PeopleSoft File Hash Generator

Design and Support Overview

Developer: Tom Lux

Contents

[Interface Purpose and Overview 2](#_Toc504985688)

[Parameters 2](#_Toc504985689)

[Job-Name Param Requirements 2](#_Toc504985690)

[Design Detail – Alternate Command Server 3](#_Toc504985691)

[Tidal Job Setup 3](#_Toc504985692)

[Console Output Example (time-stamped entries are written to log file) 3](#_Toc504985693)

[Manual Execution (Command Line) 4](#_Toc504985694)

# Interface Purpose and Overview

To help support SOX compliance surrounding PeopleSoft data files, we need a way to ensure the files generated by IRMA are not modified before they are imported into Peoplesoft – the solution is this hash-generator interface (a PowerShell script). We add a Tidal job that generates a hash-value against each PeopleSoft data file, immediately after each PeopleSoft job for each region. These hash-values can then be validated before the data from any file is imported into PeopleSoft.

This interface only ***generates*** hash values, so the intended use is for hash values to also be generated by a consumer or verification process (Finance/PeopleSoft Team) using the same hash algorithm (MD5) and if the IRMA hash value and the verification value match, we know the PeopleSoft.

This interface stores the generated hash values in a separate file with the same name as the Peoplesoft file but with a ".hash" file extention. The Production location is [\\IRMAPrdFile\PsFiles\](file:///\\IRMAPrdFile\PsFiles\); each region has a subfolder.

PowerShell script file name: PeopleSoftFileHashGenerator.ps1

# Parameters

Required:

1. Env – The environment in which you are running: Test, QA, Prd.
2. Region – The region for which you are generating hash values: EU, FL, MA, MW, NA, NE, NC, PN, RM, SO, SP, SW, TS.
3. Job – The name of the job against which the hash generator is running (hash job names jobs should match PS job names with “ hash” appended to job name).

Optional

1. Date – The specific date to process; default date is the current date.

The optional data parameter can be used, in case a day is missed or a run attempt fails.

# Job-Name Param Requirements

We (the script) expect the hash job's name (string) passed in to contain a keyword so that we can determine for which PeopleSoft data file we are generating a hash value. VALID KEYWORDS, MAPPING:

1. "PeopleSoftUpload" -- This job-name keyword identifies the normal upload file (ex: PSFile\_FL\_20160925.EDI)
2. "FINTEC" -- This job-name keyword identifies the FINTECH upload file (ex: PSFINTECH\_FL\_20160925.txt) \*\*NOTE: Sometimes the last "H" is missing from "FINTECH" in filename, so we exclude it from file refs, just in case.
3. "Transfer" -- This job-name keyword identifies the GL upload file (ex: PSGL\_FL\_20160925.txt)

The PeopleSoft-Upload jobs are named with one of the three keywords above, so the hash-gen job for each must have the same keyword.

# Design Detail – Alternate Command Server

IRMA[QA|Prd]File have old version of powershell (v2) and do not meet the requirements for installing powershell 4 or 5, therefore these servers do not have the get-filehash command. To work around this, we can copy the PeopleSoft file to an alternate server in the environment (QA=VM-Icon-QA1, Prod=IRMAPrdFile2), for example, since these are 2012 servers running powershell v4, and then run the hash command from that server against the local file (you cannot invoke a command on a remote server that then references a UNC/remote path, since that's a "double hop").

We use an input file to store the name of the server where we'll execute the hash-generation command. The env param is used to identify the file containing the hash server name.

Because this process could run simultaneously for different regions (it's intended to be triggered by the completion of an IRMA PeopleSoft job that generates a data file), the log file is region-specific.

Not all log entries are written to the log file, but all log entries are output to the console, so additional runtime information could be viewed in the job's output tab in Tidal.

# Tidal Job Setup

This interface is deployed as job instances in Tidal (IRMA’s job group) that run after each of the PeopleSoft jobs (see job dependencies in Tidal). There is a single job for each region named “…PeopleSoft File Hash”. There are two job instances under the RM region’s job group: one for “RM” and one for “TS”.

The job’s name is passed in the ‘Job’ parameter using Tidal’s runtime variable “<JobName>”. This parameter and value must be surrounded in quotes on the command line because the job names contain spaces.

Rerunning the hash-generator jobs will not hurt anything, as they will only generate a hash value for a file if the hash has not already been created (it does not overwrite existing hash-value files, unless they are empty, meaning an error occurred).

To run for a previous day, right-click the regional hash-generator job > insert into schedule. In the parameters window, add a parameter “Date=YYYYMMDD”, where YYYYMMDD is the year, month, and day you need to reprocess.

# Console Output Example (time-stamped entries are written to log file)

PeopleSoft File Hash Generator

>>> Loading functions...

