



## Pack N Hash 4.5

### Release Notes & Guide

#### Release Notes V4.5:

05/7/25

- Fixed bug resulting in value from prior PacknHash runs source folder box path being 'stuck', and not clearing properly without restarting program.
- GUI stability improvements to prevent potential issue of main window appearing to 'disappear' for periods and later return during extremely long operations.
- Updated python codebase and libraries to 3.12.

#### Release Notes V4.44:

07/15/24

- Fixed bug that would result in unnecessary rehashing of existing files in output folder during full PacknHash operation.
- Other minor bug fixes.
- Updated distribution method moving from single-file/portable application, to Windows Installable package.

#### Release Notes V4.43:

06/23/24

- Updated PacknHash executable libraries to address Splash Screen error about missing libraries if users' system was missing expected VCRUNTIME140.dll.
- Updated 7Zip Libraries to 24.07 Branch.
- Added new CSV Batch File to root of your archive output folder that will contain merged list of all individual case segments hashes from last completed batch.



## Release Notes V4.42:

03/14/24

- Hotfix adjusting commands for encryption passwords containing special characters that could create failures of 7z commands.

## Release Notes V4.41:

02/10/24

- Archive testing functions now multithreaded. Internal testing showed approximately 30% reduction in archive testing time when compared to same sized case batch.
- Lots of additional cleanup of code and message formatting in output console.
- Further improvements to messaging in Activity Dashboard.
- Improvements to automatic disabling/enabling of processing buttons in UI at start/ end of various processing tasks.

## Release Notes V4.4:

02/04/24

- Major UI Updates:
  - Refreshed UI
  - Significant recoding of subprocess modules eliminating secondary console windows and all 7zip status messages now piped into integrated 'Activity Dashboard'.
- Advanced source queuing and case information customization inspired by the updated queuing system 'BFIP', now integrated. Allows for user to verify what case folders are queued for archiving before starting processing, as well as adding cases from different drives to a single queue.
- Lots of updates to messaging and logging to take advantage of updated activity dashboard, and to minimize redundant notifications.
- Archive testing has gotten a significant update. While archives generated from PackNHash have always been automatically tested for integrity and CRC errors, this information and the results are now brought front and center to the user's attention with clear status indications for each tested archive. This new results window will be presented at the conclusion of any full PackNHash, or standalone Archive Test passes.
- API for automatically checking for updated versions of PackNHash now integrated. If connected to the internet, available updates will be queried on startup, or can be triggered from right click menu.

## Release Notes V4.3:

02/28/23

- 7z Libraries Updated to 22.01
- Updated Dependencies to Improve STDOUT Readability in console window during file integrity testing.



## Release Notes V4.2:

07/03/22

- Hashing Engine Rebuilt to now conduct multithreaded parallel hashing of archive segments. Direct tests between prior hashing engine and new multithreaded version yielded a *400% increase in speed*.
- 7z Binaries now directly integrated and no further 7Z installation necessary.
- UI Refinements including update console output that dynamically resizing when program window is maximized.
- Right Click Cut, Copy, Paste, and Help options.
- Integrated User Manual into program.
- Logs Moved to users appdata consistent with other Breakpoint Forensic Tools and button to open Logs folder added to UI.
- Other miscellaneous code cleanup, and improvements.

## Release Notes V4:

03/21/22

- Massive UI Update – Completely rebuilt from ground up
- All console entry is completely banished and handled through 100% GUI-based elements
- Exposed additional archive settings for optional configuration of:
  - Compression Level/Speed
  - Archive Encryption
- Common user configuration settings are now automatically saved and automatically set for you the next time PackNHash is run.
- New Validation Section allowing individual testing, file list generation, and hashing tasks to before performed on their own.
- Huge speed update to entire process when using recommended settings, that can yield 400% increase in speed over entire PackNHash process.
- New integrated console for cleaner feedback on processing progress with completion percentage updates provided throughout various stages.
- Refactored hashing process that now results in a single CSV file containing hash values for all segments of a case-folders archive.

