terminal with or without the administrator privilege. the zip file(s) must be in the directory.

```
"c:\program files\7-zip\7z" x "C:\Temp\backup_destination\
full_shadowspawn_test.zip.001" -o"C:\Temp\backup_destination\" -y
-1234abcd
```

2.1.3.2. test result

[10/7/2020 12:49 PM CST]

```
C:\Temp\backup_source>"c:\program files\7-zip\7z" x "C:\Temp\backup_destination\
full_shadowspawn_test.zip.001" -o"C:\Temp\backup_destination\" -y -p1234abcd
1234abcd

7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Scanning the drive for archives:
1 file, 1008111 bytes (985 KiB)

Extracting archive: C:\Temp\backup_destination\full_shadowspawn_test.zip.001
--
Path = C:\Temp\backup_destination\full_shadowspawn_test.zip.001
Type = Split
Physical Size = 1008111
Volumes = 1
Total Physical Size = 1008111
----
Path = full_shadowspawn_test.zip
Size = 1008111
--
Path = full_shadowspawn_test.zip
Type = zip
Physical Size = 1008111

Everything is Ok

Folders: 5
Files: 4
Size:       1363493
Compressed: 1008111

C:\Temp\backup_source>
```

2.2. 7-Zip CLI command for full backup

```
cd /D c:\users\allen\documents\
"c:\program files\7-zip\7z" a aam-core-archive-20200211.zip ~~-ssw~~
-v4000000000b -mcu=on -mem=AES256 -1234abcd @aam-core-archive-
directory-list.txt
```

2.3. 7-Zip CLI command for incremental backup

```
cd /D c:\users\allen\documents\

"c:\program files\7-zip\7z" a aam-core-archive-20200211.zip ~~-ssw~~
-v4000000000b -mcu=on -mem=AES256 -1234abcd @aam-core-archive-
directory-list.txt
```

2.4. 7-Zip CLI command for data restoration

[10/6/2020 5:14 PM CST]

7z e archive.zip -oc:\Doc -y **-1234abcd**
7z x archive.zip -oc:\Doc -y **-1234abcd**

2.5. archive content list file content examples

[10/6/2020 5:20 PM CST]

2.5.1. aam-core-archive-directory-list.txt contents

2.5.2. aam-big-files-archive-directory-list.txt contents

3. batch files to use

[10/6/2020 5:22 PM CST]
these files should be in the same directory as the AAM Auto Backup program file.

[10/6/2020 5:29 PM CST]
these files might have to be dynamically generated and deleted, for the custom shadow copy drive name.

3.1. AAM_Auto_Backup_shadow_copy.bat

([11/6/2020 1:59 PM CST] just use this!)

```
cd /D Q:
robocopy ...
7z ...
```

3.2. =====DEPRECATED BELOW [11/6/2020 1:58 PM CST]====

3.3. AAM_Auto_Backup_full_backup.bat

```
cd /D Q:
robocopy ...
7z ...
```

3.4. AAM_Auto_Backup_incremental_backup.bat

```
cd /D Q:
robocopy ...
7z ...
```

4. how to do AWS S3 Deep Glacier backup using the AWS CLI program

[10/2/2020 6:15 AM CST]

4.1. how to backup data to the AWS S3 Deep Glacier

4.2. how to restore data from the AWS S3 Deep Glacier

4.3. references

[10/5/2020 6:51 PM CST]
search "AWS CLI S3 Deep Glacier" online.
https://aws.amazon.com/blogs/aws/new-amazon-s3-storage-class-glacier-deep-archive/
https://docs.aws.amazon.com/prescriptive-guidance/latest/backup-recovery/amazon-s3-glacier.html
https://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html
https://aws.amazon.com/s3/pricing/
https://docs.aws.amazon.com/cli/latest/userguide/cli-services-glacier.html
https://aws.amazon.com/premiumsupport/knowledge-center/restore-s3-object-glacier-storage-class/
https://docs.aws.amazon.com/cli/latest/reference/glacier/index.html
https://docs.aws.amazon.com/cli/latest/reference/s3/index.html

https://docs.aws.amazon.com/amazonglacier/latest/dev/introduction.html

https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-getting-started.html
https://docs.aws.amazon.com/cli/latest/userguide/cli-services-glacier.html

search "AWS CLI S3 Glacier Deep Archive" online.
https://docs.aws.amazon.com/AmazonS3/latest/dev/restoring-objects.html
https://docs.aws.amazon.com/cli/latest/reference/glacier/upload-archive.html
https://docs.aws.amazon.com/cli/latest/reference/s3api/put-object.html

search "how to use AWS CLI to archive to S3 Glacier Deep Archive" online.
https://docs.amazonaws.cn/en_us/AmazonS3/latest/dev/restoring-objects.html
After you receive a temporary copy of the restored object, the object's storage class remains S3 Glacier or S3 Glacier Deep Archive. (A HEAD Object or the GET Object API operations request returns S3 Glacier or S3 Glacier Deep Archive as the storage class.)
https://docs.amazonaws.cn/en_us/AmazonS3/latest/dev/restoring-objects-console.html
You can use the Amazon S3 console to restore a copy of an object that has been archived to Amazon S3 Glacier. For instructions on how to restore an archive using the AWS Management Console, see How Do I Restore an S3 Object that has been Archived to Amazon S3 Glacier? in the Amazon Simple Storage Service Console User Guide.

https://docs.aws.amazon.com/prescriptive-guidance/latest/backup-recovery/amazon-s3-glacier.html
To automate your backup and restore processes, you can access Amazon S3 Glacier and S3 Glacier Deep Archive via the AWS Management Console, AWS CLI, and AWS SDKs. For more information, see Amazon S3 Glacier.

4.3.1. AWS S3 Glacier Deep Archive overview

[10/5/2020 6:59 PM CST]

https://aws.amazon.com/blogs/aws/new-amazon-s3-storage-class-glacier-deep-archive/

Amazon S3 Glacier Deep Archive Storage Class
The new Glacier Deep Archive storage class is designed to provide durable and secure long-term storage for large amounts of data at a price that is competitive with off-premises tape archival services. Data is stored across 3 or more AWS Availability Zones and can be retrieved in 12 hours or less. You no longer need to deal with expensive and finicky tape drives, arrange for off-premises storage, or worry about migrating data to newer generations of media.

The existing S3 Glacier storage class allows you to access your data in minutes (using expedited retrieval) and is a good fit for data that requires faster access. To learn more about the entire range of options, read Storage Classes in the S3 Developer Guide. If you are already making use of the Glacier storage class and rarely access your data, you can switch to Deep Archive and begin to see cost savings right away.

Using Glacier Deep Archive – CLI / Programmatic Access
I can use the CLI to upload a new object and set the storage class:

$ aws s3 cp new.mov s3://awsroadtrip-videos-raw/ --storage-class DEEP_ARCHIVE

I can also change the storage class of an existing object by copying it over itself:

$ aws s3 cp s3://awsroadtrip-videos-raw/new.mov s3://awsroadtrip-videos-raw/new.mov --storage-class DEEP_ARCHIVE

Now Available
The S3 Glacier Deep Archive storage class is available today in all commercial regions and in both AWS GovCloud regions. Pricing varies by region, and the storage cost is up to 75% less than for the existing S3 Glacier storage class; visit the S3 Pricing page for more information.

ok, so easier than AWS S3 Glacier to use via AWS CLI! [10/5/2020 7:00 PM CST]

4.3.1.1. test command

```
cd c:/temp

aws --profile cloud-backup-uploader s3 cp test.txt s3://aam-data-
backup/ --storage-class DEEP_ARCHIVE
```

(the above command works, beautifully, no trouble. [10/8/2020 8:32 PM CST])

4.3.2. relevant AWS CLI commands for AWS S3 Glacier Deep Archive

https://docs.aws.amazon.com/cli/latest/reference/s3/cp.html
--storage-class (string) The type of storage to use for the object. Valid choices are: STANDARD | REDUCED_REDUNDANCY | STANDARD_IA | ONEZONE_IA | INTELLIGENT_TIERING | GLACIER | DEEP_ARCHIVE. Defaults to 'STANDARD'

[10/5/2020 7:33 PM CST] I'm beginning to suspect that the AWS CLI S3 Glacier is not used for AWS S3 Glacier Deep Archive.

[10/5/2020 7:33 PM CST] search "AWS CLI how to restore S3 Glacier Deep Archive" online.
https://aws.amazon.com/premiumsupport/knowledge-center/restore-s3-object-glacier-storage-class/
https://docs.aws.amazon.com/cli/latest/reference/s3api/restore-object.html
https://docs.aws.amazon.com/AmazonS3/latest/dev/restoring-objects.html
https://docs.aws.amazon.com/AmazonS3/latest/user-guide/restore-archived-objects.html

https://stackoverflow.com/questions/20033651/how-to-restore-folders-or-entire-buckets-to-amazon-s3-from-glacier

https://aws.amazon.com/premiumsupport/knowledge-center/restore-s3-object-glacier-storage-class/
aws s3api restore-object --bucket awsexamplebucket --key dir1/example.obj --restore-request '{"Days":25,"GlacierJobParameters":{"Tier":"Standard"}}'

After you run this command, a temporary copy of the object is made available for the duration specified in the restore request, such as the 25 days used in this example.

Monitor the status of your restore request
Run the following command to monitor the status of your restore request:
aws s3api head-object --bucket awsexamplebucket --key dir1/example.obj

[10/5/2020 7:33 PM CST] search "AWS CLI S3 how I check if an object has been restored?" online.
(no need to search. the answer is above. [10/5/2020 7:38 PM CST])

[10/5/2020 7:41 PM CST] to copy the AWS S3 object to a local data storage, use the following.
aws s3 cp s3://awsexamplebucket/dir1/example.obj c://awsexamplebucket/dir2/example2.obj

[10/5/2020 7:40 PM CST] **awesome! much easier than using the AWS CLI commands for AWS S3 Glacier!!!!**

https://docs.aws.amazon.com/cli/latest/reference/s3/cp.html
https://docs.aws.amazon.com/cli/latest/reference/s3api/index.html
https://docs.aws.amazon.com/cli/latest/reference/s3api/restore-object.html
https://docs.aws.amazon.com/AmazonS3/latest/dev/restoring-objects.html
https://docs.aws.amazon.com/AmazonS3/latest/user-guide/restore-archived-objects.html
https://stackoverflow.com/questions/20033651/how-to-restore-folders-or-entire-buckets-to-amazon-s3-from-glacier

https://stackoverflow.com/questions/20033651/how-to-restore-folders-or-entire-buckets-to-amazon-s3-from-glacier
If you're using the AWS CLI tool (it's nice, you should), you can do it like this:
aws s3 ls s3://<bucket_name> | awk '{print $4}' | xargs -L 1 aws s3api restore-object --restore-request Days=<days> --bucket <bucket_name> --key

The above answers didn't work well for me because my bucket was mixed with objects on Glacier and some that were not. The easiest thing for me was to create a list of all GLACIER objects in the bucket, then attempt to restore each one individually, ignoring any errors (like already in progress, not an object, etc).

Get a listing of all GLACIER files (keys) in the bucket

```
aws s3api list-objects-v2 --bucket <bucketName> --query "Contents[?
StorageClass=='GLACIER']" --output text | awk '{print $2}' > glacier-restore.txt
```

Create a shell script and run it, replacing your "bucketName".

```
#!/bin/sh
for x in `cat glacier-restore.txt`
  do
    echo "Begin restoring $x"
    aws s3api restore-object --restore-request Days=7 --bucket <bucketName> --key "$x"
    echo "Done restoring $x"
  done
```

I recently needed to restore a whole bucket and all its files and folders. You will need s3cmd and aws cli tools configured with your credentials to run this.

4.3.3. how to set expiration on the objects in AWS S3 Glacier Deep Archive

[10/5/2020 8:52 PM CST]

find how to auto-expire and auto-delete the objects in AWS S3 Glacier Deep Archive.
[10/5/2020 8:54 PM CST]

[10/5/2020 8:54 PM CST]
search "how to set expiration on the objects in AWS S3 Glacier Deep Archive" online?

how about this? create an AWS S3 bucket for the backup, name it properly, and set life cycle rule with auto-delete when older than 90 day, and auto-archive to AWS S3 Glacier Deep Archive when older than one day? actually, "auto-archive to AWS S3 Glacier Deep Archive when older than one day" will make the restoration coding complex; so, just try life cycle rule with auto-delete when older than 181 days (the required minimum storage days for AWS S3 Glacier Deep Archive are 180 days). see if that works.

https://aws.amazon.com/s3/pricing/
Objects that are archived to S3 Glacier and S3 Glacier Deep Archive have a minimum 90 days and 180 days of storage, respectively. Objects deleted before 90 days and 180 days incur a pro-rated charge equal to the storage charge for the remaining days.

S3 Glacier ** - For long-term backups and archives with retrieval option from 1 minute to 12 hours
All Storage / Month: $0.004 per GB

S3 Glacier Deep Archive ** - For long-term data archiving that is accessed once or twice in a year and can be restored within 12 hours
All Storage / Month: $0.00099 per GB

([10/5/2020 9:04 PM CST] note that the minimum storage duration for S3 Glacier Deep Archive is twice as long as S3 Glacier. even still, S3 Glacier Deep Archive monthly storage cost is $0.00198 per GB, about 50% cheaper than the S3 Glacier monthly storage cost.)

S3 Glacier ** Data Retrieval requests (per 1,000 requests) | Data retrievals (per GB)
Standard $0.05 $0.01
Bulk $0.025 $0.0025

S3 Glacier Deep Archive ** Data Retrieval requests (per 1,000 requests) | Data retrievals (per GB)
Standard $0.10 $0.02
Bulk $0.025 $0.0025

5. AWS Python API usage techniques

[10/13/2020 3:46 PM CST]
In the 'Implementation Techniques' part of the AAM Auto Backup document, create a chapter titled 'AWS Python API usage techniques'.

https://boto3.amazonaws.com/v1/documentation/api/latest/guide/index.html

5.1. AWS Python API usage for data backup

[10/13/2020 3:48 PM CST]

5.1.1. AWS S3 references

5.1.1.1. AWS S3 object listing and uploading

[10/18/2020 6:49 PM CST]
    #https://boto3.amazonaws.com/v1/documentation/api/latest/index.html
    #https://boto3.amazonaws.com/v1/documentation/api/latest/guide/s3-examples.html
    #https://boto3.amazonaws.com/v1/documentation/api/latest/guide/s3-example-creating-buckets.html
    #https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Bucket.objects
    #bucket.objects.limit()
    #
    #search "Python how to check internet connection' online.
    #https://stackoverflow.com/questions/3764291/checking-network-connection

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#bucket filter(**kwargs)

```
bucket.objects.filter()
```

[10/18/2020 7:39 PM CST] providing credentials for an Boto3 session
https://boto3.amazonaws.com/v1/documentation/api/latest/guide/credentials.html
https://boto3.amazonaws.com/v1/documentation/api/latest/guide/resources.html

[10/18/2020 8:36 PM CST] AWS S3 file upload
https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Object.upload_file

[10/19/2020 8:56 AM CST] search "AWS boto3 upload to AWS S3 Glacier Deep Archive" online.
https://stackoverflow.com/questions/59039076/how-to-upload-a-file-to-amazon-glacier-deep-archive-using-boto3
You can either use put_object(), which has a StorageClass parameter, or you can use upload_file() while specifying ExtraArgs:

```
import boto3
s3_client = boto3.client('s3')
s3_client.upload_file('/tmp/hello.txt', 'my-bucket', 'hello.txt', ExtraArgs={'StorageClass':
'DEEP_ARCHIVE'})
```

Permitted ExtraArgs can be found at ALLOWED_UPLOAD_ARGS.
https://boto3.amazonaws.com/v1/documentation/api/1.9.42/reference/customizations/s3.html#boto3.s3.transfer.S3Transfer.ALLOWED_UPLOAD_ARGS

5.1.1.2. AWS S3 object restoration

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Client.restore_object

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Object.restore_object

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.ObjectSummary.restore_object

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#restore-glacier-objects-in-an-amazon-s3-bucket
Restore Glacier objects in an Amazon S3 bucket

The following example shows how to initiate restoration of glacier objects in an Amazon S3 bucket, determine if a restoration is on-going, and determine if a restoration is finished.

```
import boto3
```

```
s3 = boto3.resource('s3')
bucket = s3.Bucket('glacier-bucket')
for obj_sum in bucket.objects.all():
    obj = s3.Object(obj_sum.bucket_name, obj_sum.key)
    if obj.storage_class == 'GLACIER':
        # Try to restore the object if the storage class is glacier and
        # the object does not have a completed or ongoing restoration
        # request.
        if obj.restore is None:
            print('Submitting restoration request: %s' % obj.key)
            obj.restore_object(RestoreRequest={'Days': 1})
        # Print out objects whose restoration is on-going
        elif 'ongoing-request="true"' in obj.restore:
            print('Restoration in-progress: %s' % obj.key)
        # Print out objects whose restoration is complete
        elif 'ongoing-request="false"' in obj.restore:
            print('Restoration complete: %s' % obj.key)
```

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#object

https://aws.amazon.com/premiumsupport/knowledge-center/restore-s3-object-glacier-storage-class/

```
aws s3api restore-object --bucket awsexamplebucket --key dir1/example.obj --restore-request '{"Days":25,"GlacierJobParameters":{"Tier":"Standard"}}'
```

After you run this command, a temporary copy of the object is made available for the duration specified in the restore request, such as the 25 days used in this example.

Monitor the status of your restore request
Run the following command to monitor the status of your restore request:

```
aws s3api head-object --bucket awsexamplebucket --key dir1/example.obj
```

After you run the command, and if the restore is still in progress, you receive a response similar to the following:

```
{
    "Restore": "ongoing-request=\"true\"",
    ...
    "StorageClass": "GLACIER",
    "Metadata": {}
}
```

After the restore is complete, you receive a response similar to the following:

```
{
   "Restore": "ongoing-request=\"false\", expiry-date=\"Sun, 13 Aug 2017 00:00:00 GMT\"",
   ...
   "StorageClass": "GLACIER",
   "Metadata": {}
}
```

Note the expiry-date in the response—you have until this time to access the temporary store object (stored in the Reduced Redundancy storage class). The temporary object is made available alongside the archived object that's in the S3 Glacier storage class. After the expiry-date elapses, the temporary object is removed. You must change the object's storage class before the temporary object expires. To change the object's storage class after the expiry-date, you must initiate a new restore request.

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Client.head_object
The HEAD operation retrieves metadata from an object without returning the object itself. This operation is useful if you're only interested in an object's metadata. To use HEAD, you must have READ access to the object.
Request Syntax

```
response = client.head_object(
   Bucket='string',
   IfMatch='string',
   IfModifiedSince=datetime(2015, 1, 1),
   IfNoneMatch='string',
   IfUnmodifiedSince=datetime(2015, 1, 1),
   Key='string',
   Range='string',
   VersionId='string',
   SSECustomerAlgorithm='string',
   SSECustomerKey='string',
   RequestPayer='requester',
   PartNumber=123,
   ExpectedBucketOwner='string'
)
```

Examples

The following example retrieves an object metadata.

```
response = client.head_object(
   Bucket='examplebucket',
   Key='HappyFace.jpg',
)
```

```
print(response)
```

Expected Output:

```
{
    'AcceptRanges': 'bytes',
    'ContentLength': '3191',
    'ContentType': 'image/jpeg',
    'ETag': '"6805f2cfc46c0f04559748bb039d69ae"',
    'LastModified': datetime(2016, 12, 15, 1, 19, 41, 3, 350, 0),
    'Metadata': {
    },
    'VersionId': 'null',
    'ResponseMetadata': {
        '...': '...',
    },
}
```

class S3.Waiter.ObjectExists
    waiter = client.get_waiter('object_exists')
    wait(**kwargs)
        Polls S3.Client.head_object() every 5 seconds until a successful state is reached. An error is returned after 20 failed checks.

class S3.Waiter.ObjectNotExists
    waiter = client.get_waiter('object_not_exists')
    wait(**kwargs)
        Polls S3.Client.head_object() every 5 seconds until a successful state is reached. An error is returned after 20 failed checks.

load()
    Calls S3.Client.head_object() to update the attributes of the Object resource. Note that the load and reload methods are the same method and can be used interchangeably.

wait_until_exists(**kwargs)
    Waits until this Object is exists. This method calls S3.Waiter.object_exists.wait() which polls. S3.Client.head_object() every 5 seconds until a successful state is reached. An error is returned after 20 failed checks.

wait_until_not_exists(**kwargs)

Waits until this Object is not exists. This method calls S3.Waiter.object_not_exists.wait() which polls. S3.Client.head_object() every 5 seconds until a successful state is reached. An error is returned after 20 failed checks.

load(*args, **kwargs)
    Calls s3.Client.head_object to update the attributes of the ObjectSummary resource.

5.1.1.3. AWS S3 object downloading

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Bucket.download_file

```
import boto3
s3 = boto3.resource('s3')
s3.Bucket('mybucket').download_file('hello.txt', '/tmp/hello.txt')
```

[10/5/2020 7:41 PM CST] to copy the AWS S3 object to a local data storage, use the following.
aws s3 cp s3://awsexamplebucket/dir1/example.obj c://awsexamplebucket/dir2/example2.obj

5.1.2. AWS S3 testing

5.1.2.1. upload waiting

```
boto3_session = boto3.Session( \
    aws_access_key_id=strBackupAwsIamUserAccessKeyId,
    aws_secret_access_key=strBackupAwsIamUserSecretAccessKey)
s3_resource = boto3_session.resource('s3')
backup_s3_bucket = s3_resource.Bucket(strBackupAwsS3BucketName)

s3_objectToAdd = backup_s3_bucket.upload_file( \
    'C:\\Temp\\stock intro.m4a',
    'test_20201019_093103.777777_c--temp-_full-backup.zip.001',
    ExtraArgs={'StorageClass': 'DEEP_ARCHIVE'})
s3_objectToAdd.wait_until_exists()
```

[10/19/2020 9:47 AM CST] the above produces the following error in terminal.

```
Traceback (most recent call last):
  File "AWS_S3_tester.py", line 22, in <module>
```

```
    s3_objectToAdd.wait_until_exists()
AttributeError: 'NoneType' object has no attribute 'wait_until_exists'
```

[10/19/2020 9:48 AM CST] the object got uploaded.