>>> Trying script location...

LOG FILE: [E:\ScheduledJobs\PeopleSoft File Hash Generator\Test\Logs\PeopleSoftFileHashGenerator.ps1.20160731.log]

[2016-07-31 23:37:06] Script Executing: [PeopleSoftFileHashGenerator.ps1] by [Tom.Lux] on [CEWD6557] at [07/31/2016 23:37:06]

Parsing args...

Parameters Found:

Environment: Test

Region: MA

Job Name (Type): MA - DEV - PeopleSoftUpload File Hash (PSFile)

[2016-07-31 23:37:06] Date parameter received.

Date: 20160724

[2016-07-31 23:37:06] Environment=Test, Region=MA, Date=20160724, FileType=PSFile

Identifying hash server...

[2016-07-31 23:37:06] Hash Server: vm-icon-test1 (cewd6592.wfm.pvt)

Checking for local data files to process [e:\Data\PSFiles\MA\PSFile\*20160724\*]...

Identifying data files for which hash values have already been generated...

[2016-07-31 23:37:06] FILE SKIPPED: Hash exists for [PSFile\_MA\_20160724.EDI], value: E8449E0801C4F9AB5EB290D4CAC3E2FA

[2016-07-31 23:37:06] Hash file exists for [PSFINTECH\_MA\_20160724.txt] but is empty so hash will be written.

[2016-07-31 23:37:06] Hash needed for [PSGL\_MA\_20160724.txt]

Removing any data files for target date from hash server...

VERBOSE: Performing operation "Remove File" on Target "\\cewd6592.wfm.pvt\PsFileHash\MA\PSFINTECH\_MA\_20160724.txt".

VERBOSE: Performing operation "Remove File" on Target "\\cewd6592.wfm.pvt\PsFileHash\MA\PSGL\_MA\_20160724.txt".

Copying file(s) to hash server:

Directory: E:\Data\PSFiles\MA

Mode LastWriteTime Length Name

---- ------------- ------ ----

-a--- 7/24/2016 11:31 PM 1322256 PSFINTECH\_MA\_20160724.txt

-a--- 7/24/2016 11:30 PM 154229 PSGL\_MA\_20160724.txt

VERBOSE: Performing operation "Copy File" on Target "Item: E:\Data\PSFiles\MA\PSFINTECH\_MA\_20160724.txt Destination: \\cewd6592.wfm.pvt\PsFileHash\MA\PSFINTECH\_MA\_20160724.txt".

VERBOSE: Performing operation "Copy File" on Target "Item: E:\Data\PSFiles\MA\PSGL\_MA\_20160724.txt Destination: \\cewd6592.wfm.pvt\PsFileHash\MA\PSGL\_MA\_20160724.txt".

[2016-07-31 23:37:06] Generating hash files for (2) target file(s) [\\cewd6592.wfm.pvt\PsFileHash\MA\\*20160724\*]...

[2016-07-31 23:37:07] Writing hash file [e:\Data\PSFiles\MA\PSFINTECH\_MA\_20160724.txt.hash], details: @{Algorithm=MD5; Hash=AAA91F31667B9272DDBF17470C81FE4C; Path=E:\temp\PsFileHash\MA\PSFINTECH\_MA\_20160724.txt; PSComputerName=cewd6592.wfm.pvt; RunspaceId=80cb48fa-1b46-40ac-89a0-9d3b87c239aa; PSShowComputerName=True}

[2016-07-31 23:37:08] Writing hash file [e:\Data\PSFiles\MA\PSGL\_MA\_20160724.txt.hash], details: @{Algorithm=MD5; Hash=2CEF515E954C4956AE0F4B5471B1D1C3; Path=E:\temp\PsFileHash\MA\PSGL\_MA\_20160724.txt; PSComputerName=cewd6592.wfm.pvt; RunspaceId=77a1936a-9da6-4993-bb5d-b9d4416f7404; PSShowComputerName=True}

[2016-07-31 23:37:08] Created 2 total hash-value file(s).

# Manual Execution (Command Line)

**\*NOTE**: Execution from Tidal is different than from a Windows command console because Tidal provides us the name of the job within which we are executing.

From cmd.exe:

PowerShell.exe -Command "& 'e:\ScheduledJobs\PeopleSoft File Hash Generator\Test\PeopleSoftFileHashGenerator.ps1' Env=Test Region=MA 'Job=MA - DEV - PeopleSoftUpload File Hash'"

PowerShell.exe -Command "& 'e:\ScheduledJobs\PeopleSoft File Hash Generator\Test\PeopleSoftFileHashGenerator.ps1' Env=Test Region=MA 'Job=MA - DEV - PeopleSoftUpload File Hash' Date=20160718"