Requirements:

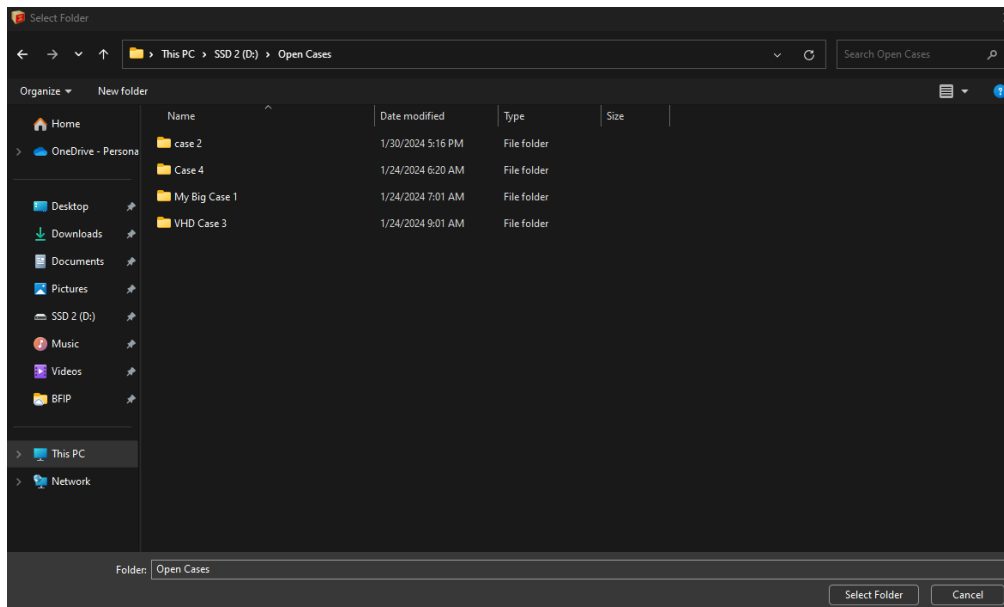
- Windows 10, 11



## Archiving Guide

1. Select the Source Folder containing **all your various case folders** requiring archive.

Note: Cases requiring archiving should be placed inside their own 'Source Folder'. This source folder is then what is selected in the folder chooser. \*Do not select an individual case folder in the Source Folder Selection.



Please select Source Folder containing case-folders for archive:

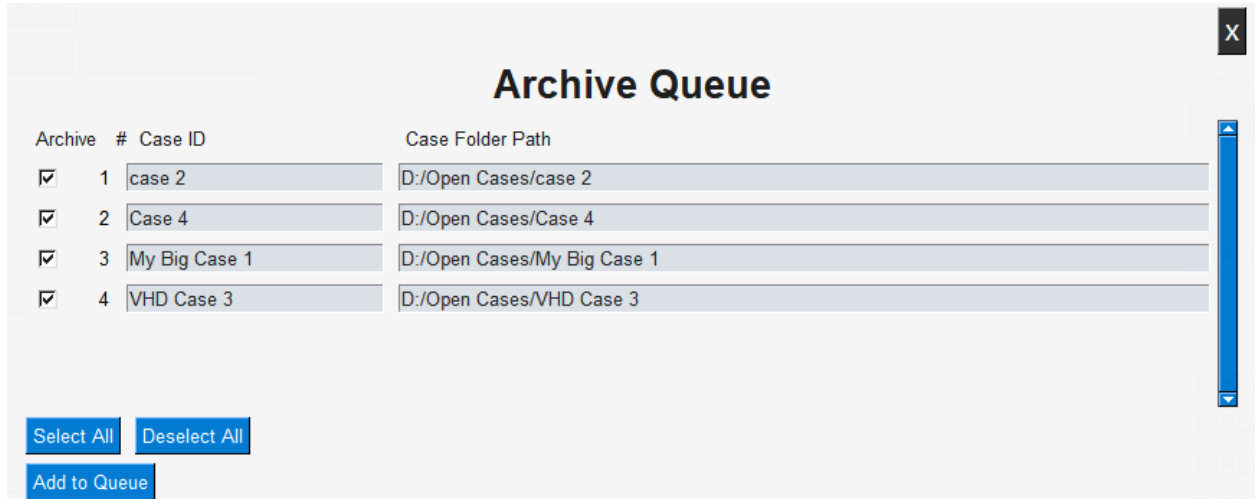
D:/Open Cases

Browse

Scan Source Folder

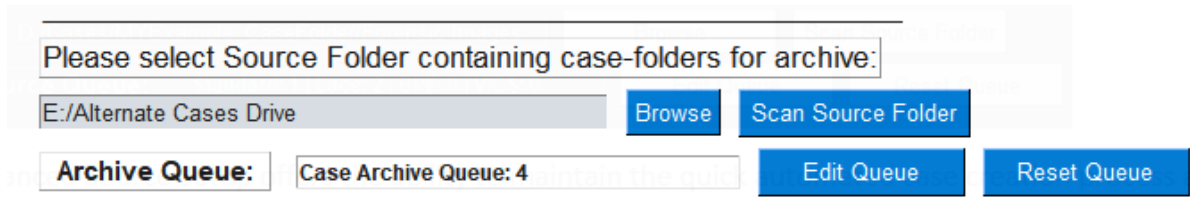


- Click 'Scan Source Folder'. PackNHash will then automatically identify all the individual case subfolders one level down, and will present located folders in a new Archive Queue window where the user can except the defaults, deselect any cases you do not wanted archived, or can change the final archive name.



- Adding additional files to queue (Optional):

The user may repeat this workflow adding a different 'Source Folder' to scan, pressing 'Scan Source Folder', and adding case folders to the queue multiple times. This allows for stacking multiple case folders scattered amongst several different folder locations to a single queue.



- Edit Queue

This opens a menu containing all previously selected cases and their current settings that have already been added to the queue using the 'Scan Source Folder' button. Existing folder paths, case ID's, etc., can all be further edited, removed or changed from here.



### Edit Archive Queue

Archive #	Case ID	Case Folder Path
<input checked="" type="checkbox"/>	1 case 2	D:/Open Cases/case 2
<input checked="" type="checkbox"/>	2 Case 4	D:/Open Cases/Case 4
<input checked="" type="checkbox"/>	3 My Big Case 1	D:/Open Cases/My Big Case 1
<input checked="" type="checkbox"/>	4 VHD Case 3	D:/Open Cases/VHD Case 3
<input checked="" type="checkbox"/>	5 case 5	E:/Alternate Cases Drive/case 5
<input checked="" type="checkbox"/>	6 case 6	E:/Alternate Cases Drive/case 6

ARCHIVING

Please select Source Folder containing case folders

E:/Alternate Cases Drive

Archive Queue: Case Archive Queue 1

Please select Archive Destination

E:/Alternate Cases Drive

5. Reset Queue:

This function completely clears the Case Archive Queue.

6. Archive Destination

Designate a destination folder for where to send the archives. To avoid unnecessary additional hashing, please make sure the destination directory is initially empty.

Please select Archive Destination:

E:/Archive Out

7. Configure optional segment size and compression settings.

a. Segment Size

Configurable in increments of Gigabytes

b. Compression Levels

There are 10 levels 0-9.

Level 0 = No Compression and is the fastest.

Level 9 = Most Compression and will take the longest to archive.

Level 1 is set as initial default and offers reasonable compression and good speed.

c. Encryption

Check box and enter password to encrypt with AES encryption



---

Optional Configuration Options

500 Archive Segment Size(GB)

---

Set Compression Level from 0-9  
(0=No Compression/Fastest to 9=Most Compression/Slowest)

1 Recommended=1

---

Optional Encryption Settings

\*If selected encryption will be applied to newly generated archives or will be used to test previously encrypted archives\*

Encrypted Archive      Encryption Password

---

8. Click 'Start' to Initiate Complete PacknHash Process

This will include:

- i. Package each case folder into individual case subfolders and archives.
- ii. Each generated archive will then be tested to ensure it can be opened and all files are intact.
- iii. A file listing text file is generated for each individual case archive.
- iv. All archive segments are MD5 hashed and logged into a CSV file in the case archives destination directory.





## Single Run/Validation Commands

PacknHash includes the option to selectively execute the various validation process from the full PacknHash process on an individual as-hoc basis.