[10/19/2020 9:48 AM CST] so, use backup_s3_bucket.upload_file() without a return value.

5.1.2.2. object listing

[10/19/2020 9:52 AM CST]

```
iteratorAwsS3Objects = backup_s3_bucket.objects.filter( \
    Prefix='test_202010')

for s3_object in iteratorAwsS3Objects:
    print(str(s3_object))
```

[10/19/2020 9:52 AM CST] the above code produces the following error.

```
C:\Users\Allen\Documents\
\AAM Auto Backup>python AWS_S3_tester.py
Traceback (most recent call last):
  File "AWS_S3_tester.py", line 31, in <module>
    for s3_object in iteratorAwsS3Objects:
  File "C:\Users\Allen\AppData\Local\Programs\Python\Python38\lib\site-packages\
boto3\resources\collection.py", line 83, in __iter__
    for page in self.pages():
  File "C:\Users\Allen\AppData\Local\Programs\Python\Python38\lib\site-packages\
boto3\resources\collection.py", line 166, in pages
    for page in pages:
  File "C:\Users\Allen\AppData\Local\Programs\Python\Python38\lib\site-packages\
botocore\paginate.py", line 255, in __iter__
    response = self._make_request(current_kwargs)
  File "C:\Users\Allen\AppData\Local\Programs\Python\Python38\lib\site-packages\
botocore\paginate.py", line 332, in _make_request
    return self._method(**current_kwargs)
  File "C:\Users\Allen\AppData\Local\Programs\Python\Python38\lib\site-packages\
botocore\client.py", line 357, in _api_call
    return self._make_api_call(operation_name, kwargs)
  File "C:\Users\Allen\AppData\Local\Programs\Python\Python38\lib\site-packages\
botocore\client.py", line 676, in _make_api_call
    raise error_class(parsed_response, operation_name)
botocore.exceptions.ClientError: An error occurred (AccessDenied) when calling t
he ListObjects operation: Access Denied

C:\Users\Allen\Documents\
```

\AAM Auto Backup>

[10/19/2020 9:52 AM CST] so, grant ListObjects permission.
UPDATE [10/19/2020 10:33 AM CST] s3:ListBucket is the permission that's required in the AWS S3 bucket policy JSON for listing objects in the bucket. more details in the "How to configure AWS S3 for AAM Auto Backup" section of the Online Content-Marketing Website Mastery audiovisual script.

5.2. AWS Python API usage for data restoration

[10/13/2020 3:48 PM CST]

6. Use a class named Aam_Auto_Backup to implement all the functionalities

[10/5/2020 7:57 PM CST]

7. backup directories computation algorithms

[10/21/2020 4:47 PM CST]

7.1. Robocopy test 2

[11/28/2020 3:13 PM CST]
for finding out whether using wild cards in exclusion directory and file paths and names is supported by Robocopy.

7.1.1. test backup source directory structure

[11/28/2020 4:03 PM CST]

- 01test1
- 01test2
- 01test3
- 02test1
  - test1.txt
  - test2.txt
- 02test2
  - test1.txt
  - test2.txt
- 02test3
  - 1 (folder)
- 03test1
- 03test2
- 03test3

7.1.2. test cases

7.1.2.1. [failure] test case 1: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\*\backup_source\03test*" /xf "*1.txt" "C:\*\
backup_source\02test1\test*.txt"
```

RESULT [11/28/2020 4:31 PM CST]
Robocopy failure. invalid parameter

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\*\backup_source\03test*" /xf "*1.txt" "C:\*\backup_source\02test1\
test*.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:30:29 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
 Exc Dirs : 01test*

  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #6 : "C:\*\backup_source\03test*"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.
   For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.1.2.2. [failure] test case 2: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test*" /xf "*1.txt" "C:\*\
backup_source\02test1\test*.txt"
```

RESULT [11/28/2020 4:34 PM CST]
Robocopy failure. invalid parameter. **Robocopy has no wild card support in exclusion directory names at least for a full path!**

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test*" /xf "*1.txt" "C:\*\backup_source\02tes
t1\test*.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:32:12 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
 Exc Dirs : 01test*

  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #6 : "C:\Temp\backup_source\03test*"

       Simple Usage :: ROBOCOPY source destination /MIR
```

```
          source :: Source Directory (drive:\path or \\server\share\path).
     destination :: Destination Dir  (drive:\path or \\server\share\path).
            /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.1.2.3. [failure] test case 3: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "C:\*\
backup_source\02test1\test*.txt"
```

RESULT [11/28/2020 4:31 PM CST]
Robocopy failure. invalid parameter.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "C:\*\backup_source\02tes
t1\test*.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:35:05 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
Exc Files : *1.txt

 Exc Dirs : C:\Temp\backup_source\03test1
            01test*
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #9 : "C:\*\backup_source\02test1\test*.txt"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.1.2.4. [failure] test case 4: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "C:\Temp\
backup_source\02test1\test*.txt"
```

RESULT [11/28/2020 4:31 PM CST]
Robocopy failure. invalid parameter. **Robocopy has no wild card support in exclusion file names at least for a full path!**

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "C:\Temp\backup_source\02
test1\test*.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:36:05 2020
```

```
   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
Exc Files : *1.txt

 Exc Dirs : C:\Temp\backup_source\03test1
            01test*

  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #9 : "C:\Temp\backup_source\02test1\test*.txt"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.1.2.5. [success] test case 5: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "C:\Temp\
backup_source\02test1\test2.txt"
```

RESULT [11/28/2020 4:31 PM CST]
Robocopy success. exactly the desired results produced.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "C:\Temp\backup_source\02
```

```
test1\test2.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:38:10 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : C:\Temp\backup_source\02test1\test2.txt
            *1.txt

 Exc Dirs : C:\Temp\backup_source\03test1
            01test*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\Temp\backup_source\
          *EXTRA File               162        ~$test.doc
          New Dir          2    C:\Temp\backup_source\02test1\
          New Dir          2    C:\Temp\backup_source\02test2\
100%        New File                  0        test2.txt
          New Dir          0    C:\Temp\backup_source\02test3\
          New Dir          0    C:\Temp\backup_source\03test2\
          New Dir          0    C:\Temp\backup_source\03test3\

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :        10         5         5         0         0         0
   Files :         4         1         3         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Sat Nov 28 16:38:11 2020

C:\Users\Allen>
```

7.1.2.6. [failure] test case 6: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" "02test3\1" /xf
"*1.txt" "C:\Temp\backup_source\02test1\test2.txt"
```

RESULT [11/28/2020 4:31 PM CST]
failure. 02test3\1 is copied. it seems that Robocopy supports only directory name and full directory path exclusion, no relative path exclusion.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" "02test3\1" /xf "*1.txt" "C:\Temp\back
up_source\02test1\test2.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:43:18 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : C:\Temp\backup_source\02test1\test2.txt
            *1.txt

 Exc Dirs : C:\Temp\backup_source\03test1
            02test3\1
            01test*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\Temp\backup_source\
          *EXTRA File              162        ~$test.doc
                           2    C:\Temp\backup_source\02test1\
                           2    C:\Temp\backup_source\02test2\
                           0    C:\Temp\backup_source\02test3\
        New Dir         0    C:\Temp\backup_source\02test3\1\
                  0    C:\Temp\backup_source\03test2\
                  0    C:\Temp\backup_source\03test3\

------------------------------------------------------------------------------

             Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :       11         1        10         0         0         0
  Files :        4         0         4         0         0         1
  Bytes :        0         0         0         0         0       162
  Times :   0:00:00   0:00:00                       0:00:00   0:00:00

  Ended : Sat Nov 28 16:43:18 2020

C:\Users\Allen>
```

7.1.2.7. [failure] test case 7: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" "\02test3\1" /xf
"*1.txt" "C:\Temp\backup_source\02test1\test2.txt"
```

RESULT [11/28/2020 4:46 PM CST]
failure. \02test3\1 is copied. **Robocopy supports only directory name and full directory path exclusion, no relative path exclusion.**

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" "\02test3\1" /xf "*1.txt" "C:\Temp\bac
kup_source\02test1\test2.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:46:30 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\
   Files : *.*

Exc Files : C:\Temp\backup_source\02test1\test2.txt
          *1.txt

 Exc Dirs : C:\Temp\backup_source\03test1
          \02test3\1
          01test*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                      0    C:\Temp\backup_source\
        *EXTRA File              162        ~$test.doc
        New Dir          2    C:\Temp\backup_source\02test1\
        New Dir          2    C:\Temp\backup_source\02test2\
100%        New File                 0        test2.txt
        New Dir          0    C:\Temp\backup_source\02test3\
        New Dir          0    C:\Temp\backup_source\02test3\1\
        New Dir          0    C:\Temp\backup_source\03test2\
        New Dir          0    C:\Temp\backup_source\03test3\

------------------------------------------------------------------------------

            Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :       11         6         5         0         0         0
  Files :        4         1         3         0         0         1
  Bytes :        0         0         0         0         0       162
  Times :   0:00:00   0:00:00                       0:00:00   0:00:00

  Ended : Sat Nov 28 16:46:30 2020

C:\Users\Allen>
```

7.1.2.8. [failure] test case 8: Robocopy directory and file exclusion test

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "02test1\
test2.txt"
```

RESULT [11/28/2020 4:46 PM CST]

failure. 02test1\test2.txt is copied. **Robocopy supports only file name and full file path exclusion, no relative file path exclusion.**

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"01test*" "C:\Temp\backup_source\03test1" /xf "*1.txt" "02test1\test2.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Sat Nov 28 16:59:14 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : 02test1\test2.txt
            *1.txt

 Exc Dirs : C:\Temp\backup_source\03test1
            01test*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\Temp\backup_source\
          *EXTRA File               162        ~$test.doc
          New Dir          2    C:\Temp\backup_source\02test1\
100%        New File                  0        test2.txt
          New Dir          2    C:\Temp\backup_source\02test2\
100%        New File                  0        test2.txt
          New Dir          0    C:\Temp\backup_source\02test3\
          New Dir          0    C:\Temp\backup_source\02test3\1\
          New Dir          0    C:\Temp\backup_source\03test2\
          New Dir          0    C:\Temp\backup_source\03test3\

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :        11         6         5         0         0         0
   Files :         4         2         2         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Sat Nov 28 16:59:14 2020

C:\Users\Allen>
```

7.1.3. Robocopy exclusion capability conclusions

[11/28/2020 4:47 PM CST]
Robocopy supports only directory name and full/absolute directory path exclusion, no relative path exclusion.

Robocopy has no wild card support in exclusion directory full/absolute path. wild card support only in directory names, not absolute directory paths.

Robocopy supports only file name and full/absolute file path exclusion, no relative file path exclusion.

Robocopy has no wild card support in exclusion file full/absolute path. wild card support only in file names, not absolute file paths.

7.2. Robocopy test

[10/22/2020 9:18 PM CST]

7.2.1. test requirements

both relative and absolute directory paths supported? perform Robocopy test(s) and find out!

both file full paths and file full names supported? perform Robocopy test(s) and find out!

7.2.2. test cases

7.2.2.1. test case 1: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create sub-dir1\, sub-dir2\, and sub-dir3\.

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"sub-dir2" "c:\temp\backup_source\sub-dir3"
```

RESULT [10/22/2020 9:36 PM CST]
the above Robocopy command excludes both sub-dir2\ and sub-dir3\.

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"sub-dir2" "c:\temp\backup_source\sub-dir3"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 21:34:55 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

 Exc Dirs : sub-dir2
            c:\temp\backup_source\sub-dir3

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\Temp\backup_source\
          *EXTRA File                 162        ~$test.doc
          New Dir          0    C:\Temp\backup_source\sub-dir1\

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         4         1         3         0         0         0
   Files :         0         0         0         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Thu Oct 22 21:34:55 2020

C:\Users\Allen>
```

7.2.2.2. test case 2: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "sub-dir1\", and "-sub-dir2\".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"-*"
```

RESULT [10/22/2020 9:57 PM CST]
the above robocopy command fails, saying "invalid parameter" (as expected).

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"-*"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 21:57:23 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #5 : "-*"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !
```

```
C:\Users\Allen>
```

7.2.2.3. test case 3: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create sub-dir1\test, and test\.

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"test"
```

RESULT [10/22/2020 10:01 PM CST]
the above Robocopy command only copies sub-dir1. that is, it excludes every directory named "test".

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"test"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:00:55 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

 Exc Dirs : test

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\Temp\backup_source\
          *EXTRA File              162        ~$test.doc
          New Dir          0    C:\Temp\backup_source\sub-dir1\

------------------------------------------------------------------------------
```

```
          Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :         4         1         3         0         0         0
  Files :         0         0         0         0         0         1
  Bytes :         0         0         0         0         0       162
  Times :   0:00:00   0:00:00                       0:00:00   0:00:00

  Ended : Thu Oct 22 22:00:55 2020

C:\Users\Allen>
```

7.2.2.4. test case 4: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create sub-dir1\test, and test\.

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"\test"
```

RESULT [10/22/2020 10:03 PM CST]
the above Robocopy command copies all the directories. no omission.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"\test"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:03:30 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

 Exc Dirs : \test

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------
```

```
                0    C:\Temp\backup_source\
        *EXTRA File              162        ~$test.doc
        New Dir         0    C:\Temp\backup_source\sub-dir1\
        New Dir         0    C:\Temp\backup_source\sub-dir1\test\
        New Dir         0    C:\Temp\backup_source\test\

------------------------------------------------------------------------------

            Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :        4         3         1         0         0         0
  Files :        0         0         0         0         0         1
  Bytes :        0         0         0         0         0       162
  Times :   0:00:00   0:00:00                       0:00:00   0:00:00

  Ended : Thu Oct 22 22:03:31 2020

C:\Users\Allen>
```

7.2.2.5. test case 5: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create sub-dir1\test, and test\.

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"test\"
```

RESULT [10/22/2020 10:05 PM CST]
the above Robocopy command copies all the directories. no omission.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
"test\"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:05:34 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\
   Files : *.*

 Exc Dirs : test"

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                  0    C:\Temp\backup_source\
        *EXTRA File              162        ~$test.doc
        New Dir          0    C:\Temp\backup_source\sub-dir1\
        New Dir          0    C:\Temp\backup_source\sub-dir1\test\
        New Dir          0    C:\Temp\backup_source\test\

------------------------------------------------------------------------------

            Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :        4         3         1         0         0         0
  Files :        0         0         0         0         0         1
  Bytes :        0         0         0         0         0       162
  Times :   0:00:00   0:00:00                       0:00:00   0:00:00

  Ended : Thu Oct 22 22:05:34 2020

C:\Users\Allen>
```

7.2.2.6. test case 6: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create sub-dir1\test, and test\.

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
".\test"
```

RESULT [10/22/2020 10:06 PM CST]
the above Robocopy command copies all the directories. no omission.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
".\test"
```

```
-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:06:48 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

 Exc Dirs : .\test

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\Temp\backup_source\
          *EXTRA File              162        ~$test.doc
          New Dir          0    C:\Temp\backup_source\sub-dir1\
          New Dir          0    C:\Temp\backup_source\sub-dir1\test\
          New Dir          0    C:\Temp\backup_source\test\

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         4         3         1         0         0         0
   Files :         0         0         0         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Thu Oct 22 22:06:48 2020

C:\Users\Allen>
```

7.2.2.7. test case 7: Robocopy directory exclusion test

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create sub-dir1\test, and test\.

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
".\test\"
```

RESULT [10/22/2020 10:08 PM CST]
the above Robocopy command copies all the directories. no omission.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xd
".\test\"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:08:23 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

 Exc Dirs : .\test"

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                          0    C:\Temp\backup_source\
          *EXTRA File               162        ~$test.doc
          New Dir          0    C:\Temp\backup_source\sub-dir1\
          New Dir          0    C:\Temp\backup_source\sub-dir1\test\
          New Dir          0    C:\Temp\backup_source\test\

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         4         3         1         0         0         0
   Files :         0         0         0         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Thu Oct 22 22:08:23 2020

C:\Users\Allen>
```

7.2.2.8. test case 8: Robocopy file exclusion test

[10/22/2020 9:58 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"-*" "C:\Temp\test3.txt"
```

RESULT [10/22/2020 10:12 PM CST]
the above Robocopy command says invalid parameter, surprisingly.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"-*" "C:\Temp\test3.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:12:00 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #5 : "-*"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.2.2.9. test case 9: Robocopy file exclusion test

[10/22/2020 9:58 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"-?" "C:\Temp\test3.txt"
```

RESULT [10/22/2020 10:13 PM CST]
the above Robocopy command says invalid parameter, surprisingly.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"-?" "C:\Temp\test3.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:14:03 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #5 : "-?"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?
```

```
****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.2.2.10. test case 10: Robocopy file exclusion test

[10/22/2020 10:15 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"-*.*" "C:\Temp\backup_source\test3.txt"
```

RESULT [10/22/2020 10:14 PM CST]
the above Robocopy command says invalid parameter.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"-*.*" "C:\Temp\test3.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:15:20 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #5 : "-*.*"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
```

```
              /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.2.2.11. test case 11: Robocopy file exclusion test

[10/22/2020 10:24 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"\-*.*" "C:\Temp\backup_source\test3.txt"
```

RESULT [10/22/2020 10:24 PM CST]
the above Robocopy command says invalid parameter.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"\-*.*" "C:\Temp\backup_source\test3.txt"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:24:41 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #5 : "\-*.*"

       Simple Usage :: ROBOCOPY source destination /MIR
```

```
            source :: Source Directory (drive:\path or \\server\share\path).
       destination :: Destination Dir  (drive:\path or \\server\share\path).
              /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !

C:\Users\Allen>
```

7.2.2.12. test case 12: Robocopy file exclusion test

[10/22/2020 10:27 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"t*.*"
```

RESULT [10/22/2020 10:27 PM CST]
the above Robocopy command copies only -test1.txt. so, apparently, Robocopy doesn't support - at the start in the /xf argument.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"t*.*"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:27:58 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : t*.*
```

```
  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                       3    C:\Temp\backup_source\
        *EXTRA File              162       ~$test.doc
100%       New File                 0       -test1.txt

------------------------------------------------------------------------------

            Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :        1        0        1        0        0        0
  Files :        3        1        2        0        0        1
  Bytes :        0        0        0        0        0      162
  Times :   0:00:00   0:00:00                   0:00:00   0:00:00

  Ended : Thu Oct 22 22:27:58 2020

C:\Users\Allen>
```

7.2.2.13. test case 13: Robocopy file exclusion test

[10/22/2020 10:32 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "~test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"~*.*"
```

RESULT [10/22/2020 10:32 PM CST]
the above Robocopy command copies "test2.txt" and "test3.txt".

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"~*.*"

------------------------------------------------------------------------------
  ROBOCOPY     ::     Robust File Copy for Windows

------------------------------------------------------------------------------
```

```
  Started : Thu Oct 22 22:32:08 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : ~*.*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                          3    C:\Temp\backup_source\
100%        New File               0        test2.txt
100%        New File               0        test3.txt

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         1         0         1         0         0         0
   Files :         3         2         1         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Thu Oct 22 22:32:08 2020

C:\Users\Allen>
```

7.2.2.14. test case 14: Robocopy file exclusion test

[10/22/2020 10:32 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"--*.*"
```

RESULT [10/22/2020 10:32 PM CST]
the above Robocopy command says invalid parameter.
the solution (workaround) below.
https://stackoverflow.com/questions/30607819/robocopy-files-starting-with-a-dash-result-in-error

FINAL CONCLUSION [10/22/2020 10:38 PM CST] not using /xf with starting - is probably the best way to go.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"--*.*"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:33:54 2020

   Source - C:\Temp\backup_source\
     Dest - C:\Temp\backup_destination\

    Files :
  Options : /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

ERROR : Invalid Parameter #5 : "--*.*"

       Simple Usage :: ROBOCOPY source destination /MIR

             source :: Source Directory (drive:\path or \\server\share\path).
        destination :: Destination Dir  (drive:\path or \\server\share\path).
               /MIR :: Mirror a complete directory tree.

    For more usage information run ROBOCOPY /?


****  /MIR can DELETE files as well as copy them !
```

7.2.2.15. test case 15: Robocopy file exclusion test

[10/22/2020 10:42 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "--test1.txt", "test2.txt", and "test3.txt".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"*?-*.*"
```

RESULT [10/22/2020 10:42 PM CST]
The above Robocopy command copies "-test1.txt", "test2.txt", and "test3.txt".
FINAL CONCLUSION [10/22/2020 10:45 PM CST] forget about excluding files starting with - when using Robocopy. not worth bothering with.

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"*?-*.*"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:43:44 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : *?-*.*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           4    C:\Temp\backup_source\
          *EXTRA File               162        ~$test.doc
100%        New File                  0        -test1.txt
100%        New File                  0        test2.txt
100%        New File                  0        test3.txt

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         1         0         1         0         0         0
   Files :         4         3         1         0         0         1
   Bytes :         0         0         0         0         0       162
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Thu Oct 22 22:43:45 2020

C:\Users\Allen>
```

7.2.2.16. test case 16: Robocopy file exclusion test

[10/22/2020 10:51 PM CST]

delete all the contents of C:\Temp\backup_destination.

In C:\Temp\backup_source, delete everything, and create "-test1.txt", "--test1.txt", "test2.txt", "test3.txt", and "~WRL3620.tmp".

execute the following Robocopy command.

```
robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"~WRL*.tmp"
```

RESULT [10/22/2020 10:51 PM CST]
The above copies all the files except "~WRL3620.tmp".

```
C:\Users\Allen>robocopy C:\Temp\backup_source C:\Temp\backup_destination /e /xf
"~WRL*.tmp"

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Thu Oct 22 22:53:28 2020

   Source : C:\Temp\backup_source\
     Dest : C:\Temp\backup_destination\

    Files : *.*

Exc Files : ~WRL*.tmp

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           5    C:\Temp\backup_source\
          *EXTRA File               162        ~$test.doc
100%        New File                  0        -test1.txt
100%        New File                  0        --test1.txt
100%        New File                  0        test2.txt
100%        New File                  0        test3.txt
```

```
------------------------------------------------------------------------------

            Total    Copied   Skipped  Mismatch    FAILED    Extras
   Dirs :        1         0         1         0         0         0
  Files :        5         4         1         0         0         1
  Bytes :        0         0         0         0         0       162
  Times :   0:00:00   0:00:00                       0:00:00   0:00:00

  Ended : Thu Oct 22 22:53:29 2020

C:\Users\Allen>
```

7.2.3. Robocopy references

https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/robocopy

robocopy C:\Temp\backup_source C:\Temp\backup_destination /e

/xf <filename>[...]
Excludes files that match the specified names or paths. Wildcard characters (* and ?) are supported.

/xd <directory>[...]
Excludes directories that match the specified names and paths.

[10/22/2020 10:17 PM CST]
search "Windows robocopy /xf wildcard examples" online.
https://serverfault.com/questions/304896/wildcard-directory-exclusions-with-robocopy-weird-case
https://superuser.com/questions/1353959/robocopy-exclude-directories-with-wildcard
https://pureinfotech.com/exclude-files-folders-robocopy-windows-10/
https://community.spiceworks.com/topic/2022209-how-to-exclude-files-using-robocopy
https://stackoverflow.com/questions/22513117/robocopy-to-wildcard-directory

[10/22/2020 10:34 PM CST]
search "Windows robocopy exclude file starting with minus" online.
https://stackoverflow.com/questions/30607819/robocopy-files-starting-with-a-dash-result-in-error
https://support.microsoft.com/en-us/help/2646454
Here's a possible workaround:

```
robocopy c:\temp c:\temp2 *-a.txt /xf *?-a.txt
```

*-a.txt will still match "-a.txt", but it also matches "x-a.txt", "xx-a.txt", etc.

The /xf file exclusion knocks out "x-a.txt", "xx-a.txt", and any other file with characters (specifically, at least one character) in front of the hyphen.

I've confirmed that the above command will match only "-a.txt" even if c:\temp also contains these files:

7.3. shadow-copy mount directories computation algorithm

backup-source directory factoring algorithm

7.3.1. update [11/28/2020 6:28 AM CST] new shortest or earliest shared or common parent directories finding algorithm

```
#FINAL DECISION [11/28/2020 6:09 AM CST]
#update the shortest or earliest shared or common parent directory
#finding code above, with the algorithm at the end of this note.
#FINAL DECISION [11/28/2020 6:19 AM CST]
#I'm implementing this now, not later!
#[11/28/2020 5:37 AM CST] algorithm optimization idea
#keep iInnerListIndex that is for
#str_listFinalShortestMatchingDirectoryPath.  then, using
#that index, remove the matched directory path from
#str_list_listUserSpecifiedBackupSourceDirectoryFullPaths,
#as applicable, to reduce the number of operations.
#also, implementing the above needs also checking in
#str_list_listShadowMountDirectoryFullPaths at the end,
#after the second, inner for loop.
#actually, to be more efficient, at the end,
#after the second, inner for loop, all the directories
#in str_list_listUserSpecifiedBackupSourceDirectoryFullPaths
#with the shortest final matching directory must be removed
#by reverse list item popping or removal;
#to properly handle the length change of
#str_list_listUserSpecifiedBackupSourceDirectoryFullPaths,
#while loop(s) must be used instead of for loop(s).
#
#also, when there is no shortest/earliest(use this?)
#final/shared[use this]/common[or use this] parent directory
#other than the drive, all the directories can be recorded in
#str_list_listShadowMountDirectoryFullPaths, and the
#shortest final parent directories finding operation can be existed
#since all the directories have no shared parent directory.
#actually, the above is not always true; later directories in
#the list can have shared parent directories.
```

```
    #in order to optimize the earliest shared parent directory finding
    #operation, the entire directory list should be sorted in
    #ascending order first, then iterated over
    #only once to build the shortest or earliest shared parent
    #directory list--build and update
    #str_list_listShadowMountDirectoryFullPaths as needed for
    #each directory in the sorted directory list, including
    #modifying an entry in str_list_listShadowMountDirectoryFullPaths,
    #with checking against only the latest or last entry in
    #str_list_listShadowMountDirectoryFullPaths to fully harness
    #the sortedness of the user-specified backup directory list:
    #this is the final algorithm I will implement,
    #since this is the most efficient and least computationally
    #costly.
```

[11/28/2020 6:41 AM CST] going from an exponential operation to a linear operation! great! so damn great!!! (UPDATE [11/28/2020 6:47 AM CST] the earliest or shortest shared or common parent directory finding algorithm itself is constant operation, even better than log operation) any way to make it a log operation? maybe not? one way I can think of right now is processing each user input and building str_list_listShadowMountDirectoryFullPaths as user-input is read—actually, using a dictionary, not a list (this will still be a linear operation, since reading the user input is a linear operation—the earliest or shortest shared or common parent directory finding algorithm itself is linear or constant). I'll think about it later, not now.

7.3.1.1. OLD CODE [11/28/2020 6:29 AM CST]

```
#-----------
#Stage 2-2-2
#Create str_list_listShadowMountDirectoryFullPaths.
#-----------

str_list_listShadowMountDirectoryFullPaths = []

iNumberOfUserSpecifiedBackupSourceDirectoryFullPaths = \
    len(str_list_listUserSpecifiedBackupSourceDirectoryFullPaths)

for iOuterListIndex in \
    range(0, iNumberOfUserSpecifiedBackupSourceDirectoryFullPaths):

    str_listOuterUserSpecifiedBackupSourceDirectoryFullPath = \
        str_list_listUserSpecifiedBackupSourceDirectoryFullPaths[ \
            iOuterListIndex]

    iNumberOfOuterUserSpecifiedBackupSourceDirectoryFullPathElements = \
        len(str_listOuterUserSpecifiedBackupSourceDirectoryFullPath)
```

```
if iNumberOfOuterUserSpecifiedBackupSourceDirectoryFullPathElements \
   == 1:
   if str_listOuterUserSpecifiedBackupSourceDirectoryFullPath not in \
      str_list_listShadowMountDirectoryFullPaths:
      str_list_listShadowMountDirectoryFullPaths.append( \
         str_listOuterUserSpecifiedBackupSourceDirectoryFullPath)
   continue

str_listFinalShortestMatchingDirectoryPath = \
   str_listOuterUserSpecifiedBackupSourceDirectoryFullPath
iNumberOfElementsInFinalShortestMatchingDirectoryPath = \
   len(str_listFinalShortestMatchingDirectoryPath)

for iInnerListIndex in \
   range(0, iNumberOfUserSpecifiedBackupSourceDirectoryFullPaths):

   if iOuterListIndex == iInnerListIndex: continue

   str_listInnerUserSpecifiedBackupSourceDirectoryFullPath = \
      str_list_listUserSpecifiedBackupSourceDirectoryFullPaths[ \
         iInnerListIndex]

   iNumberOfInnerUserSpecifiedBackupSourceDirectoryFullPathElements = \
      len(str_listInnerUserSpecifiedBackupSourceDirectoryFullPath)

   if iNumberOfOuterUserSpecifiedBackupSourceDirectoryFullPathElements \
      < iNumberOfInnerUserSpecifiedBackupSourceDirectoryFullPathElements:
      iSmallerNumberOfUserSpecifiedBackupSourceDirectoryFullPathElements \
         = iNumberOfOuterUserSpecifiedBackupSourceDirectoryFullPathElements
      #iLargerNumberOfUserSpecifiedBackupSourceDirectoryFullPathElements \
      #    = iNumberOfInnerUserSpecifiedBackupSourceDirectoryFullPathElements
   else:
      iSmallerNumberOfUserSpecifiedBackupSourceDirectoryFullPathElements \
         = iNumberOfInnerUserSpecifiedBackupSourceDirectoryFullPathElements
      #iLargerNumberOfUserSpecifiedBackupSourceDirectoryFullPathElements \
      #    = iNumberOfOuterUserSpecifiedBackupSourceDirectoryFullPathElements

   iNumberOfElementsInShortestMatchingDirectoryPath = 0

   for iZeroBasedDirectoryPathIndex in range( \
      0, iSmallerNumberOfUserSpecifiedBackupSourceDirectoryFullPathElements):
      if str_listOuterUserSpecifiedBackupSourceDirectoryFullPath[ \
         iZeroBasedDirectoryPathIndex] == \
         str_listInnerUserSpecifiedBackupSourceDirectoryFullPath[ \
         iZeroBasedDirectoryPathIndex]:
            iNumberOfElementsInShortestMatchingDirectoryPath += 1
        else:
            break

    if iNumberOfElementsInShortestMatchingDirectoryPath == 0:
        pass
    elif iNumberOfElementsInShortestMatchingDirectoryPath == 1:
        if iNumberOfOuterUserSpecifiedBackupSourceDirectoryFullPathElements \
            == 1 or \
            iNumberOfInnerUserSpecifiedBackupSourceDirectoryFullPathElements \
            == 1:
            str_listFinalShortestMatchingDirectoryPath = \
                str_listOuterUserSpecifiedBackupSourceDirectoryFullPath[ \
                0:1]
            #iNumberOfElementsInFinalShortestMatchingDirectoryPath \
            #    = 1

            if str_listFinalShortestMatchingDirectoryPath not in \
                str_list_listShadowMountDirectoryFullPaths:
                str_list_listShadowMountDirectoryFullPaths.append( \
                    str_listFinalShortestMatchingDirectoryPath)

            break
    else:
        if iNumberOfElementsInShortestMatchingDirectoryPath < \
            iNumberOfElementsInFinalShortestMatchingDirectoryPath:
            str_listFinalShortestMatchingDirectoryPath = \
                str_listOuterUserSpecifiedBackupSourceDirectoryFullPath[ \
                0:iNumberOfElementsInShortestMatchingDirectoryPath]
            iNumberOfElementsInFinalShortestMatchingDirectoryPath \
                = iNumberOfElementsInShortestMatchingDirectoryPath

if str_listFinalShortestMatchingDirectoryPath not in \
    str_list_listShadowMountDirectoryFullPaths:
    str_list_listShadowMountDirectoryFullPaths.append( \
        str_listFinalShortestMatchingDirectoryPath)
```

7.3.2. updated algorithm [10/24/2020 10:55 AM CST]

7.3.2.1. DRAFT 3 [10/24/2020 3:16 PM CST]

[10/24/2020 10:55 AM CST] going for the shortest matching directory path other than drive root path (unless the user specifies the drive root path).

(NOTE [10/24/2020 10:26 AM CST] each element of str_listShadowCopyMountDirectoryFullPath and str_listUserSpecifiedBackupSourceDirectoryFullPath contains each directory path element or each directory name, if the first element, the drive letter. e.g. 'c', 'dir1', 'sub-dir1', 'sub-dir2', etc.)

(consider the following cases.)
c:\users\allen\documents\folder1\folder1-2\
c:\users\allen\documents\folder1\folder1-4\
c:\users\allen\documents\folder2\
c:\users\allen\documents\folder2\folder2-1\
c:\users\allen\documents\folder2\folder2-1-1\
c:\users\allen\documents\folder3\
c:\users\allen\AAM_posting
c:\temp\test\
c:\temp\test2\

in the first stage, create self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths using self._str_listUserSpecifiedBackupSourceDirectoryFullPaths.

in the second stage, in one double-layer for-loop, for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, compare to all the other elements (i.e. except itself) in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, and log the shortest matching directory path (other than the drive root path unless the user specified the drive root path) in self._str_list_listShadowCopyMountDirectoryFullPaths, if the match is not already present in self._str_list_listShadowCopyMountDirectoryFullPaths. if no match (other than the drive root path unless the user specified the drive root path), record the full path in self._str_list_listShadowCopyMountDirectoryFullPaths.

then, in the third stage, convert self._str_list_listShadowCopyMountDirectoryFullPaths to self._str_listShadowCopyMountDirectoryFullPaths.

FINAL DECISION [10/24/2020 3:42 PM CST] I'm implementing the above. the above algorithm is exactly what I want!

7.3.2.2. DRAFT 2 [10/24/2020 11:10 AM CST]

[10/24/2020 10:55 AM CST] going for the shortest matching directory path other than drive root path (unless the user specifies the drive root path).

(NOTE [10/24/2020 10:26 AM CST] each element of str_listShadowCopyMountDirectoryFullPath and str_listUserSpecifiedBackupSourceDirectoryFullPath contains each directory path element or each directory name, if the first element, the drive letter. e.g. 'c', 'dir1', 'sub-dir1', 'sub-dir2', etc.)

(consider the following cases.)
c:\users\allen\documents\folder1\folder1-2\
c:\users\allen\documents\folder1\folder1-4\
c:\users\allen\documents\folder2\
c:\users\allen\documents\folder2\folder2-1\
c:\users\allen\documents\folder2\folder2-1-1\
c:\users\allen\documents\folder3\
c:\users\allen\AAM_posting
c:\temp\test\
c:\temp\test2\

in the first stage, in one double-layer for-loop, for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, compare to all the other elements (i.e. except itself) in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, and log the shortest matching directory path (other than the drive root path unless the user specified the drive root path) in self._str_list_listShadowCopyMountDirectoryFullPaths, if the match is not already present in self._str_list_listShadowCopyMountDirectoryFullPaths. if no match (other than the drive root path unless the user specified the drive root path), record the full path in self._str_list_listShadowCopyMountDirectoryFullPaths.

then, in the second stage, process each element of
self._str_list_listShadowCopyMountDirectoryFullPaths
to eliminate every child path of a path in
self._str_list_listShadowCopyMountDirectoryFullPaths. use while loop to handle the change in the length of self._str_list_listShadowCopyMountDirectoryFullPaths.

then, in the third stage, convert self._str_list_listShadowCopyMountDirectoryFullPaths to self._str_listShadowCopyMountDirectoryFullPaths.

7.3.2.3. DRAFT 1

[10/24/2020 10:55 AM CST] going for the shortest matching directory path other than drive root path (unless the user specifies the drive root path).

(NOTE [10/24/2020 10:26 AM CST] each element of
str_listShadowMountCandidateDirectoryPath and
str_listUserSpecifiedBackupSourceDirectoryFullPath contains each directory path element or each directory name, or if the first element, the drive letter. e.g. 'c', 'dir1', 'sub-dir1', 'sub-dir2', etc.)

(consider the following cases.)
c:\users\allen\documents\folder1\folder1-2\

c:\users\allen\documents\folder1\folder1-4\
c:\users\allen\documents\folder2\
c:\users\allen\documents\folder2\folder2-1\ (this will be eliminated beforehand by this program)
c:\users\allen\documents\folder2\folder2-1-1\ (this will be eliminated beforehand by this program)
#c:\users\allen\documents\folder3\
c:\temp\test\
c:\temp\test2\
yeah, I think the above can definitely be optimized.

in the first stage, in one double-layer for-loop, for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, compare to all the elements below in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, and log the shortest matching directory path (other than the drive root path unless the user specified the drive root path) in self._str_list_listShadowMountCandidateDirectoryPaths, if the match is not already present in self._str_list_listShadowMountCandidateDirectoryPaths. if no match (other than the drive root path unless the user specified the drive root path), record the full path in self._str_list_listShadowMountCandidateDirectoryPaths.

then, in the second stage, for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, find the longest matching parent directory in self._str_list_listShadowMountCandidateDirectoryPaths, and record it in self._str_listShadowCopyMountDirectoryFullPaths if it is not already present in self._str_listShadowCopyMountDirectoryFullPaths.

then, in the third stage, process each element of
self._str_listShadowCopyMountDirectoryFullPaths
to eliminate every child path of a path in self._str_listShadowCopyMountDirectoryFullPaths.
use while loop to handle the change in the length of
self._str_listShadowCopyMountDirectoryFullPaths.

7.3.3. the latest algorithm idea from the original notes [10/24/2020 10:14 AM CST]

(NOTE [10/24/2020 10:26 AM CST] each element of str_listMaximumMatchingDirectoryPath and str_listUserSpecifiedBackupSourceDirectoryFullPath contains each directory path element or each directory name, or if the first element, the drive letter. e.g. 'c', 'dir1', 'sub-dir1', 'sub-dir2', etc.)

use the shorter number of path elements to compare. when there is no more match, quit the double loop.

process the minimized self._str_list_listMaximumMatchingDirectoryPaths.
for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, find the longest matching parent directory in self._str_list_listMaximumMatchingDirectoryPaths, and record it in self._str_listDirectoriesToShadowCopyMount.
(how about the above? I think it is perfect.)

(maybe the above can be merged into one double-layer for-loop.)

(consider the following cases.)
c:\users\allen\documents\folder1\folder1-2\
c:\users\allen\documents\folder1\folder1-4\
c:\users\allen\documents\folder2\
c:\users\allen\documents\folder2\folder2-1\ (this will be eliminated beforehand by this program)
c:\users\allen\documents\folder2\folder2-1-1\ (this will be eliminated beforehand by this program)
#c:\users\allen\documents\folder3\
c:\temp\test\
c:\temp\test2\
yeah, I think the above can definitely be optimized.

in the first stage, in one double-layer for-loop, for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, compare to all the elements below in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, and log each longest matching directory path in self._str_list_listMaximumMatchingDirectoryPaths, if the match is not already present in self._str_list_listMaximumMatchingDirectoryPaths. if no match, record the full path in self._str_list_listMaximumMatchingDirectoryPaths.

then, in the second stage, for each entry in self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths, find the longest matching parent directory in self._str_list_listMaximumMatchingDirectoryPaths, and record it in self._str_listDirectoriesToShadowCopyMount if it is not already present in self._str_listDirectoriesToShadowCopyMount.

then, in the third stage, process each element of self._str_listDirectoriesToShadowCopyMount to eliminate every child path of a path in self._str_listDirectoriesToShadowCopyMount. use while loop to handle the change in the length of self._str_listDirectoriesToShadowCopyMount.

(I think the above will do perfectly.)
(nope, the above won't do. If I don't force not including the entire drive [root directory], it wouldn't be nicely done.)
(however, using the above wouldn't be too bad, although

it may result in many shadow copy mounts.)
(UPDATE [10/18/2020 11:30 PM CST] ok, I think the above is absolutely perfect now. I'll review it tomorrow.)

[10/24/2020 10:52 AM CST] the above algorithm would not shadow-mount c:\users\allen\documents, and result in too many shadow mounting. I think I should go for the shortest matching directory path other than drive root path (unless the user specifies the drive root path).

7.3.4. ORIGINAL NOTES [10/18/2020 10:29 PM CST]

[10/21/2020 12:55 PM CST]

```
    ##########################################################################
    #Stage 3
    #-------
    #AAM Auto Backup instruction plain text file processing
    ##########################################################################
    open() #?

    #self._str_listUserSpecifiedDirectoriesToBackup = []
    self._str_listUserSpecifiedBackupSourceDirectoryFullPaths  = []
       #all lower case when the OS is Windows.

    #self._str_listUserSpecifiedBackupSourceDirectoryFullPaths
    #should have no child directory of a parent directory
    #self._str_listUserSpecifiedBackupSourceDirectoryFullPaths.
    #process each element in
    #self._str_listUserSpecifiedBackupSourceDirectoryFullPaths,
    #and eliminate every child directory.
    #use a while loop to handle the
    #self._str_listUserSpecifiedBackupSourceDirectoryFullPaths
    #length change.

    #--------------------------------------------------------------------
    #Stage 3-x
    #In self._str_listUserSpecifiedBackupSourceDirectoryFullPaths,
    #check the existence of each user-specified directory:
    #Remove the ones that do not exist.
    #--------------------------------------------------------------------



    #--------------------------------------------------------------------
    #Stage 3-x
    #Create self._str_listDirectoriesToShadowCopyMount.
    #--------------------------------------------------------------------
    iNumberOfUserSpecifiedBackupSourceDirectoryFullPaths = \
       len(self._str_listUserSpecifiedBackupSourceDirectoryFullPaths)

    self._str_list_listDirectoriesToShadowCopyMount = []
    self._str_listDirectoriesToShadowCopyMount = []

    #[10/18/2020 9:32 PM CST]
    #how to factor the most common directories?  use
    #path separator data variable and a dictionary?  very likely so?
    #how about comparing each user-specified directory to all the rest,
    #then build self.str_listDirectoriesToShadowCopyMount in the first
    #iteration, then process self.str_listDirectoriesToShadowCopyMount
    #again to reduce self.str_listDirectoriesToShadowCopyMount
    #to the longest directory full paths only?

    #UPDATE [10/18/2020 9:55 PM CST]
    #I think the proper strategy is to group the user-specified
    #backup-source directory full paths.
    for iOuterIndex in \
       range(0, iNumberOfUserSpecifiedBackupSourceDirectoryFullPaths):

       strOuterUserSpecifiedBackupSourceDirectoryFullPath = \
           self._str_listUserSpecifiedBackupSourceDirectoryFullPaths[ \
               iOuterIndex]
       str_listOuterUserSpecifiedBackupSourceDirectoryFullPath = \
           strOuterUserSpecifiedBackupSourceDirectoryFullPath.split( \
               self._PathSeparator)
       iNumberOfElementsInOuterUserSpecifiedBackupSourceDirectoryFullPath \
           = len(str_listOuterUserSpecifiedBackupSourceDirectoryFullPath)

       for iInnerIndex in \
           range(0, iNumberOfUserSpecifiedBackupSourceDirectoryFullPaths):

           strInnerUserSpecifiedBackupSourceDirectoryFullPath = \
               self._str_listUserSpecifiedBackupSourceDirectoryFullPaths[ \
                   iInnerIndex]
           str_listInnerUserSpecifiedBackupSourceDirectoryFullPath = \
               strInnerUserSpecifiedBackupSourceDirectoryFullPath.split( \
                   self._PathSeparator)
           iNumberOfElementsInInnerUserSpecifiedBackupSourceDirectoryFullPath \
               = len(str_listInnerUserSpecifiedBackupSourceDirectoryFullPath)

           if iNumberOfElementsInOuterUserSpecifiedBackupSourceDirectoryFullPath \
               < iNumberOfElementsInInnerUserSpecifiedBackupSourceDirectoryFullPath:
               iLessNumberOfElements = \
                   iNumberOfElementsInOuterUserSpecifiedBackupSourceDirectoryFullPath
           else:
```

```
        iLessNumberOfElements = \
          iNumberOfElementsInInnerUserSpecifiedBackupSourceDirectoryFullPath

      iLastMatchIndex = -1

      for iPathElementIndex in range(0, iLessNumberOfElements):
        if str_listOuterUserSpecifiedBackupSourceDirectoryFullPath[ \
          iPathElementIndex] == \
          str_listInnerUserSpecifiedBackupSourceDirectoryFullPath[ \
          iPathElementIndex]:
          iLastMatchIndex += 1
        else:
          break

      self._str_list_listDirectoriesToShadowCopyMount.append( \
        str_listOuterUserSpecifiedBackupSourceDirectoryFullPath[ \
        0:iLastMatchIndex + 1])

#[10/18/2020 10:21 PM CST]
#what would be the best way to implement the above?
#I need to come up with an answer.  I will come up with an answer!
#use the longest non-drive root directory path for shadow-copy mounting?

#AN IDEA [10/18/2020 10:25 PM CST]
#how about a removal tactic?
#create an intermediate self._str_listUserSpecifiedBackupSourceDirectoryFullPaths.
#nested for loops, two layers.
#in each list element of the above list, ...
#(no, no, no!)

#[10/18/2020 10:29 PM CST]
#copy self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths
#to self._str_list_listMaximumMatchingDirectoryPaths (an intermediate output
#data variable).
#then, iteratively process self._str_list_listMaximumMatchingDirectoryPaths
#until there is no more match.
#for each user-specified backup source directory full path,
#find the maximum or longest consecutive path element match, and append
#that match to self._str_list_listMaximumMatchingDirectoryPaths
#if the match
#is not already present self._str_list_listMaximumMatchingDirectoryPaths.
#use the shorter number of path elements to compare.
#when there is no more match, quit the double loop.
#(ok, the above seems to hold much promise.)
#
#process self._str_list_listMaximumMatchingDirectoryPaths
```

```
#to remove every the path that is either not present in
#self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths
#in the full form (or path), or do not have more than one
#child directory included in
#self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths.
#simply convert self._str_list_listMaximumMatchingDirectoryPaths into
#self._str_listDirectoriesToShadowCopyMount.
#(wait, the above still does not eliminate c:.  I need to rework
#the above.  how about the following?)
#--
#process the minimized self._str_list_listMaximumMatchingDirectoryPaths.
#for each entry in
#self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths,
#find the longest matching parent directory in
#self._str_list_listMaximumMatchingDirectoryPaths, and record it
#in self._str_listDirectoriesToShadowCopyMount.
#(how about the above?  I think it is perfect.)
#(maybe the above can be merged into one double-layer for-loop.)
#(consider the following cases.)
#c:\users\allen\documents\folder1\folder1-2\
#c:\users\allen\documents\folder1\folder1-4\
#c:\users\allen\documents\folder2\
#c:\users\allen\documents\folder2\folder2-1\ (this will be eliminated beforehand by this program)
#c:\users\allen\documents\folder2\folder2-1-1\ (this will be eliminated beforehand by this program)
##c:\users\allen\documents\folder3\
#c:\temp\test\
#c:\temp\test2\
#yeah, I think the above can definitely be optimized.
#in the first stage, in one double-layer for-loop,
#for each entry in
#self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths,
#compare to all the elements below in
#self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths,
#and log each longest matching directory path
#in self._str_list_listMaximumMatchingDirectoryPaths,
#if the match is not already present in
#self._str_list_listMaximumMatchingDirectoryPaths.
#if no match, record the full path in
#self._str_list_listMaximumMatchingDirectoryPaths.
#then, in the second stage,
#for each entry in
#self._str_list_listUserSpecifiedBackupSourceDirectoryFullPaths,
#find the longest matching parent directory in
#self._str_list_listMaximumMatchingDirectoryPaths, and record it
```

```
#in self._str_listDirectoriesToShadowCopyMount.
#then, in the third stage, process each element of
#self._str_listDirectoriesToShadowCopyMount
#to eliminate every child path of a path in
#self._str_listDirectoriesToShadowCopyMount.
#use while loop to handle the change in the length of
#self._str_listDirectoriesToShadowCopyMount.
#(I think the above will do perfectly.)
#(nope, the above won't do.  If I don't force not including the
#entire drive [root directory], it wouldn't be nicely done.)
#(however, using the above wouldn't be too bad, although
#it may result in many shadow copy mounts.)
#(UPDATE [10/18/2020 11:30 PM CST] ok, I think the above is absolutely
#perfect now.  I'll review it tomorrow.)
```

7.4. exclusion child directories computation algorithm

exclusion child directory listing algorithm

(NOTE [10/21/2020 4:41 PM CST] get the folders to exclude absolutely right; Robocopy must exclude every folder in the shadow-copy mounted directory, that should not be copied; Robocopy must not exclude folders in copying that must not be excluded.)

[10/21/2020 4:47 PM CST]

[10/21/2020 5:06 PM CST] maybe process each child directory level (1st level, 2nd level, and so on) to compute all the child directories to exclude?

7.4.1. update [12/2/2020 1:58 PM CST]

```
############################################################
#Build self._str_listShadowCopyNonIncludedFolderFullPaths.
############################################################
self._str_listShadowCopyNonIncludedFolderFullPaths = []

str_listParentDirectoryFullPathsToProcessToFindExclusion = \
   [self._aam_abl.strShadowCopyDriveName + ':']
str_listNextLevelDirectoryFullPathsToProcessToFindExclusion = []

try:
   #(update the following code to build
   #self._str_listShadowCopyNonIncludedFolderFullPaths)

   #TO DO [12/1/2020 4:06 PM CST]
```

```
#code update done below.
#delete the code below that should be removed.
#then, proofread and verify the below code.

while str_listParentDirectoryFullPathsToProcessToFindExclusion != []:
   #build str_listNextLevelDirectoryFullPathsToProcessToFindExclusion
   #using os.listdir() and os.path.isdir()
   for strParentDirectoryFullPath in \
      str_listParentDirectoryFullPathsToProcessToFindExclusion:
      for vDirectoryItem in os.listdir(strParentDirectoryFullPath):
         if not os.path.isdir(vDirectoryItem): continue

         if platform.system() == 'Windows':
            vDirectoryItem = vDirectoryItem.lower()


         #[10/25/2020 10:15 PM CST] [12/1/2020 2:36 PM CST]
         #How should I check vDirectoryItem against
         #self._str_listShadowCopyInclusionFolderFullPaths
         #to determine whether it should be excluded or not?
         #a separate function or code here?
         #(code, no separate function.)
         #
         #the below will not do.
         #if vDirectoryItem not in \
         #    self._str_listShadowCopyInclusionFolderFullPaths:
         #    xx
         #
         #the following will work.
         #(what does the following do?  I need to express that.
         #I will express that!  [12/1/2020 2:44 PM CST])
         #([12/1/2020 3:25 PM CST] the following doesn't feel
         #right.  I think it should be very elegant, not
         #convoluted.)
         #The following updates
         #self._str_listShadowCopyNonIncludedFolderFullPaths and
         #str_listNextLevelDirectoryFullPathsToProcessToFindExclusion.
         #
         bExcludedDirectory = True
         bShadowCopyInclusionFolder = False
         for strShadowCopyInclusionFolderFullPath in \
            self._str_listShadowCopyInclusionFolderFullPaths:

            if vDirectoryItem == \
               strShadowCopyInclusionFolderFullPath:
               bExcludedDirectory = False
               bShadowCopyInclusionFolder = True
               break

            iDirectoryItemLength = len(vDirectoryItem)
            if vDirectoryItem == \
               strShadowCopyInclusionFolderFullPath[ \
               0:iDirectoryItemLength]:
               bExcludedDirectory = False
               #No break here, since vDirectoryItem can be
               #strShadowCopyInclusionFolderFullPath detected
               #later in this for loop.  or can it be?
               #if self._str_listShadowCopyInclusionFolderFullPaths
               #is sorted in descending order, it can't.
               #check that to finalize this code.
               #[12/1/2020 4:32 PM CST]
               #should I sort
               #self._str_listShadowCopyInclusionFolderFullPaths
               #in descending order, beforehand?
               #think and decide on that, and then implement
               #the decision!
               #[12/1/2020 4:38 PM CST]
               #it is in descending order, because
               #_ExecuteBackup() creates it that way!
               #fortuitous!  no, break can be done here!
               #no chance of vDirectoryItem later being
               #strShadowCopyInclusionFolderFullPath
               #in this for loop!
               #also, self._str_listShadowCopyInclusionFolderFullPaths
               #does not contain both a parent and a child
               #directory.  so, if a match happens here,
               #a later match with a child directory
               #in self._str_listShadowCopyInclusionFolderFullPaths
               #will not happen in this for loop.
               #[12/1/2020 4:45 PM CST]
               #I need to think this more.
               #vDirectoryItem = 'c:\\test'.
               #a strShadowCopyInclusionFolderFullPath is
               #'c:\\test\\subdir1'.  there are other
               #sub-directories in 'c:\\test'.
               #vDirectoryItem should be logged in
               #str_listNextLevelDirectoryFullPathsToProcessToFindExclusion.
               #it does that below!  it is right, as of now!
               break

         if bExcludedDirectory:
            self._str_listShadowCopyNonIncludedFolderFullPaths.append( \
                    vDirectoryItem)
                elif not bShadowCopyInclusionFolder: #bExcludedDirectory == False
                    str_listNextLevelDirectoryFullPathsToProcessToFindExclusion.append( \
                        vDirectoryItem)

            str_listParentDirectoryFullPathsToProcessToFindExclusion = \
                str_listNextLevelDirectoryFullPathsToProcessToFindExclusion

            str_listNextLevelDirectoryFullPathsToProcessToFindExclusion = []

    except Exception as exc:
        sys.exit('File I/O error while computing \
str_listShadowCopyNonIncludedFolderFullPaths.\n' + str(exc))
```

7.4.2. algorithm brainstorm 1 [10/24/2020 8:45 PM CST]

```
    self._str_dictShadowMountDirectoryFullPathsToExclusionDirectoryLists = {}
        #absolute or relative exclusion directory full paths?
        #perform Robocopy test(s), and find out!
        #UPDATE [10/22/2020 10:47 PM CST] very likely absolute
        #without the shadow-mount drive letter.
        #actually, with the shadow-mount drive letter.
        #
        #UPDATE [10/24/2020 3:57 PM CST]
        #the exclusion directory in this data variable can be an absolute
        #path starting with the shadow-mount drive letter,
        #a relative path, or just a directory name.  all exclusion
        #directories in this data variable will be used in the robocopy
```

[10/24/2020 8:58 PM CST] ok, this feels like an even more complex coding problem to solve than the shadow-copy mount directories computation.

I've solved the shadow-copy mount directories computation software coding problem to my perfect liking.

I will solve this 'exclusion child directories computation' software coding problem to my perfect liking also!

[10/24/2020 9:06 PM CST] very likely the following algorithm outline will be used.
~~first stage is~~ processing each element in self._str_listShadowMountDirectoryFullPaths, and computing every sub-directory to exclude using self._str_listUserSpecifiedBackupSourceDirectoryFullPaths. perhaps process each sub-directory level? the applicable deepest sub-directory level in

self._str_listUserSpecifiedBackupSourceDirectoryFullPaths decides how many sub-directory levels must be processed for sub-directory (path) exclusion in the Robocopy command and the 7z.exe list-file generation.
(NOTE [10/24/2020 9:32 PM CST] yeah, the above algorithm of processing each sub-directory level is in all likely chance what I will do.)
(UPDATE [10/24/2020 10:26 PM CST] str_list_listShadowMountDirectoryFullPaths should be used, instead of self._str_listShadowMountDirectoryFullPaths, for the number of directory elements.) (QUESTION TO ANSWER [10/24/2020 10:36 PM CST] any better way? I prefer the least coding [complication].)
(QUESTION TO ANSWER [10/24/2020 11:02 PM CST] do I want a separate function for getting the longest matching parent directory? will that make my coding easier? very likely.)
(QUESTION TO ANSWER [10/24/2020 11:06 PM CST] use os.listdir() and os.path.isdir(), or use the Python walk function to recursively scan each directory to backup in the shadow copy drive?)
PROCESSING TECHNIQUE TO USE [10/24/2020 11:12 PM CST] at each sub-directory level to process, build a list to keep the sub-directories to keep and further process. (what to name this list though? [10/25/2020 8:26 PM CST])
QUESTION TO ANSWER [10/24/2020 11:18 PM CST] do I want some utility function(s) for doing this computation or not? I probably do.

UPDATE [10/25/2020 9:02 PM CST] maybe I should use a dictionary, or multi-level list, for processing every sub-directory level? str_multilevel_listSubDirectoriesToProcess? maybe recursion (using a recursive function) will solve the problem neatly?
[10/25/2020 9:12 PM CST] actually, how about this? build str_listCurrentLevelSubDirectoriesToProcess using os.listdir() and os.path.isdir(). in a loop, process each sub-directory in str_listCurrentLevelSubDirectoriesToProcess to build str_listNextLevelSubDirectoriesToProcess. When processing str_listCurrentLevelSubDirectoriesToProcess is over, assign str_listNextLevelSubDirectoriesToProcess to str_listCurrentLevelSubDirectoriesToProcess, set str_listNextLevelSubDirectoriesToProcess to the empty list ([]), and process str_listCurrentLevelSubDirectoriesToProcess again for the next-level sub-directories to process. repeat the process until there is no more sub-directory to process. I think this algorithm holds a great promise. if the structure of the user-specified directories are not too complex, this processing of finding the directories to exclude will end rather quickly.
[10/25/2020 9:35 PM CST] to be more precise, use str_listCurrentLevelSubDirectoriesToProcessToFindExclusion and str_listNextLevelSubDirectoriesToProcessToFindExclusion. NOTE [10/25/2020 9:37 PM CST] I don't need to find and use iNumberOfSubDirectoryLevelsToProcess, when I'm using these data variables!!!!
[10/25/2020 10:02 PM CST] actually, use str_listCurrentLevelDirectoriesToProcessToFindExclusion and str_listNextLevelDirectoriesToProcessToFindExclusion for the algorithm implementation issue.
[10/25/2020 10:06 PM CST] use str_listParentDirectoriesToProcessToFindExclusion instead of str_listCurrentLevelDirectoriesToProcessToFindExclusion.

UPDATE [10/25/2020 10:35 PM CST] ok, drafting the code is done. rather a small amount of code, unlike what I expected! see, once again, I can never know what will happen until I actually do it!!!!!

~~second stage is excluding every directory specified by the user, in self._str_listUserSpecifiedDirectoriesToExclude.~~
~~(UPDATE [10/24/2020 10:46 PM CST] I don't think this stage is required. if a full path in self._str_listUserSpecifiedDirectoriesToExclude is not in self._str_dictShadowMountDirectoryFullPathsToExclusionDirectoryLists, add that to the excluded directory list. yeah, this is definitely not required.)~~

7.4.3. DEPRECATED CODE DRAFT [10/25/2020 9:43 PM CST]

```
#--------------------------------------------------------------------
#Stage 2-3
#Shadow-copy mount directories exclusion child directories computation.
#Create self._str_dictShadowMountDirectoryFullPathsToExclusionDirectoryLists.
#--------------------------------------------------------------------

for str_listShadowMountDirectoryFullPath in \
   str_list_listShadowMountDirectoryFullPaths:

   #Calculate the number of sub-directory levels to process, and
   #store the sub-directories of this
   #str_listShadowMountDirectoryFullPath to include in backup
   #(i.e. not exclude).
   iNumberOfSubDirectoryLevelsToProcess = 0
   for str_listUserSpecifiedBackupSourceDirectoryFullPath in \
      str_list_listUserSpecifiedBackupSourceDirectoryFullPaths:
      pass

   iStartingSubDirectoryLevelToProcess = 0
   iEndingSubDirectoryLevelToProcess = 0

   str_listSubDirectoriesToExcludeForThisShadowMountDirectory = []

   str_listCurrentLevelSubDirectoriesToProcessToFindExclusion = []
   str_listNextLevelSubDirectoriesToProcessToFindExclusion = []

   #Process each sub-directory level, and add every excluded
   #sub-directory in each sub-directory level (as the shadow-mount path).
   for iSubDirectoryLevelToProcess in range( \
      iStartingSubDirectoryLevelToProcess,
      iEndingSubDirectoryLevelToProcess):
```

```
		#at each sub-directory level to process, build a list to keep the sub-directories to keep
and further process.
		str_listDirectoriesToProcessAtThisLevel = []

		strParentDirectoryFullPath = ''

		#...
```

7.5. file exclusion computation

[12/2/2020 2:47 PM CST]

```
        #Build self._str_listShadowCopyExclusionFileRegularExpressions and
        #self._regexp_obj_listShadowCopyExclusionFiles.
        #([11/23/2020 7:04 AM CST] is this necessary? 7z.exe list file
        #probably supports wild cards?  look into that.
        #according to the "TEST 1 [11/5/2020 9:03 AM CST]"
        #in the AAM Auto Backup document, 7z.exe list files do support
        #wild cards.  using the regular expression may not be necessary,
        #even for incremental backup.
        #decide on that, and then implement the decision!)
        #([11/23/2020 7:39 AM CST] I'm beginning to think I may not need
        #to use regular expressions for incremental backup.  actually, I do,
        #for omitting files especially.  how about building inclusion list file,
        #and also using exclusion list file[s] when doing incremental backup?
        #nah, I need to build the inclusion list file with specific file path,
        #due to the modification and creation time check.  no need to
        #create exclusion list file[s] when creating a perfect
        #inclusion list file with no unnecessary inclusions is cleanest.
        #or is it?  for now, let's go with creating minimum list file;
        #I like the minimum policy.)
        #(CONCLUSION [11/23/2020 7:51 AM CST] for counting the number of
        #files to incrementally backup, I definitely need to use the regular
        #expressions.) (ADDITION [11/23/2020 8:13 AM CST] file exclusion check
        #using regular expressions is also needed for validating the Robocopy
        #results.)
        for strShadowCopyExclusionFile in \
            self._str_listShadowCopyExclusionFiles:

            strShadowCopyExclusionFileRegularExpression = ''

            for strShadowCopyExclusionFileCharacter in \
                strShadowCopyExclusionFile:

                if strShadowCopyExclusionFileCharacter == '?':
                    strShadowCopyExclusionFileRegularExpression += '.'
                elif strShadowCopyExclusionFileCharacter == '*':
```

```
        strShadowCopyExclusionFileRegularExpression += '*?'
    elif strShadowCopyExclusionFileCharacter == '.':
        strShadowCopyExclusionFileRegularExpression += '\\.'
    else:
        strShadowCopyExclusionFileRegularExpression += \
            strShadowCopyExclusionFileCharacter

regexp_objShadowCopyExclusionFile = re.compile( \
    strShadowCopyExclusionFileRegularExpression, re.IGNORECASE)

self._str_listShadowCopyExclusionFileRegularExpressions.append( \
    strShadowCopyExclusionFileRegularExpression)
self._regexp_obj_listShadowCopyExclusionFiles.append( \
    regexp_objShadowCopyExclusionFile)
```

8. backup AWS S3 bucket query processing

[11/9/2020 9:56 AM CST]
Update the general AWS S3 query function.
No value returned.
Raise error if AWS S3 query error.
If no valid full backup, set the full-backup date time strings to empty strings.
[11/9/2020 9:54 AM CST] no error raising. Nothing is returned in the case of error.

8.1. backup AWS S3 bucket query algorithm

UPDATE [10/30/2020 11:18 AM CST]
question to answer and implement the answer of.
how to code the queried backup AWS S3 object names processing?
first, I think I should decide on the output format.
I think I should build a list of backup AWS S3 object name list for each date time, named str_list_listDateTimeSortedAndGroupedBackupAwsS3ObjectNames, that excludes all the backup AWS S3 object names with non-applicable unique backup-source location names.

[10/30/2020 11:54 AM CST] wait a minute. I need to first resolve the issue of a backup source unique name in the backup AWS S3 object name containing all or part of "incremental-backup_(date in YYYYMMDD)_(time in HHMMSS.SSSSSS, 24 hour time format)". I think preventing including "incremental-backup " in code is the best way to go. (not needed now, for my own use. needed later in . actually, a simple solution is making the backup AWS S3 object name uniform. I'll do that now.

UPDATE [10/28/2020 6:53 PM CST] how about using the content in the "AAM Auto Backup data query rules" document sector? for now, I don't see anything to change or add to what's in the "AAM Auto Backup data query rules" document sector.
[10/28/2020 6:58 PM CST] so, three AWS S3 query functions?
_QueryBackupAwsS3BucketForRegularBackup(),
_QueryBackupAwsS3BucketForIncrementalOnlyBackup(), and
_QueryBackupAwsS3BucketForDataRestoration()? very likely so?

[10/27/2020 1:45 PM CST]
1.   ...

in fact, in order to properly handle full backup upload failure, I must implement handling more than one full backup in the same month in the backup AWS S3 bucket. so, I will do so.
[10/22/2020 8:02 PM CST]

if I implement more than two full backups in a month, I will have to use regular expressions, lists, and dictionaries to process everything related properly. [10/22/2020 8:02 PM CST]

TO ANSWER [10/26/2020 7:30 AM CST] should I implement the AWS S3 bucket query in a separate function, because the query should be executed only when AWS S3 backup is on? very likely. think and answer that, and then implement the answer!
UPDATE [10/26/2020 8:11 AM CST] in all likely chance, I will implement the AWS S3 bucket query in a separate function, because it makes perfect sense. what should I name the function? _QueryBackupAwsS3Bucket()? most likely so? very likely so.

NOTE [10/26/2020 10:06 AM CST] _QueryBackupAwsS3Bucket() retrieves the AWS S3 full backup with all of the backup sources or shadow-mounted directories fully uploaded. AAM Auto Backup does not consider a full backup with an incomplete-upload backup source as a complete-upload full backup.

8.2. CONSIDERATIONS

[10/22/2020 8:05 PM CST]

(TO DO [10/26/2020 7:53 PM CST] put the following in the appropriate places in "AAM Auto Backup Operation Modes and Command Formats" part.)
resolve the issue of the content of the AAM Auto Backup instruction plain text file changing. what should I do with incomplete full backup AWS S3 objects, with incomplete uploads and/or missing backup sources, or additional backup sources? do I delete the backup AWS S3 objects with incomplete uploads and/or irrelevant or outdated backup sources? no, I can't delete with the upload-only account for cybersecurity reasons. I'm leaning toward creating additional full-backup in the same month as needed, when the content of the AAM Auto Backup instruction plain text file changes (i.e. the shadow-mounted directories do not match what is in the backup AWS S3 bucket); also, when doing incremental-only backup, all the backup sources or shadow-

mounted directories in the AAM Auto Backup instruction plain text file will have to be checked in the backup AWS S3 bucket to decide that all the complete-upload full backups exist in the backup AWS S3 bucket. think and decide on this, and then implement the decision. document all this. also, the AWS S3 backup object name formats must be updated. the current ones aren't conducive to doing easy backup AWS S3 bucket querying. [10/22/2020 7:48 PM CST] if I implement more than two full backups in a month, I will have to use regular expressions, lists, and dictionaries to process everything related properly. in fact, in order to properly handle full backup upload failure, I must implement handling more than one full backup in the same month in the backup AWS S3 bucket. so, I will do so. [10/22/2020 8:02 PM CST]

TO ADDRESS [10/26/2020 7:39 AM CST] do I need to provide a 'force full backup on AWS' capability to address the above issue? to enable the user to manually perform full backup on AWS after changing the AAM Auto Backup instruction plain text file content? if I do provide that option (in all likely chance I probably will), the implementation method probably will be providing an additional argument.
TO DECIDE [10/26/2020 8:02 PM CST] do 'force full backup on AWS', check the modification date and time on the AAM Auto Backup instruction plain text file to auto-decide doing full backup, or do both? I think the second option of auto-deciding is really bad and unreliable, so I won't do it. well, maybe I will. I think I'll support both. maybe support another command argument, 'full backup when AAM Auto Backup instruction plain text file is modified after the last backup date and time'. note that an AWS S3 full backup is required, because new sub-directories within the same shadow-mounted directory can contain files that are older than the last backup date and time, which would prevent incremental backup of the new files in the new sub-directories.

UPDATE [10/26/2020 7:55 AM CST] I'm leaning toward implementing multiple AWS full backup in the same month, and force full backup on AWS.

NOTE, REMINDER [10/26/2020 10:04 AM CST] if the latest complete-upload AWS S3 full backup does not contain all the backup sources or shadow-mounted directories (and their contents), AAM Auto Backup performs a full backup.

TO ANSWER [10/26/2020 7:30 AM CST]
should I implement the AWS S3 bucket query in a separate function, because the query should be executed only when AWS S3 backup is on? very likely. think and answer that, and then implement the answer!
UPDATE [10/26/2020 8:11 AM CST] in all likely chance, I will implement the AWS S3 bucket query in a separate function, because it makes perfect sense. what should I name the function? _QueryBackupAwsS3Bucket()? most likely so? very likely so.

NOTE [10/26/2020 10:06 AM CST] _QueryBackupAwsS3Bucket() retrieves the AWS S3 full backup with all of the backup sources or shadow-mounted directories fully uploaded. AAM Auto Backup does not consider a full backup with an incomplete-upload backup source as a complete-upload full backup.

TO CONSIDER [10/26/2020 10:08 AM CST] if 7z.exe can append to existing archive files, and multiple source directory issue can be handled somehow (so that there is source directory separation in the archived files), creating only one set of archive files, instead of multiple, is an option. I need to test 7z.exe commands to find out for sure. is there a way to add directory in the archived file relative paths?
NOTE [10/26/2020 10:24 AM CST] finding and trying some other program, such as GNU ZIP on Windows, is also an option for appending to existing archive files, and providing relative path head or root in the archived files.

TO CONSIDER [10/26/2020 12:51 PM CST] how about running the 7z command on the backup destination files after executing all the robocopy commands? the file modified time can be used for incremental backup. no need to include the absolute path (only relative), and no need to append to an existing archive file(s). the only risk is some the files in the backup destination being in use. consider this option also.
[10/26/2020 12:58 PM CST] -ssw may have to be used if running the 7z command on the backup destination files.

consider running the 7z command on the backup destination files after executing all the robocopy commands. [10/26/2020 12:57 PM CST] decide on the final backup ~~(restoration?)~~ destination path naming and 7z.exe usage method. store full path or not in the backup destination? what should be 7z.exe file source? shadow mount drive, or local backup destination? answer and document everything. [10/26/2020 1:05 PM CST]

FINAL DECISION/CONCLUSION 1 [10/26/2020 1:17 PM CST]
I need multi-volume archive files. Uploading one big archive file that can and would be tens or more gigabytes or even terabytes is not doable or a good method. Since I need multi-volume archive files, appending to 7z.exe created zip files cannot be done (using 7z.exe at the least).

A POSSIBILITY [10/26/2020 1:28 PM CST]
in the backup destination, each shadow-mounted backup source directory name can be something else, like SHA-256 or SHA-1, or mapped to a shorter directory name like 1, to avoid the path length limit overflow. a separate plain text file in the backup destination directory can be used for saving the mapping, each line containing a directory mapping using a tap character to separate the original directory path and the mapped or transformed directory path.
A MUST [10/26/2020 1:35 PM CST] path length limit overflow must be avoided.

STATUS UPDATE [10/26/2020 1:35 PM CST]
how about the following?
using multi-volume archive files.
using shadow-mounted backup source directory name mapping or shortening with a mapping plain text file.

using the local backup destination files for AWS S3 backup, so that one set of AWS S3 backup files can be used, instead of multiple for each shadow-mounted backup source directory. (this requires always doing local backup for AWS S3 backup, which is not what I want.)

(TO DO [10/26/2020 7:53 PM CST] put the following in the appropriate places in "AAM Auto Backup Operation Modes and Command Formats" and "Software Process Specs" parts.)
FINAL DECISION [10/26/2020 1:39 PM CST]

- **support more than one AWS S3 full backup in the same month (for use when incomplete-upload AWS S3 full backup, and/or missing or additional shadow-mounted backup source directory[ies] in complete-upload AWS S3 full backup [due to the AAM Auto Backup instruction plain text file content change]).** (NOTE, REMINDER [10/26/2020 10:04 AM CST] if the latest complete-upload AWS S3 full backup does not contain all the backup sources or shadow-mounted directories (and their contents), AAM Auto Backup performs a full backup.)
- **support force AWS S3 full backup via an additional regular-backup command argument.** (TO DECIDE [10/26/2020 8:02 PM CST] do 'force full backup on AWS', check the modification date and time on the AAM Auto Backup instruction plain text file to auto-decide doing full backup, or do both? I think the second option of auto-deciding is really bad and unreliable, so I won't do it. well, maybe I will. I think I'll support both. maybe support another command argument, 'full backup when AAM Auto Backup instruction plain text file is modified after the last backup date and time'. note that an AWS S3 full backup is required, because new sub-directories within the same shadow-mounted directory can contain files that are older than the last backup date and time, which would prevent incremental backup of the new files in the new sub-directories.)
- **use multi-volume archive files. I must.**
- **use the original method of creating multiple sets of AWS S3 backup files, one for each shadow-mounted backup source directory.**
- **use the shadow mounts. no reason to rely on local backup for AWS S3 backup.**
- **use relative path in archived files to avoid path limit overflow.**
- **when restoring AWS S3 backup, use a directory named with a number starting at 1 for each sub-set of backup, each shadow-mounted backup source directory. (or better yet, starting at a. [not in this version, later on].)**
- [10/26/2020 8:58 PM CST] (one more thing to handle) the local backup destination directory structure—the shadow-mount directory naming in the local backup destination directory. how to do this? use a directory path mapping text file in the local backup destination directory? it can be very unreliable—the file could go missing or get damaged. I don't like the idea of creating a folder per directory path element, since that can and will result in copied file path overflow.

(the above handles all the issues. using multi-volume archive files, and avoiding path limit overflow. the above is the way to go.)

TO CONSIDER [10/26/2020 1:20 PM CST] when there is archive file upload failure, how about auto-trying it later, instead of deleting all the archive files, remaking all the archive files, then reuploading the next day? well, I don't think it would be a good design choice; due to the main work computer files constantly being updated, if there is a upload failure, the archive files should be generated again—for now, that's what I think. upload failure won't happen often anyway.

8.3. 7z.exe test commands

8.3.1. 7z.exe commands to try

create c:\temp\test1.txt and f:\temp\test2.txt.

execute the following commands.

```
cd /D c:\temp\
"C:\Program Files\7-Zip\7z" a test.zip -spf c:\temp\test1.txt
"C:\Program Files\7-Zip\7z" a test.zip -spf f:\temp\test2.txt
```

after executing each 7z.exe command, check the zip file content.

delete c:\temp\test1.txt and f:\temp\test2.txt.

create c:\temp\extract\, and move test.zip to that directory.

execute the following.

```
cd /D c:\temp\extract\
"C:\Program Files\7-Zip\7z" x test.zip
```

check the extracted files.

RESULT [10/26/2020 12:38 PM CST] the above does exactly what I need, appending to existing archive files, with the colon in drive letter replaced with an underscore. now, I seriously need to think about creating only one set of multi-volume archive files for each backup source or computer, instead of creating one set of archive files per shadow-mount directory.

[10/26/2020 12:41 PM CST] the only concern is some archived files failing to being written due to having the extra two or more characters in the path, the drive letter and underscore, which could make the file path too long for being written.

8.3.2. more 7z.exe commands to try (with encryption)

create c:\temp\test1.txt and f:\temp\test2.txt.

execute the following commands.

```
cd /D c:\temp\

"C:\Program Files\7-Zip\7z" a test.zip -v4000000000b -mcu=on -
mem=AES256 -1234abcd -spf c:\temp\test1.txt

"C:\Program Files\7-Zip\7z" a test.zip -v4000000000b -mcu=on -
mem=AES256 -1234abcd -spf f:\temp\test2.txt
```

after executing each 7z.exe command, check the zip file content.

delete c:\temp\test1.txt and f:\temp\test2.txt.

create c:\temp\extract\, and move test.zip to that directory.

execute the following.

```
cd /D c:\temp\extract\

"C:\Program Files\7-Zip\7z" x test.zip -1234abcd
```

check the extracted files.

8.3.2.1. error

[10/26/2020 12:54 PM CST]

```
c:\Temp>"C:\Program Files\7-Zip\7z" a test.zip -v4000000000b -mcu=on -mem=AES256
 -1234abcd -spf f:\temp
\test2.txt

7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21


Error:
Updating for multivolume archives is not implemented
test.zip.001
Not implemented


System ERROR:
Not implemented
```

[10/26/2020 12:54 PM CST] I'm glad I have performed this test! otherwise, I would've wasted so much time writing the code that would not have worked!

8.3.3. NOTES [10/26/2020 11:36 AM CST]

```
7z a test.zip test1.txt
```

```
7z a test.zip test2.txt
```

-spf (Use fully qualified file paths) switch
Enables the mode that allows to use fully qualified file paths in archives. If -spf switch is not specified, 7-Zip reduces file paths to relative paths when it adds files to archive, and 7-Zip converts paths to relative paths when you extract archive. If -spf switch is specified, 7-Zip doesn't try to process or convert paths.

Examples
```
7z a a.7z -spf c:\Files\test.txt d:\test.txt
```
stores both txt files with full paths.
```
7z x a.7z -spf
```
extracts files from a.7z archive with exact file paths specified in archive.
Commands that can be used with this switch
a (Add), d (Delete), e (Extract), u (Update), x (Extract with full paths)

-u (Update options) switch
Note: the updating of solid .7z archives can be slow, since it can require some recompression.

-w (set Working directory) switch
Sets the working directory for the temporary base archive. By default, 7-Zip builds a new base archive file in the same directory as the old base archive file. By specifying this switch, you can set the working directory where the temporary base archive file will be built. After the temporary base archive file is built, it is copied over the original archive; then, the temporary file is deleted.

-x (Exclude filenames) switch
Specifies which filenames or wildcarded names must be excluded from the operation.
Multiple exclude switches are supported.

e (Extract) command
Extracts files from an archive to the current directory or to the output directory. The output directory can be specified by -o (Set Output Directory) switch.
This command copies all extracted files to one directory. If you want extract files with full paths, you must use x (Extract with full paths) command.

x (Extract with full paths) command
Extracts files from an archive with their full paths in the current directory, or in an output directory if specified.
See the e (Extract) command description for more details.

Examples
7z x archive.zip
extracts all files from the archive archive.zip to the current directory.
7z x archive.zip -oc:\soft *.cpp -r
extracts all *.cpp files from the archive archive.zip to c:\soft folder.

8.4. other compression programs

[10/26/2020 2:07 PM CST]
search "GNU zip windows" online.
http://gnuwin32.sourceforge.net/packages/zip.htm
(CONCLUSION [10/26/2020 2:16 PM CST] not worth bothering with. I will use 7z.exe.)

8.5. additional notes in function comments (deprecated) [12/8/2020 4:33 PM CST]

clean up the starting comments in _QueryBackupAwsS3Bucket() and _QueryBackupAwsS3BucketForDataRestoration(). copy what I want to keep to the AAM Auto Backup document. update what I want to clarify. [12/8/2020 4:08 PM CST]

see if the _QueryBackupAwsS3Bucket() portion of the code can be reused in _QueryBackupAwsS3BucketForDataRestoration() by calling _QueryBackupAwsS3Bucket(), or merging _QueryBackupAwsS3BucketForDataRestoration() into _QueryBackupAwsS3Bucket() with a function argument indicating whether the query is for backup or restoration. I don't like the redundant code; there should not be any redundant code. maybe I can merge the two functions into one function, since _QueryBackupAwsS3BucketForDataRestoration() is based on _QueryBackupAwsS3Bucket(), and what it does is building some additional data variables. [12/8/2020 4:13 PM CST] (UPDATE [12/8/2020 4:20 PM CST] I just reviewed the codes of both functions. I don't think they can be merged. they look significantly different.)

```
    #UPDATE [11/2/2020 6:25 AM CST] function design change
    #no returning any value.
    #when AWS S3 bucket query error, set self._bAwsS3QueryFailure to True,
    #then return.
    #if no complete-upload full-backup AWS S3 objects in the month given,
    #with all the backup sources,
    #set all of the following data variables to the empty string.
    #self._strLatestCompleteUploadAwsS3FullBackupDate
    #self._strLatestCompleteUploadAwsS3FullBackupTime
    #self._strLatestCompleteUploadAwsS3IncrementalBackupDate
    #self._strLatestCompleteUploadAwsS3IncrementalBackupTime
    #Otherwise, all of the above four data variables are set to the values
    #retrieved from AWS S3.
    #if no incremental backup, the latest incremental date and time is
    #set to the full-backup date and time.
    #because this function is for data backup, not restoration, this
    #function does not retrieve the AWS S3 backup object lists.
    #
    #I'll probably keep on using str_listRemainingBackupSourceAwsS3UniqueNames.
    #
    #
    #When the AWS S3 query is successful, this function sets the relevant
    #data variables to the latest completed-upload full and incremental
    #backup dates and times.
    #
    #
    #NOTE [10/26/2020 10:06 AM CST]
    #this function, _QueryBackupAwsS3Bucket(), retrieves
    #the AWS S3 full backup with all of the backup sources or
    #shadow-mounted directories fully uploaded.  AAM Auto Backup does not
    #consider a full backup with an incomplete-upload backup source as
    #a complete-upload full backup.

    #[10/22/2020 7:24 PM CST]
    #multiple backup sources or shadow-copy mounted directories.
    #how to handle that properly?
    #in all likely chance, check for the upload-completeness of
    #full backup for every backup source or
    #shadow-copy mounted directory.  how?  to do that,
    #I need to process the AAM Auto Backup instruction plain text file
    #first.




    #Returns True if AWS S3 query success.
    #Returns False if AWS S3 query failure.
    #
```

```
#When the AWS S3 query is successful, this function sets the relevant
#data variables to the latest completed-upload full and incremental
#backup dates and times.
#
#this function sets the following data variables when the query
#is successful.
#self._strLatestCompleteUploadAwsS3FullBackupDate
#self._strLatestCompleteUploadAwsS3FullBackupTime
#self._strLatestCompleteUploadAwsS3IncrementalBackupDate
#self._strLatestCompleteUploadAwsS3IncrementalBackupTime
#
#self._dictBackupSourceNamesToFullBackupAwsS3ObjectNameLists
#self._dictBackupSourceNamesToIncrementalBackupAwsS3ObjectNameLists
#self._dictBackupSourceNamesToIncrementalBackupArchiveFileNameLists
#
#NOTE [10/26/2020 10:06 AM CST]
#this function, like _QueryBackupAwsS3Bucket(), retrieves
#the AWS S3 full backup with all of the backup sources or
#shadow-mounted directories fully uploaded.  AAM Auto Backup does not
#consider a full backup with an incomplete-upload backup source as
#a complete-upload full backup.


#FINAL DECISION [10/30/2020 4:37 PM CST]
#copy, paste, and edit the code from
#_QueryBackupAwsS3BucketForRegularBackup(), since heavy
#modification is needed.
#
#UPDATE [10/30/2020 4:46 PM CST]
#I think I should modify _QueryBackupAwsS3BucketForRegularBackup()
#to save object names by backup-source names.
#actually, I should create a general AWS S3 query function, named
#_QueryBackupAwsS3Bucket(), and modify the other
#three AWS S3 query functions to use the general AWS S3 query function.
#_QueryBackupAwsS3Bucket() should take iYear, iMonth,
#and strUniqueBackupSourceLocationName as its function arguments.
#(with the above method, I think I can rather quickly complete
#all the AWS S3 query functions!  now I think I can!)


#TO DO [10/30/2020 6:30 PM CST]
#hold on.  an issue to resolve first before coding.
#when restoring, AAM Auto Backup does not know about the backup sources,
#and it shouldn't care about the backup sources in the backup
#instruction file(s).  so, what to do about that?
#how about simply getting the latest full backup with complete upload
```

```
#for all backup sources,
#then using the incremental backups of that full backup?
#I'll probably do that.  _QueryBackupAwsS3BucketForDataRestoration()
#might have its own AWS S3 code.
#
#update str_listRemainingBackupSourceAwsS3UniqueNames
#to dictBackupSourceAwsS3NamesToPresenceStatuses
#(a presence status is a boolean value.)
#(UPDATE [10/30/2020 7:12 PM CST] actually,
#using dictBackupSourceAwsS3NamesToPresenceStatuses is required
#only in restoration.)
#
#build self._dictBackupSourceNamesToFullBackupAwsS3ObjectNameLists and
#self._dictBackupSourceNamesToIncrementalBackupAwsS3ObjectNameLists
#throughout the code.


#[10/22/2020 7:24 PM CST]
#multiple backup sources or shadow-copy mounted directories.
#how to handle that properly?
#in all likely chance, check for the upload-completeness of
#full backup for every backup source or
#shadow-copy mounted directory.  how?  to do that,
#I need to process the AAM Auto Backup instruction plain text file
#first.
```

8.6. old, deprecated code [12/9/2020 5:37 AM CST]

```
"""
######################################################################
######################################################################
######################################################################
strUniqueBackupSourceLocationName = \
    regexp_match_objBackupAwsS3ObjectName.group(1)

strFullBackupDate = regexp_match_objBackupAwsS3ObjectName.group(2)
strFullBackupTime = \
    regexp_match_objBackupAwsS3ObjectName.group(3) + '.' + \
    regexp_match_objBackupAwsS3ObjectName.group(4)

strIncrementalBackupDate = \
    regexp_match_objBackupAwsS3ObjectName.group(5)
strIncrementalBackupTime = \
    regexp_match_objBackupAwsS3ObjectName.group(6) + '.' + \
    regexp_match_objBackupAwsS3ObjectName.group(7)

if regexp_match_objBackupAwsS3ObjectName.group(8) == 'yes':
    bUploadCompletionIndicator = True
else:
    bUploadCompletionIndicator = False

strBackupSourceUniqueName = \
    regexp_match_objBackupAwsS3ObjectName.group(9)

#[10/30/2020 2:03 PM CST]
#God, it's so good to finally have retrieving all of the above
#in code!  finally!!!!!

if strFullBackupDate != strLastFullBackupDate and \
    strFullBackupTime != strLastFullBackupTime:
    pass
"""




"""
####################################################################
####################################################################
####################################################################
strUniqueBackupSourceLocationName = \
    regexp_match_objBackupAwsS3ObjectName.group(1)

strFullBackupDate = regexp_match_objBackupAwsS3ObjectName.group(2)
strFullBackupTime = \
    regexp_match_objBackupAwsS3ObjectName.group(3) + '.' + \
    regexp_match_objBackupAwsS3ObjectName.group(4)

strIncrementalBackupDate = \
    regexp_match_objBackupAwsS3ObjectName.group(5)
strIncrementalBackupTime = \
    regexp_match_objBackupAwsS3ObjectName.group(6) + '.' + \
    regexp_match_objBackupAwsS3ObjectName.group(7)

if regexp_match_objBackupAwsS3ObjectName.group(8) == 'yes':
    bUploadCompletionIndicator = True
else:
    bUploadCompletionIndicator = False

strBackupSourceUniqueName = \
   regexp_match_objBackupAwsS3ObjectName.group(9)

#[10/30/2020 2:03 PM CST]
#God, it's so good to finally have retrieving all of the above
#in code!  finally!!!!!

if strFullBackupDate != strLastFullBackupDate and \
   strFullBackupTime != strLastFullBackupTime:
   pass
"""



"""
############################################################
############################################################
############################################################
strUniqueBackupSourceLocationName = \
   regexp_match_objBackupAwsS3ObjectName.group(1)

strFullBackupDate = regexp_match_objBackupAwsS3ObjectName.group(2)
strFullBackupTime = \
   regexp_match_objBackupAwsS3ObjectName.group(3) + '.' + \
   regexp_match_objBackupAwsS3ObjectName.group(4)

strIncrementalBackupDate = \
   regexp_match_objBackupAwsS3ObjectName.group(5)
strIncrementalBackupTime = \
   regexp_match_objBackupAwsS3ObjectName.group(6) + '.' + \
   regexp_match_objBackupAwsS3ObjectName.group(7)

if regexp_match_objBackupAwsS3ObjectName.group(8) == 'yes':
   bUploadCompletionIndicator = True
else:
   bUploadCompletionIndicator = False

strBackupSourceUniqueName = \
   regexp_match_objBackupAwsS3ObjectName.group(9)

#[10/30/2020 2:03 PM CST]
#God, it's so good to finally have retrieving all of the above
#in code!  finally!!!!!

if strFullBackupDate != strLastFullBackupDate and \
   strFullBackupTime != strLastFullBackupTime:
   pass
```

```
        """



        """
        ##################################################################
        ##################################################################
        ##################################################################
        strUniqueBackupSourceLocationName = \
            regexp_match_objBackupAwsS3ObjectName.group(1)

        strFullBackupDate = regexp_match_objBackupAwsS3ObjectName.group(2)
        strFullBackupTime = \
            regexp_match_objBackupAwsS3ObjectName.group(3) + '.' + \
            regexp_match_objBackupAwsS3ObjectName.group(4)

        strIncrementalBackupDate = \
            regexp_match_objBackupAwsS3ObjectName.group(5)
        strIncrementalBackupTime = \
            regexp_match_objBackupAwsS3ObjectName.group(6) + '.' + \
            regexp_match_objBackupAwsS3ObjectName.group(7)

        if regexp_match_objBackupAwsS3ObjectName.group(8) == 'yes':
            bUploadCompletionIndicator = True
        else:
            bUploadCompletionIndicator = False

        strBackupSourceUniqueName = \
            regexp_match_objBackupAwsS3ObjectName.group(9)

        #[10/30/2020 2:03 PM CST]
        #God, it's so good to finally have retrieving all of the above
        #in code!  finally!!!!!

        if strFullBackupDate != strLastFullBackupDate and \
            strFullBackupTime != strLastFullBackupTime:
            pass
        """
```

[12/11/2020 9:58 AM CST]

```
        #strIncrementalBackupDate = \
        #    regexp_match_objBackupAwsS3ObjectName.group(5)
```

```
#if strIncrementalBackupDate == '0': #a full backup
#    continue


#strFullBackupDate = regexp_match_objBackupAwsS3ObjectName.group(2)
#strFullBackupTime = \
#    regexp_match_objBackupAwsS3ObjectName.group(3) + '.' + \
#    regexp_match_objBackupAwsS3ObjectName.group(4)

#if strFullBackupDate != \
#    self._strLatestCompleteUploadAwsS3FullBackupDate or \
#    strFullBackupTime != \
#    self._strLatestCompleteUploadAwsS3FullBackupTime:
#    continue
```

[12/11/2020 10:54 AM CST]

```
#strIncrementalBackupDate = \
#    regexp_match_objBackupAwsS3ObjectName.group(5)

#if strIncrementalBackupDate == '0': #a full backup
#    continue


#strFullBackupDate = regexp_match_objBackupAwsS3ObjectName.group(2)
#strFullBackupTime = \
#    regexp_match_objBackupAwsS3ObjectName.group(3) + '.' + \
#    regexp_match_objBackupAwsS3ObjectName.group(4)

#if strFullBackupDate != \
#    self._strLatestCompleteUploadAwsS3FullBackupDate or \
#    strFullBackupTime != \
#    self._strLatestCompleteUploadAwsS3FullBackupTime:
#    continue
```

8.7. NOTES [12/11/2020 9:31 AM CST]

update the _QueryBackupAwsS3Bucket() and _QueryBackupAwsS3BucketForDataRestoration() code to keep self._strLatestCompleteUploadAwsS3IncrementalBackupDate and self._strLatestCompleteUploadAwsS3IncrementalBackupTime the latest. the current code updates them to the earliest. [12/10/2020 2:44 PM CST] it looks like I need to use strLastIncrementalBackupDate and strLastIncrementalBackupTime. [12/10/2020 3:02 PM CST]

9. AWS S3 upload

[11/2/2020 8:02 PM CST]

```
    #https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/
s3.html#S3.Object.upload_file
        #import boto3
        #s3 = boto3.resource('s3')
        #s3.Object('mybucket', 'hello.txt').upload_file('/tmp/hello.txt')
        #https://boto3.amazonaws.com/v1/documentation/api/latest/_modules/boto3/s3/
transfer.html
        try:
            ##https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/
s3.html#S3.Bucket.upload_file
            self.s3_resource.Bucket('mybucket').upload_file('/tmp/hello.txt', 'hello.txt')

            #the following for test only.
            s3_objectToAdd = self.s3_resource.Object('mybucket',
'hello.txt').upload_file('/tmp/hello.txt')
            s3_objectToAdd.wait_until_exists()
        except Exception as exc:
            str(exc)
```

10. "Incremental-only Backup Operation Mode" implementation method

[11/3/2020 11:15 AM CST]

[11/3/2020 12:01 PM CST] I need further existing code reading to verify this, but for now, all that I need to do to add the incremental-only backup code, is to move the current ExecuteRegularBackup() code to _ExecuteBackup(), and modify only the AWS S3 query code.

think and decide on how to implement ExecuteIncrementalOnlyBackup(). do I want to create and use a shared class function, _ExecuteBackup(), that is used by ExecuteRegularBackup() and ExecuteIncrementalOnlyBackup()? document everything applicable; update code as needed. [11/3/2020 10:09 AM CST]

11. 7z.exe list file generation methods

[11/4/2020 12:13 PM CST]

11.1. design decision(s) to make: including and excluding files in and from archive files

[11/5/2020 8:19 AM CST] (In the 7z.exe list file, I think using wildcard is possible. As such, I can and should simplify my job. List the directories to archive in the 7z.exe list file, with wild card, not every file. See if there's a way to omit certain directories and files. Test this.)

how to do this, exactly?

option 1 (the easiest way)
if no folders and/or files to exclude, use the relative paths of the directories to backup ~~with *~~ to indicate backing up everything in each directory.

~~option 2 (the most crude method) (not using this. no need. use the -x switch.)~~
~~when there are folders and/or files to exclude, scan all the relevant folders, check the full path of every file, and leave out the folders and files to exclude in the 7z.exe list file. that is, build the list of all the files to include in the 7z.exe list file.~~

option 2 (use available 7z.exe switches)
~~-i (Include filenames) switch (supports list file) (no need to use this separately.)~~
-r (Recurse subdirectories) switch (use this if needed, for wildcards. the default is off?)
-x (Exclude filenames) switch (supports list file) (use this.)
so, use one list file for the directories and files to include, and use one list file for the directories and files to exclude. (test this.)
[11/5/2020 9:25 AM CST] very likely, use -x twice, with two separate exclusion 7z.exe list files—one for no sub-directory recursion (i.e. non-recursive exclusion, when the exclusion is specified in full path), one for wildcard and file-name sub-directory recursion (i.e. recursive exclusion).

11.2. additional 7z.exe test(s)

11.2.1. TEST 2 [11/28/2020 5:22 PM CST]

11.2.1.1. backup-source directory content

[11/28/2020 4:03 PM CST]

- 01test1
- 01test2
- 01test3
- 02test1
    - test1.txt
    - test2.txt
    - 02test1-1
        - test1.txt

- 02test2
  - test1.txt
  - test2.txt
- 02test3
  - 1 (folder)
- 03test1
- 03test2
- 03test3

11.2.1.2. inclusion_listfile.txt content

02test1\
02test2\
02test3\

11.2.1.3. non-recursive_exclusion_listfile.txt content

02test3\1\

11.2.1.4. recursive_exclusion_listfile.txt content

02test1-*\
test2.*

11.2.1.5. 7z.exe command to execute

```
cd /D c:\temp\backup_source

"c:\program files\7-zip\7z" a test.zip -v4000000000b -mcu=on -
mem=AES256 -1234abcd @inclusion_listfile.txt -xr-@non-
recursive_exclusion_listfile.txt -
xr@recursive_exclusion_listfile.txt
```

11.2.1.6. test result

[11/29/2020 9:33 AM CST] the exactly desired result produced! **7z.exe supports wild cards in directory names!**

[11/29/2020 9:32 AM CST]

```
C:\Users\Allen>cd /D c:\temp\backup_source

c:\Temp\backup_source>"c:\program files\7-zip\7z" a test.zip -v4000000000b -mcu=
on -mem=AES256 -
p1234abcd2D4A03F4DF7932CABE5F2DBB86FDC6A0D047A31CD7E07BBE0EC
A @inclusion_listfile.txt -xr-@non-recursive_exclusion_listfile.txt -xr@recursiv
e_exclusion_listfile.txt

7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Scanning the drive:
3 folders, 2 files, 0 bytes

Creating archive: test.zip

Add new data to archive: 3 folders, 2 files, 0 bytes


Files read from disk: 2
Archive size: 798 bytes (1 KiB)
Everything is Ok

c:\Temp\backup_source>
```

11.2.2. TEST 1 [11/5/2020 9:03 AM CST]

11.2.2.1. backup-source directory content

```
\include1\
\include1\include1-1\
\include1\include1-1-1\
\include1\include1-1-2\
\include1\include1-2\
\include2\
\include2\include2-1\
\include2\include2-2\
\include2\include2-2-1\
\include2\include2-2-2\
\include2\include2-2-3\
\include3\
\exclude1\
\exclude2\
```

a ~WRL*.tmp file in every folder.

a text file in every folder.

11.2.2.2. inclusion_listfile.txt content

```
include1\
include2\
include3\
```

11.2.2.3. non-recursive_exclusion_listfile.txt content

```
include2\include2-2\
```

11.2.2.4. recursive_exclusion_listfile.txt content

```
~WRL*.tmp
include1\include1-1\*.txt
```

11.2.2.5. 7z.exe command to execute

```
cd /D c:\temp\backup_source

"c:\program files\7-zip\7z" a test.zip -v4000000000b -mcu=on -
mem=AES256 -1234abcd @inclusion_listfile.txt -xr-@non-
recursive_exclusion_listfile.txt -
xr@recursive_exclusion_listfile.txt
```

11.2.2.6. test result

[11/5/2020 11:37 AM CST]

```
C:\Users\Allen>cd /D c:\temp\backup_source

c:\Temp\backup_source>"c:\program files\7-zip\7z" a test.zip -v4000000000b -mcu=
on -mem=AES256 -
p1234abcd2D4A03F4DF7932CABE5F2DBB86FDC6A0D047A31CD7E07BBE0EC
A @inclusion_listfile.txt -xr-@non-recursive_exclusion_listfile.txt -xr@recursiv
e_exclusion_listfile.txt

7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Scanning the drive:
8 folders, 5 files, 0 bytes
```

```
Creating archive: test.zip

Add new data to archive: 8 folders, 5 files, 0 bytes


Files read from disk: 5
Archive size: 2248 bytes (3 KiB)
Everything is Ok

c:\Temp\backup_source>
```

[11/5/2020 11:45 AM CST] **the archive file contains exactly what I expected it to contain! 7z.exe works exactly in the way I want it to!**

11.2.3. 7z.exe exclusion capability conclusions

[11/29/2020 9:37 AM CST]

7z.exe supports directory name and full/absolute directory path exclusion.

7z.exe supports wild cards in directory names.

7z.exe supports file name and full/absolute file path exclusion.

7z.exe supports wild cards in file names.

11.3. full-backup 7z.exe list file generation method

[11/4/2020 12:14 PM CST]

FINAL DECISION [11/5/2020 11:50 AM CST]
use the 7z.exe test method above. one inclusion list file, one non-recursive exclusion list file, and one recursive exclusion list file. use the following 7z.exe command.

```
cd /D q:

"c:\program files\7-zip\7z" a "c:\xx\backup.zip" -v4000000000b -
mcu=on -mem=AES256 -1234abcd @"c:\xx\inclusion_listfile.txt" -xr-
@"c:\xx\non-recursive_exclusion_listfile.txt" -xr@"c:\xx\
recursive_exclusion_listfile.txt"
```

11.3.1. issue of omitting folders and files in archiving via 7z.exe

[11/4/2020 3:17 PM CST]

FINAL DECISION [11/5/2020 11:48 AM CST]
use two -x 7z.exe switches, one with non-recursive exclusion list file, and one with recursive exclusion list file.

11.4. incremental-backup 7z.exe list file generation method

[11/4/2020 12:14 PM CST]

FINAL DECISION [11/5/2020 11:50 AM CST]
list all the files to include in the archive file(s), in one inclusion list file; no exclusion list file.
use the following 7z.exe command.

```
cd /D q:

"c:\program files\7-zip\7z" a "c:\xx\backup.zip" -v4000000000b -
mcu=on -mem=AES256 -1234abcd @"c:\xx\inclusion_listfile.txt"
```

11.4.1. update [12/3/2020 12:58 PM CST]

7z.exe incremental backup list file generation algorithm
Using the os.walk() function, check each file of the shadow-mounted directory.
Skip every excluded directory.
Skip every excluded file.
If the file is created or modified after the last backup time, add the file to the 7z.exe incremental backup list file.

If necessary, keep both exclusion file and folder full-path lists, and regular expressions.

Review the current code, and decide what to do exactly.

Keep the above notes in a proper 'implementation techniques' section in the AAM Auto Backup document.

NOTE [12/3/2020 1:33 PM CST]
how about a smarter implementation in code? full-path and name lists, detection and use of file separator and wild cards (? and *) in the shadow-mounted drive file. I'm leaning toward this.
FINAL DECISION [12/3/2020 1:53 PM CST] I'm doing the above. It feels right!

11.4.2. issue of omitting folders and files in archiving via 7z.exe

[11/6/2020 5:50 AM CST]
for file-name wild cards, use regular expression.

[11/6/2020 5:51 AM CST]
no directory-name wild cards support for now. maybe later for , if Robocopy supports directory-name wild cards. (UPDATE [12/22/2020 4:03 PM CST] this is supported in the AAM Auto Backup source code.)

REQUIRED UPDATE [11/6/2020 7:16 AM CST]
wait. file modification time (alone) is not what should be used, in deciding which files to incrementally back up. I think both file creation and modification times must be used. file creation time (Windows only) for files that are copied; file modification time for files that are modified. last file access time should not be used since unchanged file (copied or modified) has no reason to be incrementally backed up.

if file creation and modification time is after the last (full or incremental) backup time, incrementally back up the file.

```
#------------------------------------------------
#Check if the user-specified file exclusion(s)
#apply to the files in this folder.
#Record the files that are not excluded.
#------------------------------------------------
for strFileName in str_listFileNames:

   #[11/6/2020 5:25 AM CST]
   #how to do this exactly, especially for wildcards?
   #use regexp?  yeah, I'll probably use regexp.
   #to build the regexp, scan each character of
   #each user-specified excluded file spec, and
   #create the regexp by
   #performing the following character conversion.
   #build the file-exclusion tester regexp list
   #beforehand to use here.
   #replace ? with ..
   #replace * with *?.
   #replace . with \..
   #no character conversion for all the other
   #characters.

   #REQUIREMENT [11/6/2020 6:26 AM CST]
```

```
#if the exclusion-file spec includes directory
#path, not just the file full name,
#that must be handled properly also.
```

11.4.3. 7z.exe execution with empty inclusion list file

FINAL DECISION [12/22/2020 4:29 PM CST]
when building strBackupSourceArchiveFilesUploadCompletionIndicatorFileFullName, strIncrementalBackupDate and strIncrementalBackupTime are used, which are extracted from the archive file name. it is possible to bypass using strIncrementalBackupDate and strIncrementalBackupTime from the archive file name, by using the backup date and time. however, I want to keep the empty archive file in AWS S3 Glacier Deep Archive, as a record of a sort, and have it decompressed during restoration. updating the source code is not worth the time and effort.

resolve the "if iNumberOfFilesIncrementallyBackedUp == 0:" and "#auto-generate (overwrite or create) the backup batch file." code blocks. use "str7zExeCommand = "" or not?
run 7z.exe with an empty list file. see what terminal output that generates. [12/22/2020 12:22 PM CST] (I am leaning toward not running 7z.exe when there is no reason to. [12/22/2020 4:00 PM CST])
check the shadow copy output check code, and find out what it does when there is no files to incrementally backup to AWS. [12/22/2020 12:23 PM CST]

11.4.3.1. 7z.exe output with empty inclusion list file

[12/22/2020 4:10 PM CST]

```
C:\Users\Allen\Documents\
\AAM Auto Backup\source code>"c:\program files\7-zip\7z" a "backup.zip" -v400000
0000b -mcu=on -mem=AES256 -
p1234abcd2D4A03F4DF7932CABE5F2DBB86FDC6A0D047A31C
D7E07BBE0ECA @"20201222 112502.225333_inclusion_7z_list_file.txt"

7-Zip 19.00 (x64) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Scanning the drive:
0 files, 0 bytes

Creating archive: backup.zip

Add new data to archive: 0 files, 0 bytes
```

```
Files read from disk: 0
Archive size: 22 bytes (1 KiB)
Everything is Ok

C:\Users\Allen\Documents\
\AAM Auto Backup\source code>
```

11.5. algorithm notes

[11/5/2020 2:06 PM CST]

Full-backup 7z.exe list files creation algorithm.
Create one 7z.exe list file at a time.
if drive letter in the exclusion folder or file, non-recursive exclusion.
if no drive letter in the exclusion folder or file, recursive exclusion.

Incremental-backup 7z.exe list file creation algorithm.
Walk each directory to back up.
Check and omit each file if affected by directory or file exclusion list.
Check each file modification date time. Include the file in the 7z.exe inclusion list file if needed.

Search "python get file modified time" online.
https://stackoverflow.com/questions/237079/how-to-get-file-creation-modification-date-times-in-python
os.path.getmtime(path)
os.path.getctime()

[11/6/2020 6:57 AM CST]
search "Python convert epoch time to datetime" online.
https://stackoverflow.com/questions/12400256/converting-epoch-time-into-the-datetime

Search "python remove drive letter from path" online
https://stackoverflow.com/questions/10507298/splitting-path-strings-into-drive-path-and-file-name-parts
os.path.splitdrive

12. Robocopy result verification

[11/5/2020 2:06 PM CST]

Robocopy verification method
Walk every backup-source directory in the shadow-mounted drive.

Pass every excluded file.
If non-excluded file in the backup destination does not have matching modification (or creation?) date time as the source, log the error.

[11/6/2020 6:26 PM CST]
search "Python how to get file size" online.
https://stackoverflow.com/questions/6591931/getting-file-size-in-python
os.path.getsize(path)

13. AWS S3 data restoration implementation techniques

[11/7/2020 10:53 AM CST]

In the one-step data restoration terminal, pause for user input after displaying that the full and incremental backups to use in restoration has been saved to a text file, and tell user to open and check the file before proceeding.
[11/9/2020 9:59 AM CST] the above will not be done. the full and incremental backups to use in data restoration will be saved in the local log file, and displayed in terminal.

13.1. overview

decide on how to complete the remaining Aam_Auto_Backup class functions. first, decide on what to do with the local backup log file opening and closing in the data restoration class functions; decide on whether to open the log file only if the backup operation type is regular backup, incremental-only backup, or one-step AWS S3 data restoration. local backup log file is used in _QueryBackupAwsS3BucketForDataRestoration() currently; handle that issue (maybe use the local backup log file, only if the operation type is one-step AWS S3 data restoration?). document and implement all the decisions. [11/7/2020 7:25 AM CST] (UPDATE [11/7/2020 10:10 AM CST] I'm leaning toward using the local backup log file, only for regular backup, incremental-only backup, and one-step AWS S3 data restoration—not for the individual data-restoration stage operations. so, the terminal output only for the individual data-restoration stage operations? UPDATE [11/7/2020 10:27 AM CST] yeah, in all likely chance, this is what I'll do.) (UPDATE [11/7/2020 10:43 AM CST] in all likely chance, I'll implement the following. when there's an error in an individual data-restoration stage operation, an exception will be raised; such an exception will be handled in RestoreAwsS3DataToLocalComputerInOneStep() for closing the local backup log file, but it will not be handled when an individual data-restoration stage operation is executed separately.) (UPDATE [11/7/2020 11:21 AM CST] also, send email only for regular backup, incremental-only backup, and one-step AWS S3 data restoration.) (NOTE [11/7/2020 10:44 AM CST] ok, I just need to decide on these issues, and then I'll be ready to complete the AAM Auto Backup code drafting!!!!!)

update _QueryBackupAwsS3BucketForDataRestoration() to use the local backup log file only for the one-step data restoration. [11/7/2020 1:13 PM CST] reconsider and decide on using str_listUniqueBackupSourceLocationNames. using strUniqueBackupSourceLocationName is easier to code. separate AAM Auto Backup instances can be run for data restoration, one for each backup-source location. document the decision and implement the decision. [11/8/2020 1:12 PM CST] I'm heavily leaning toward using strUniqueBackupSourceLocationName. no need to complicate coding. [11/8/2020 1:16 PM CST] (AN OPTION [11/8/2020 1:51 PM CST] use self.str_listUniqueBackupSourceLocationNames with a for loop in RestoreAwsS3DataToLocalComputerInOneStep(). use the first element of self.str_listUniqueBackupSourceLocationNames only, when executing individual data restoration operation? I keep thinking using just strUniqueBackupSourceLocationName is the best solution.) (UPDATE [11/8/2020 2:25 PM CST] yeah, I am definitely leaning toward using self.str_listUniqueBackupSourceLocationNames. otherwise, I'll have to use a thread to check the different sets of AWS S3 objects separately for restoration.) (FINAL DECISION [11/10/2020 9:53 AM CST] I'm using strUniqueBackupSourceLocationName, not str_listUniqueBackupSourceLocationNames.)

make sure that _QueryBackupAwsS3BucketForDataRestoration() is called only once in RestoreAwsS3DataToLocalComputerInOneStep(). that is, if the list of AWS S3 objects to use in data restoration exists, do not call _QueryBackupAwsS3BucketForDataRestoration() again to get the list of the AWS S3 objects to check for restoration from AWS S3 Deep Glacier to AWS S3. [11/7/2020 1:15 PM CST]

make sure to create the Boto3 AWS S3 object(s) for the minimal number of times, without any redundancy. [11/7/2020 4:00 PM CST]

(ok, I have no objection to the above software implementation design. I'll code ~~using~~ the above software implementation design, if and when I feel like it. [11/7/2020 4:08 PM CST])

~~resolve and document the restoration AWS S3 query issue. for restoration, simply going back six months isn't enough, I think. I am leaning toward using three additional arguments in the data-restoration AAM Auto Backup command: AWS S3 archive object query starting year and month (two separate arguments), and the number of months to backtrack at the maximum (36 maximum—if smaller than 1 or bigger than 36, error display and quit). [11/10/2020 12:07 PM CST] (UPDATE [11/10/2020 12:22 PM CST] I'm very heavily leaning toward using those additional three command arguments. in fact, I'll do it.) (UPDATE [11/10/2020 12:50 PM CST] this is useless. because the backup AWS S3 bucket objects get auto-deleted after six months. I'm not implementing these additional arguments.)~~

13.2. restoring from Glacier to AWS S3

[11/7/2020 10:35 AM CST]
search "AWS boto3 restore_object" online.
https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html#S3.Client.restore_object
https://github.com/boto/boto3/issues/1422
https://github.com/boto/boto3/issues/1435
https://stackoverflow.com/questions/31625224/boto-finding-out-when-to-use-restore-to-restore-object-from-glacier-back-to-s3

14. Python waiting for a user input

[11/12/2020 2:07 PM CST]
search "python wait for user input" online.
https://stackoverflow.com/questions/983354/how-do-i-make-python-wait-for-a-pressed-key

In Python 3 use input():
input("Press Enter to continue...")

design and implement the user key-press waiting. [11/12/2020 2:11 PM CST]
think about where to place the user key-press waiting before sys.exit(). document everything. [11/12/2020 2:11 PM CST]
implement (code) the user key-press waiting. [11/12/2020 2:11 PM CST]

[11/12/2020 6:46 PM CST]
AAM Auto Backup user-input waiting (for AAM Auto Backup execution by shortcuts)
Every time before the software program end, for incremental-only backup and restoration only? Very likely so.
Probably also for regular backup when the force full backup is on.
FINAL DECISION use a single class data variable for controlling the user-input waiting. For making easy changes.
UPDATE how about using a command argument named "wait for user input before closing"? I like that the best yet?
(FINAL DECISION [11/13/2020 9:16 AM CST] since I have no objection to the above, I'll do the above.)

VII. Test Cases

1. test case 1

1.1. overview

in c:\temp\test2\, one or more large media files to go over the 4Gbyte to step-through all the code.

1.2. backup instruction plain text file contents

[10/24/2020 8:00 PM CST] [12/11/2020 4:05 PM CST]

```
c:\users\allen\documents\folder1\folder1-2\
c:\users\allen\documents\folder1\folder1-4\
c:\users\allen\documents\folder2\
c:\users\allen\documents\folder2\folder2-1\
c:\users\allen\documents\folder2\folder2-1-1\
c:\users\allen\documents\folder3\
C:\Users\Allen\AAM posting\stock linked files\
c:\temp\test\
c:\temp\test2\

c:\users\allen\documents\folder3\

~WRL*.tmp
```

1.3. incremental backup command

[12/11/2020 1:52 PM CST]

```
AAM_auto_backup.py incremental-only-backup ...
```

[12/24/2020 12:05 PM CST] Visual Studio Code launch.json entry

```
"args" : ["incremental-only-backup", "user", "C:\\Users\\Allen\\
AppData\\Local\\Programs\\Python\\Python38\\python.exe", "600",
"7200", "C:\\Users\\Allen\\Documents\\AAM Auto Backup\\logs",
"True", "AAM Backup Report", "autoreporter@allenyoung.dev",
"allenyoung@allenyoung.dev", "C:\\Users\\Allen\\Documents\\AAM
Auto Backup\\instructions\\
main_computer_backup_instructions.txt", "q", "local^|AWS", "F:\\
backup\\main", "False", "1234abcd1234abcd", "aam-data-backup",
"main-work-computer", "AWS-USER-ID", "aws-user-password", "no",
"yes"]
```

1.4. regular backup command

[12/14/2020 11:47 AM CST]

```
AAM_auto_backup.py regular-backup user "C:\Users\Allen\AppData\
Local\Programs\Python\Python38\python.exe" 600 7200 "C:\Users\
Allen\Documents\AAM Auto Backup\logs" True "AAM Backup Report"
```

```
"autoreporter@allenyoung.dev" "allenyoung@allenyoung.dev" "C:\
Users\Allen\Documents\AAM Auto Backup\instructions\
main_computer_backup_instructions.txt" "q" "local|AWS" "F:\
backup\main" False "1234abcd1234abcd" "aam-data-backup" "main-
work-computer" "AWS-USER-ID" "aws-user-password" "no" "yes"
```

[12/15/2020 11:08 AM CST] Visual Studio Code launch.json entry

```
"args" : ["regular-backup", "user", "C:\\Users\\Allen\\AppData\\
Local\\Programs\\Python\\Python38\\python.exe", "600", "7200",
"C:\\Users\\Allen\\Documents\\AAM Auto Backup\\logs", "True",
"AAM Backup Report", "autoreporter@allenyoung.dev",
"allenyoung@allenyoung.dev", "C:\\Users\\Allen\\Documents\\AAM
Auto Backup\\instructions\\
main_computer_backup_instructions.txt", "q", "local^|AWS", "F:\\
backup\\main", "False", "1234abcd1234abcd", "aam-data-backup",
"main-work-computer", "AWS-USER-ID", "aws-user-password", "no",
"yes"]
```

[12/15/2020 11:19 AM CST] search "JSON escape |" online.
(nothing useful.)

[12/15/2020 11:24 AM CST] search "Python sys.argv escape |" online.
https://stackoverflow.com/questions/9590838/python-escape-special-characters-in-sys-argv

[12/15/2020 11:26 AM CST] search "Visual Studio Code launch.json escape |" online.
(nothing useful.)

[12/15/2020 11:28 AM CST] search "Visual Studio Code launch.json escape vertical bar" online.
(nothing useful.)

[12/15/2020 11:33 AM CST] search "Python sys.argv escape vertical bar" online.
https://mail.python.org/pipermail/tutor/2009-June/070112.html
As far as I know, the vertical bar isn't a special character in Python
at all. It is, however, special in CMD.EXE, the command interpreter for
recent Windows versions. It is used to specify multiple programs on the
same command line, where the stdout of the first is piped into the stdin
of the next. Similarly, the > and < symbols are used to specify
redirection, and the & symbol is used to separate two commands that are
to be run sequentially.
in cmd.exe, Windows, | is a character for separating different commands to execute.
[12/15/2020 11:44 AM CST] try putting | in a quoted string. try "\"local|AWS\"".
result: "is not recognized as an internal or external command,
operable program or batch file." (doesn't work. "local||AWS" doesn't work either.)

[12/15/2020 11:56 AM CST] search "Windows terminal escape vertical bar" online.
https://stackoverflow.com/questions/12644837/how-can-i-escape-the-vertical-bar-when-doing-command-line-with-ruby

https://docs.microsoft.com/en-us/windows/console/console-virtual-terminal-sequences
The ampersand (&), pipe (|), and parentheses ( ) are special characters that must be preceded by the escape character (^) or quotation marks when you pass them as arguments.
[12/15/2020 11:59 AM CST] try "local^|AWS". (ok, it works now!)

1.5. shadow-copy commands

1.5.1. regular-backup shadow-copy command

[12/16/2020 6:13 PM CST] shadow-copy handling code test procedure
to step-through the ShadowSpawn call processing code of AAM Auto Backup, create shadow_copy_instruction.txt with local directory paths, step-through AAM Auto Backup in Visual Studio Code with the right command arguments in launch.json. ~~while stepping-through, update the applicable values to change the shadow-mounted directory paths to local directory paths.~~ copy, paste, and save the Visual Studio Code terminal output in a separate plain text file. verify the outputs after the step-through completion.

[12/14/2020 11:47 AM CST]

```
ShadowSpawn.exe "c:\\temp" q: "C:\\Users\\Allen\\AppData\\Local\\
Programs\\Python\\Python38\\python.exe" AAM_auto_backup.py
regular-backup ShadowSpawn " " " " "20201216 170302.225333" "c:\\
temp" "q" "local | AWS" "F:\\backup\\main" "False"
"1234abcd1234abcd" "main-work-computer" "c:\\temp\\main-work-
computer20201216_170302.225333" "no" >
"shadow_copy_command_output.txt"
```

[12/16/2020 5:17 PM CST] Visual Studio Code launch.json entry, full-backup

```
"args" : ["regular-backup", "ShadowSpawn", " ", " ", "20201216
170302.225333", "c:\\temp", "q", "local ^| AWS", "F:\\backup\\
main", "False", "1234abcd1234abcd", "main-work-computer", "c:\\
temp\\main-work-computer20201216_170302.225333", "no", "full-
backup"]
```

```
"args" : ["regular-backup", "ShadowSpawn", "20201221
134337.851361", "20201221 134337.851361", "20201222
112502.225333", "c:\\temp", "q", "local ^| AWS", "F:\\backup\\
main", "False", "1234abcd1234abcd", "main-work-computer", "c:\\
temp\\main-work-computer20201222_112502.225333", "no",
"incremental-backup"]
```

1.5.2. incremental-only backup shadow-copy command

[12/22/2020 11:05 AM CST]

[12/22/2020 11:07 AM CST] Visual Studio Code launch.json entry

```
"args" : ["incremental-only-backup", "ShadowSpawn", " ", " ",
"20201216 170302.225333", "c:\\temp", "q", "local ^| AWS", "F:\\
backup\\main", "False", "1234abcd1234abcd", "main-work-computer",
"c:\\temp\\main-work-computer20201216_170302.225333", "no",
"incremental-backup"]
```

1.6. one-step data restoration command

[12/14/2020 12:49 PM CST]

```
AAM_auto_backup.py one-step-aws-data-restoration "C:\Users\Allen\
Documents\AAM Auto Backup\logs" True "AAM Backup Report"
"autoreporter@allenyoung.dev" "allenyoung@allenyoung.dev" "2"
"1234abcd1234abcd" "aam-data-backup" "main-work-computer"
"(restoration AWS IAM user access key ID)" "(restoration AWS IAM
user secret access key)" "c:\temp\restored_data"
```

[12/15/2020 11:08 AM CST] Visual Studio Code launch.json entry

```
"args" : ["one-step-aws-data-restoration", "C:\\Users\\Allen\\
Documents\\AAM Auto Backup\\logs", "True", "AAM Backup Report",
"autoreporter@allenyoung.dev", "allenyoung@allenyoung.dev", "2",
"1234abcd1234abcd", "aam-data-backup", "main-work-computer",
"(restoration AWS IAM user access key ID)", "(restoration AWS IAM
user secret access key)", "c:\\temp\\restored_data"]
```

1.7. multi-step data restoration commands

[12/14/2020 12:54 PM CST]

```
AAM_auto_backup.py restore-to-aws-s3-from-aws-s3-glacier-deep-
archive ...

AAM_auto_backup.py check-aws-s3-restoration-status ...

AAM_auto_backup.py download-to-local-computer-from-aws-s3 ...

AAM_auto_backup.py decompress-all-downloaded-backup-files ...
```

1.8. result

2. test case 2

2.1. overview

[12/10/2020 7:45 PM CST]
no step-through.

set up AAM Auto Backup on Windows Task Scheduler. let Windows Task Scheduler run at the scheduled time.

check the backup report email.

check the backup report log.

using folder properties in Windows, count the files copied by Robocopy. compare the count to the backup source counts.

on the Windows 10 computer, execute the multi-step AWS S3 backup data restoration. count the restored files. compare the count to the backup source counts.

on the Windows 10 computer, execute the one-step AWS S3 backup data restoration. count the restored files. compare the count to the backup source counts.

2.2. backup instruction plain text file contents

[12/11/2020 4:08 PM CST]

2.3. incremental backup command

[12/14/2020 11:47 AM CST]

2.4. regular backup command

[12/14/2020 11:47 AM CST]

```
AAM_auto_backup.py ...
```

2.5. one-step data restoration command

2.6. multi-step data restoration commands

2.7. result

VIII. Debugging

[1/5/2021 1:18 PM CST]

1. 'utf-8' codec can't decode byte 0x9a in position 3390: invalid start byte

File I/O error. ShadowSpawn.exe command output file reading error.
'utf-8' codec can't decode byte 0x9a in position 3390: invalid start byte

search "Python 'utf-8' codec can't decode byte 0x9a in position invalid start byte" online.
https://stackoverflow.com/questions/47098435/unicodedecodeerror-utf8-codec-cant-decode-byte-0x9a-in-position-12

search "what is the encoding of Windows terminal > output text file?" online.
https://stackoverflow.com/questions/16598785/save-text-file-in-utf-8-encoding-using-cmd-exe
The default encoding for command prompt is Windows-1252.Oct 11, 2016

The default encoding for command prompt is Windows-1252. Change the code page (chcp command) to 65001 (UTF-8) first and then run your command.

```
chcp 65001
C:\Windows\system32\ipconfig /all >> output.log
```

Change it back to default when done.

```
chcp 1252
```

Here are some other code that may be useful – Lord ST Oct 31 '15 at 21:55
1252 might not be the default codepage. The OEM codepage is the default. – Anders Feb 11 '17 at 14:55

As the existing answer says, in a batch file, you can use the chcp command

```
chcp 65001 > nul
some_command > file
```

But if you are using the cmd.exe from its command line, e.g. to execute a user-defined command, you can use this syntax:

```
cmd.exe /c chcp 65001 > nul & cmd.exe /c some_command > file
```

search "what is ampersand in windows terminal" online.

https://superuser.com/questions/345602/what-is-cmds-equivalent-to-bashs-ampersand-for-running-a-command-without-w
The ampersand lets you launch the first program, and then do another one while the first is still running.Oct 12, 2011

https://htipe.wordpress.com/2009/07/19/multiple-commands-on-a-single-windows-command-line/
Using a single ampersand (&) will cause the first command and then the second command to be run in sequence. Using double ampersands (&&) ...

[1/5/2021 1:22 PM CST] debugging technique
quickly write a separate debugging Python program to process the problematic shadow_copy_command_output.txt. no need to go through the four-hour full backup process.
UPDATE [1/5/2021 1:39 PM CST] the above is not necessary and shouldn't be used. because I'll be using chcp.

[1/5/2021 4:01 PM CST] the same error. try not using chcp, and specifying the Windows-1252 encoding in open(). don't try creating a batch file that executes chcp and shadowspawn.exe. the extra coding isn't justified.

search "Python open encoding Windows-1252" online.
https://stackoverflow.com/questions/15502619/correctly-reading-text-from-windows-1252cp1252-file-in-python

search "python convert string encoding" online.
https://stackoverflow.com/questions/4299802/python-convert-string-from-utf-8-to-latin-1

search "python convert string encoding to utf-8" online.
https://stackoverflow.com/questions/4182603/how-to-convert-a-string-to-utf-8-in-python

TO DO [1/5/2021 9:17 PM CST]
quickly write a separate debugging Python program to process the problematic shadow_copy_command_output.txt.

debug the following.

File I/O error. ShadowSpawn.exe command output file reading error.
'charmap' codec can't decode byte 0x9a in position 3390: character maps to <undefined>

[1/6/2021 1:20 PM CST]
search "Python convert cp1257 string to utf-8 string" online.
https://stackoverflow.com/questions/4182603/how-to-convert-a-string-to-utf-8-in-python

```
    return codecs.charmap_decode(input,self.errors,decoding_table)[0]
UnicodeDecodeError: 'charmap' codec can't decode byte 0x81 in position 7265: character maps to <undefined>
```

[1/6/2021 1:24 PM CST]
search "Python UnicodeDecodeError: 'charmap' codec can't decode byte 0x81 in position character maps to <undefined>" online.
https://stackoverflow.com/questions/9233027/unicodedecodeerror-charmap-codec-cant-decode-byte-x-in-position-y-character

[1/6/2021 1:29 PM CST]
search "Windows terminal get encoding" online.
https://superuser.com/questions/1170656/windows-10-terminal-encoding
https://stackoverflow.com/questions/1259084/what-encoding-code-page-is-cmd-exe-using
chcp
https://docs.microsoft.com/en-us/windows/win32/intl/code-page-identifiers?redirectedfrom=MSDN
https://docs.microsoft.com/en-us/windows/win32/intl/code-pages?redirectedfrom=MSDN

[1/6/2021 1:30 PM CST]
search "Python Windows terminal Active code page 437 encoding" online.
https://stackoverflow.com/questions/6344853/python-unicode-in-windows-terminal-encoding-used

2. long-path file location error

[1/7/2021 4:15 AM CST] debugging technique write a very simple debugging Python CLI program that checks a file with previous existance check failure.

[1/7/2021 4:14 AM CST]
search "Python check Windows file using short path" online.
https://stackoverflow.com/questions/23598289/how-to-get-windows-short-file-name-in-python
https://medium.com/@ageitgey/python-3-quick-tip-the-easy-way-to-deal-with-file-paths-on-windows-mac-and-linux-11a072b58d5f
https://stackoverflow.com/questions/23598289/how-to-get-windows-short-file-name-in-python

[1/7/2021 4:20 AM CST]
search "Python Windows file check longer than max path length" online.
https://stackoverflow.com/questions/46113181/how-to-know-from-python-if-windows-path-limit-has-been-removed
https://bugs.python.org/issue18199
https://www.xspdf.com/resolution/55558810.html

[1/7/2021 4:39 AM CST] using the "\\\\?\\" prefix on Windows works.

[1/11/2021 4:31 PM CST]
File I/O error during local backup-destination validation check.
[WinError 3] The system cannot find the path specified: 'q:\\asian-american man\\AAM products\\02-03 Unified Humanity Science Womanhood Manhood\\-- Womanhood (old)\\posts\\AD 2019\\Month 01\\posting schedule\\20190902 (Intellectual Womanhood) Is it ok for a rich and successful man to cheat on his spouse video post description.txt'
No backup batch file execution success message. Aborting backup.
Critical error. ShadowSpawn.exe execution failure Aborting regular backup opera
tion.

[1/11/2021 6:52 PM CST]
File I/O error during local backup-destination validation check.
[WinError 3] The system cannot find the path specified: 'q:\\asian-american man\
\AAM products\\02-03 Unified Humanity Science Womanhood Manhood\\-- Womanhood (o
ld)\\posts\\AD 2019\\Month 01\\posting schedule\\20190902 (Intellectual Womanhoo
d) Is it ok for a rich and successful man to cheat on his spouse video post desc
ription.txt'
No backup batch file execution success message. Aborting backup.
Critical error. ShadowSpawn.exe execution failure Aborting regular backup opera
tion.

3. "You must specify a region" and email-client NoneType error

[1/12/2021 6:08 AM CST]
search "boto3.client how to specify region" online.
https://stackoverflow.com/questions/40377662/boto3-client-noregionerror-you-must-specify-a-region-error-only-sometimes

search "how to specify region in aws boto3.Session" online.

4. RestoreAlreadyInProgress error

[1/12/2021 6:31 AM CST]
search "Python how to handle multiple errors" online.

search "boto3 RestoreAlreadyInProgress error" online.
https://stackoverflow.com/questions/46174385/properly-catch-boto3-errors

search "Python how to resume after error" online.
https://stackoverflow.com/questions/26059424/on-error-resume-next-in-python
https://stackoverflow.com/questions/18655481/a-pythonic-way-for-resume-next-on-exceptions
https://stackoverflow.com/questions/38707513/ignoring-an-error-message-to-continue-with-the-loop-in-python
https://stackoverflow.com/questions/1524216/how-to-have-an-error-but-continue-the-script-in-python
https://stackoverflow.com/questions/18994334/make-python-code-continue-after-exception
https://stackoverflow.com/questions/730764/how-to-properly-ignore-exceptions

search "AWS boto3 how to handle RestoreAlreadyInProgress error" online.
https://stackoverflow.com/questions/33068055/how-to-handle-errors-with-boto3

5. 7z list file writing error.

[1/12/2021 7:31 AM CST]

```
7z list file writing error.
C:\Users\Allen\Documents\
\AAM Auto Backup\source code\20210112 072018.063753_inclusion_7z_list_file.txt
[WinError 3] The system cannot find the path specified: 'q:\\asian-american man\\AAM products\\02-03 Unified Humanity Science Womanhood Manhood\\-- Womanhood (old)\\posts\\
```

AD 2019\\Month 01\\posting schedule\\20190902 (Intellectual Womanhood) Is it ok for a rich and successful man to cheat on his spouse video post description.txt'
No backup batch file execution success message. Aborting backup.
No Robocopy execution success message. Aborting backup.
No 7z.exe execution success message. Aborting backup.
Critical error. ShadowSpawn.exe execution failure Aborting regular backup opera
tion.

this error has been fixed by appending //?/ (for Windows long paths longer than 260 characters) at the start of file path in applicable places.

6. Windows Task Scheduler task keeps getting disabled

[1/18/2021 5:26 AM CST]
search "windows 7 task scheduler task keeps getting disabled" online.
https://superuser.com/questions/808526/task-scheduler-task-keeps-becoming-disabled

A Tom's Hardware thread (forums.tomshardware.com/threads/…) suggests the culprit may be Avast's Game Mode (now Do Not Disturb mode). See techdows.com/2017/03/disable-avast-game-mode.html

https://techdows.com/2017/03/disable-avast-game-mode.html

https://www.thegreenbutton.tv/forums/viewtopic.php?t=11297

https://forum.avast.com/index.php?topic=209525.0

https://forums.tomshardware.com/threads/scheduled-tasks-keep-disabling.3143959/

[1/18/2021 5:39 AM CST] I just turned off everything in the Avast "Do Not Disturb Mode". no "Game Mode" in my Avast.
[1/18/2021 5:41 AM CST] it looks like keeping all the options on, then adding AAM Auto Backup to the "Don't block notification from these apps" list in Avast might work. I won't try it. I'll just keep everything in "Do Not Disturb Mode" off.

[1/18/2021 6:03 AM CST] "Mozilla Firefox", "Google Chrome" and "Dimmer" are set to "Do not disturb" in Avast. Running them must have made Avast auto-disable the "Daily AAM Auto Backup" and other tasks in Windows Task Scheduler. After disabling all the "Do Not Disturb Mode" settings in Avast, after running "Mozilla Firefox" and "Google Chrome", the "Daily AAM Auto Backup" task in Windows Task Scheduler stays enabled. After rebooting the computer, the "Daily AAM Auto Backup" task in Windows Task Scheduler is still enabled; after running "Mozilla Firefox" and "Google Chrome" again, the "Daily AAM Auto Backup" task in Windows Task Scheduler stays enabled. the problem fixed now.
(TO DO [1/18/2021 6:05 AM CST] after the current "Daily AAM Auto Backup" task manual execution gets completed, verify the above!)

(UPDATE [1/18/2021 9:55 AM CST] ok, I verified the above. I think the problem has been completely solved! [great! so damn great!!!! I'm precisely on the right track! I'll simply keep on executing and improving all my plans, no matter what, no matter what, no matter what, no matter what, NO MATTER WHAT!!!!!!!])

7. Robocopy validation failure

[1/19/2021 9:16 AM CST]

Robocopy failure. Mismatching file modification date times.

strFileFullPath: q:\AAM Auto Backup\source code\shadow_copy_command_output.txt
datetimeFileModification: 2021-01-19 00:33:19.842746

strDestinationFileFullPath: \\?\F:\backup\main\c\users\allen\documents\AAM Auto Backup\source code\shadow_copy_command_output.txt
datetimeDestinationFileModification: 2021-01-19 00:33:17.814742

think about why the above happens. why the two second difference?

[1/19/2021 1:35 PM CST]
The only reason I can think of is that shadow_copy_command_output.txt file is opened during Shadowspawn execution. Maybe that updated the file modification time which is reflected in the file in the shadow mount. So, shadow mount content can be updated while the mount is active. The only solution I can think of is excluding shadow_copy_command_output.txt and the file lock file.

[1/19/2021 1:38 PM CST] how about instead of testing for modification-time equality, testing datetimeFileModification >= datetimeDestinationFileModification for test pass, or datetimeFileModification < datetimeDestinationFileModification for test failure? I'm leaning toward implementing this.

[1/19/2021 2:02 PM CST] using datetimeFileModification < datetimeDestinationFileModification for file copy failure check would not do any good, since that could include when copy was not done at all and the old file was not updated.

since the copy of a file at the time of the shadow mount initiation is copied to the destination, datetimeFileModification < datetimeShadowMountInitiation should be used for the test failure check.

I think the above is the most appropriate solution. I'll think and decide on it, and then implement the decision!

or (datetimeDestinationFileModification != datetimeFileModification and datetimeFileModification < datetimeBackupExecutionStartDateTime) will do too! implementing this would require the least code modification/update! I'm leaning toward implementing this one!!!! I think this would be the most elegant and proper solution!!!!!!

[1/19/2021 2:21 PM CST]

```
    self._datetimeBackupExecutionStart = datetime.datetime.now()
    self._strBackupExecutionStartDateTime = \
      str(self._datetimeBackupExecutionStart)
    self._strCommandArgumentBackupExecutionStartDateTime = \
      self._strBackupExecutionStartDateTime.replace('-', '')
    self._strCommandArgumentBackupExecutionStartDateTime = \
      self._strCommandArgumentBackupExecutionStartDateTime.replace(':', '')

        datetimeLastIncrementalAwsS3Backup = \
          datetime.datetime.strptime( \
          self._aam_abl.strLastAwsS3IncrementalBackupDateAndTime,
          "%Y%m%d %H%M%S.%f")
```

self._strCommandArgumentBackupExecutionStartDateTime is passed in the shadowspawn AAM Auto Backup command.

self._strCommandArgumentBackupExecutionStartDateTime needs to be converted back into a datetime-type value.

test the following code in Python IDLE

```
import datetime

datetimeBackupExecutionStart = datetime.datetime.now()

strBackupExecutionStartDateTime =
str(datetimeBackupExecutionStart)

strCommandArgumentBackupExecutionStartDateTime =
strBackupExecutionStartDateTime.replace('-', '')

strCommandArgumentBackupExecutionStartDateTime =
strCommandArgumentBackupExecutionStartDateTime.replace(':', '')

datetimeCommandArgumentBackupExecutionStartDateTime =
datetime.datetime.strptime(strCommandArgumentBackupExecutionStart
DateTime, "%Y%m%d %H%M%S.%f")

print(datetimeCommandArgumentBackupExecutionStartDateTime)
```

[1/19/2021 2:30 PM CST] ok, the above code works!

[1/19/2021 2:38 PM CST] ok, I implemented the above fix/solution. I'm executing a regular backup now to test it.
RESULT [1/19/2021 3:18 PM CST] it passed.

[1/19/2021 3:00 PM CST] TO DO
move the "#Compare the file sizes." section to into the "#Compare the file modification date times." section in an else clause. perform the file size check only if datetimeFileModification <= datetimeBackupExecutionStartDateTime.
[1/19/2021 3:18 PM CST] I updated the code. I'm executing a regular backup now to test it.
RESULT [1/19/2021 3:51 PM CST] no error. now, I think AAM Auto Backup v1.0 development is fully complete! finally!

8. ShadowSpawn.exe execution error [5/27/2023 12:07 PM CST]

Search "There was an error calling CreateProcess. Process: dir Error: The system cannot find the file specified." online.
https://stackoverflow.com/questions/19621838/createprocess-error-2-the-system-cannot-find-the-file-specified
https://www.easeus.com/computer-instruction/the-system-cannot-find-the-file-specified.html

[5/27/2023 12:34 PM CST] FIXED
The error was due to the outdated and deleted Python executable full path in "Incremental-only backup.bat" and "Regular backup.bat".

IX. Appendix: Updates

1. [1/12/2021 8:59 AM CST]

AAM Auto Backup update now
Display and log each backup file uploaded to AWS S3, before the upload start of each file.
([1/12/2021 8:59 AM CST] for debugging.)

(DONE. [1/12/2021 9:08 AM CST])

2. [10/26/2020 2:22 PM CST]

this update is for incorporating the AAM Auto Backup design change, documented in "backup AWS S3 bucket query processing" above.

update process spec
1. update process spec creation.
    1.1. ~~outline the update process. (DONE. [10/26/2020 6:44 PM CST])~~
    1.2. ~~review all of "backup AWS S3 bucket query processing". (DONE. [10/26/2020 9:00 PM CST])~~
    1.3. ~~resolve the one additional critical issue. [10/26/2020 8:58 PM CST] (one more thing to handle) the local backup destination directory structure—the shadow-mount directory naming in the local backup destination directory. how to do this? (DONE. [10/27/2020 11:24 AM CST])~~
    1.4. create the 'data backup and restoration rules summary' document sector. [10/27/2020 11:22 AM CST]
        1.4.1. ~~outline the document-sector content and put all of the relevant notes in the document sector. (DONE. [10/27/2020 3:26 PM CST])~~
        1.4.2. ~~create the "Data backup and restoration terms" document sector, for writing and finalizing the 'data backup and restoration rules summary' document-sector content. [10/28/2020 2:54 PM CST] (DONE. [10/28/2020 2:54 PM CST])~~
        1.4.3. review and finalize the entire 'data backup and restoration rules summary' document-sector content, from the top to the bottom.
    1.5. write out the "backup AWS S3 bucket query algorithm" document sector. [10/27/2020 2:19 PM CST]
    1.6. review all of the applicable document sectors in the AAM Auto Backup document, and finalize the update process spec below.
2. "AAM Auto Backup Operation Modes and Command Formats" part update.
    2.1. "Regular-backup Operation Mode" updates.
        2.1.1. "operation mode overview" update.
        2.1.2. "command arguments and format" update.
    2.2. "Incremental-only Backup Operation Mode" updates.
        2.2.1. "operation mode overview" update.
    2.3. "One-step AWS S3 Data Restoration Operation Mode" updates.
        2.3.1. "operation mode overview" update.
3. "Software Process Specs" part update.
    3.1. "Regular-backup Process Spec" update.
    3.2. "Incremental-only Backup Process Spec" update.
4. AAM Auto Backup source code update.
    4.1. update the code to incorporate the force AWS S3 full backup command argument.

2.1. ADDITIONAL CONSIDERATIONS

[10/27/2020 10:54 AM CST]

AAM Auto Backup document
I may need a 'data backup and restoration rules summary' document sector.
"AAM Auto Backup adheres to the following rules when performing data backup and restoration."
AAM Auto Backup data backup rules
…

AAM Auto Backup data restoration rules
…
[10/27/2020 11:17 AM CST] maybe place the first chapter titled "Data backup and restoration rules summary" in the "AAM Auto Backup Operation Modes and Command Formats" part?

UPDATE [10/27/2020 11:14 AM CST]
no need to use the short directory path name mapping file: simply use the full directory replication in the backup-destination folder, starting with the drive letter. Robocopy will handle everything needed! check out the additional Robocopy test 2 to find out why!

(I need to first decide on the local data backup path design/rule!)

(Any choice other than full path and a path mapping text file?)

(I need to perform a robocopy test. Create a max-path file in c:\temp, using multiple sub-folders named #a...za...z. try copying that to f:\backup with sub-folder copy.)

(an option is to let the user provide the common parent directory short names in the backup instruction file, but it would be a terrible and error-prone design, because the user does not know what the common parent directories will be, and also because it places an additional burden on the user. This must be avoided.) (for now, I am thinking that the best option is for the AAM Auto Backup software to decide the common parent directory short names in the backup-destination directory. A shadow-mounted directory short name will be a number in a 26-digit numeral system, with the English alphabet letters representing the digits, a being the smallest number and z being the largest number. To convert a decimal number to the 26-digit number, find the remainder when divided by 26, convert the remainder to the 26-digit numeral, subtract the remainder from the decimal number, …[complete this]. To convert a 26-digit number to the decimal number, … [complete this]. AAM Auto Backup scans each first-level sub-directory in the backup-destination directory to determine the backup destination for each shadow-mounted directory, by trying matching sub-directory names. Actually, this is a very unreliable method, since it can result in false matches--so I won't use it. For now, I think a shadow-mounted or common-parent directory short name mapping text file must be created and used by AAM Auto Backup; if this text file is missing or corrupt, lacks one or more shadow-mounted directories, includes additional shadow-mounted directory[ies], AAM Auto Backup creates and uses needed shadow-mounted directory short name[s] [that do not overlap with existing first-level sub-directory names in the backup-destination directory], and appends the short names to the shadow-mounted or common-parent directory short name mapping text file. For now, I think using the 26-digit or 10-digit numeral system, and the shadow-mounted or common-parent directory short name mapping text file, is the best way to go. I will have to specify a method for processing the directory short name mapping text file content, especially error handling--the entire file content will be replaced if there's an error in the text file, and new short directory names that do not overlap with the existing first-level sub-directory names in the backup-destination directory will be used; that is, AAM Auto Backup does not delete the existing folders and their contents in the backup-destination folder. I will still perform the robocopy test with a max-path file, to find out what happens.)

https://en.wikipedia.org/wiki/Numeral_system
https://en.wikipedia.org/wiki/Numerical_digit

2.2. Additional Robocopy tests

2.2.1. test 1

in c:\temp1\, create the following sub-directories.

1abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
3abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
4abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
~~5abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz~~

create a text file in the last sub-directory with the longest name possible.
abcdefghijklmnopqrstuvwxyzabcde.txt

C:\Temp\1abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
3abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
4abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
abcdefghijklmnopqrstuvwxyzabcde.txt

execute the following Robocopy command.

```
robocopy C:\temp1 f:\backup /e
```

2.2.1.1. Robocopy command result

```
C:\Users\Allen>robocopy C:\temp1 f:\backup /e

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 27 11:04:29 2020

   Source : C:\temp1\
     Dest : f:\backup\

    Files : *.*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30
```

```
------------------------------------------------------------------------------

                      0    C:\temp1\
      *EXTRA Dir       -1    f:\backup\Custom Linux KVM VM images\
        New Dir        0    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\
        New Dir        0    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
        New Dir        0    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\3abcdefghij
klmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
        New Dir        1    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\3abcdefghij
klmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\4abcdefghijklmnopqrstuvwxyzabcdefghij
klmnopqrstuvwxyz\
          New File              0        abcdefghijklmnopqrstuvwxyzabcd.t
100%

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         5         4         1         0         0         1
   Files :         1         1         0         0         0         0
   Bytes :         0         0         0         0         0         0
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Tue Oct 27 11:04:30 2020

C:\Users\Allen>
```

[10/27/2020 11:06 AM CST] the file got copied. why did that happen?

[10/27/2020 11:06 AM CST] does this mean I can use the full path of the source in the backup destination directory?

2.2.2. test 2

in c:\temp1\, create the following sub-directories.

1abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
3abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
4abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz
~~5abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz~~

create a text file in the last sub-directory with the longest name possible.
abcdefghijklmnopqrstuvwxyzabcde.txt

C:\Temp\1abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
3abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
4abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
abcdefghijklmnopqrstuvwxyzabcde.txt

create the following directory.
f:\abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz

execute the following Robocopy command.
```
robocopy C:\temp1 f:\
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz /e
```

2.2.2.1. Robocopy command result

```
C:\Users\Allen>robocopy C:\temp1 f:\abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqr
stuvwxyz /e

-------------------------------------------------------------------------------
   ROBOCOPY     ::     Robust File Copy for Windows

-------------------------------------------------------------------------------

  Started : Tue Oct 27 11:09:07 2020

   Source : C:\temp1\
     Dest : f:\abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\

    Files : *.*

  Options : *.* /S /E /COPY:DAT /R:1000000 /W:30

------------------------------------------------------------------------------

                           0    C:\temp1\
          New Dir          0    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\
          New Dir          0    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
          New Dir          0    C:\temp1\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\3abcdefghij
```

```
klmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
        New Dir          1    C:\temp1\\1abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz\2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\3abcdefghij
klmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\4abcdefghijklmnopqrstuvwxyzabcdefghij
klmnopqrstuvwxyz\
          New File                  0        abcdefghijklmnopqrstuvwxyzabcd.t
100%

------------------------------------------------------------------------------

               Total    Copied   Skipped  Mismatch    FAILED    Extras
    Dirs :         5         4         1         0         0         0
   Files :         1         1         0         0         0         0
   Bytes :         0         0         0         0         0         0
   Times :   0:00:00   0:00:00                       0:00:00   0:00:00

   Ended : Tue Oct 27 11:09:07 2020

C:\Users\Allen>
```

[10/27/2020 11:09 AM CST]
F:\abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
1abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
2abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
3abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
4abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz\
abcdefghijklmnopqrstuvwxyzabc.txt

[10/27/2020 11:11 AM CST] copying and pasting the above directory path from Windows Explorer produces the following.
F:\
ABCDEF~1\1ABCDE~1\2ABCDE~1\3ABCDE~1\4abcdefghijklmnopqrstuvwxyzabcdefghijkl
mnopqrstuvwxyz

it looks like Robocopy somehow does something with the short directory name, while still conserving long directory name.

this changes the design.

no need to use the short directory path name mapping file. Robocopy will handle everything needed!

3. [10/19/2020 11:19 AM CST]

think and decide on skipping the internet connection check entirely. [10/18/2020 6:46 PM CST] think and decide on always using the backup AWS S3 bucket object querying instead of using the last AWS S3 backup dates and times plain text file; or, using the backup AWS S3 bucket object querying in addition to check the internet and AWS S3 connection, and to verify the backup AWS S3 full backup objects presence and completeness (if the full-backup is missing on AWS S3, force-perform another full backup, and report the status in terminal, log, and email report). uploading files to AWS S3 bucket costs a very small amount money anyway due to the PUT or POST requests; adding one or just a few LIST requests will cost US$0.01 or less per month. [10/18/2020 7:01 PM CST] (UPDATE [10/19/2020 8:36 AM CST] I am very heavily leaning toward doing the AWS S3 check before running 7z.exe, since especially for full backup, producing all the backup zip files will be a tremendous waste when there no is internet and/or AWS S3 connection. also, note that deciding doing a full backup must be decided before executing 7z.exe. think and decide on whether to abandon using the last AWS S3 backup dates and times plain text file entirely, and exclusively rely on the backup AWS S3 bucket objects querying, or to use the last AWS S3 backup dates and times plain text file for verification purposes. I'm leaning toward not using the last AWS S3 backup dates and times plain text file, because there really is no reason to do additional coding work. think and decide on this, and then document and implement the decision!!!!!) (NOTE [10/19/2020 10:54 AM CST] I think less coding is better coding, and the internet and AWS S3 connections must be checked before generating the backup ZIP files. so, in all likely chance, I won't use the last AWS S3 backup dates and times plain text file, although that means paying a very small amount of money to AWS per month, US$0.01 or less per month. **yeah, in all likely chance, I won't use the last AWS S3 backup dates and times plain text file.**) (UPDATE [10/19/2020 11:06 AM CST] note that for the incremental-only backup, checking for the full-backup AWS S3 objects needs to be done only six times at the maximum, because all the backup AWS S3 bucket objects are auto-deleted after 180 days. document this, and use this.) (FINAL DECISION [10/19/2020 11:11 AM CST] **no last AWS S3 backup dates and times plain text file use.**)
(UPDATE [10/20/2020 7:45 AM CST] no objection to not using the last AWS S3 backup dates and times plain text file. so, I will go ahead and remove the use of it in the design document.)

decide on creating the log after sending email. also using self._bEmailSendingFailure in _SendEmailReport(). to include email sending error in the log file. in all likely chance, I'll do this. [10/18/2020 9:24 PM CST]

think and decide on not including the ShadowSpawn execution log in the email reports. no good will ever come out of emailing all the ShadowSpawn.exe, Robocopy and 7z.exe outputs in unencrypted emails. also, for the full backups, the Robocopy and maybe 7z.exe outputs will be too big to email—so emailing those outputs should be avoided. I'm very heavily leaning toward including the ShadowSpawn execution log in the AAM Auto Backup log only. think and decide on this, document the decision, and implement the decision!!!! [10/19/2020 8:30 AM CST]

document creating the backup ZIP files in c:\temp to use less characters in the backup ZIP file full path. since the backup ZIP file names are long, it is prudent to use a short directory name. if c:\temp doesn't exist, create it in code.

update "Regular-backup Process Spec". it is not entirely correct now: redundant entries on checking the last AWS S3 backup dates and times plain text file content. [10/18/2020 6:46 PM CST]