This includes the ability to select either a parent folder containing several subfolders of various cases, or you can select an individual case archive folder containing your packed archives. PacknHash will recursively search for all archive segments within the selected folder and conduct the selected validation process.

# VALIDATION

---

Single Run/Validation Commands

Please select location of archives to validate:

Runs Selected Command against files located in path specified above

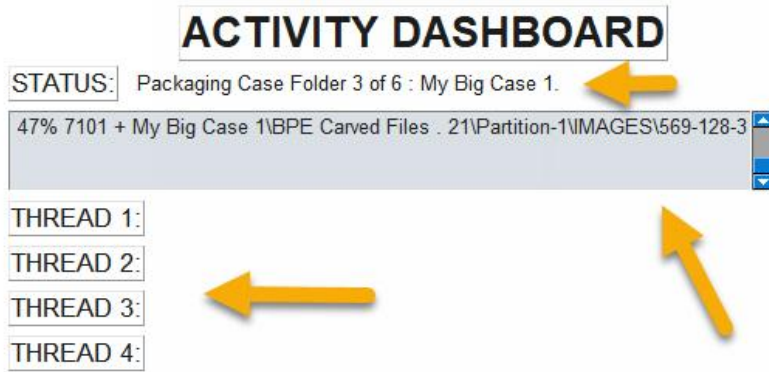
▲                      ▲                      ▲



## Output Window and Activity Dashboard

PacknHash utilizes an integrated output window and Activity Dashboard. Various confirmation messages, processing status, errors, etc. will be printed here for reference.

These areas provide a dynamic status messages that will occasionally update with hashing progress indicators and completion percentages.



**ACTIVITY DASHBOARD**

**STATUS:** Packaging Case Folder 3 of 6 : My Big Case 1.

47% 7101 + My Big Case 1\BPE Carved Files . 21\Partition-1\IMAGES\569-128-3

**THREAD 1:**

**THREAD 2:**

**THREAD 3:**

**THREAD 4:**

The screenshot shows a software interface titled "ACTIVITY DASHBOARD". It features a "STATUS:" label followed by the text "Packaging Case Folder 3 of 6 : My Big Case 1.". Below this is a progress bar or status line showing "47% 7101 + My Big Case 1\BPE Carved Files . 21\Partition-1\IMAGES\569-128-3". Underneath are four labels for "THREAD 1:" through "THREAD 4:". Three yellow arrows are overlaid on the image: one points to the "STATUS:" label, another points to the progress bar, and a third points to the "THREAD 2:" label.



## Result Window

At the conclusion of any full PackNHash or Testing run, the validation result for all generated/tested archives will be presented in a final window, along with any warnings. Validations are triggered when an archive or one of its segments is corrupted or a missing.

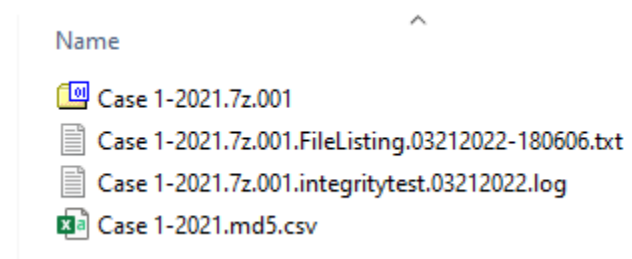
Archive	Result
E:/Archive Out/case 2	Validation Successful
E:/Archive Out/Case 4	Validation Successful
E:/Archive Out/My Big Case 1	Validation Successful
E:/Archive Out/VHD Case 3	Validation Successful
E:/Archive Out/case 5	Validation Successful
E:/Archive Out/case 6	Validation Successful

Archive	Result
case 2.7z.001	Validation Successful
Case 4.7z.001	Validation Successful
My Big Case 1.7z.001	Validation Successful
VHD Case 3.7z.001	Validation Failed

## Generated Log/Validation Files

Inside each case folder will be 3 different log files:



1. A file Listing containing a full list of all files that were added to archive.
2. An integrity test log indicating the result of the final integrity test of the completed archive.
3. A CSV holding MD5 values for a segment of the archive.

Hash Timestamp	Hash Type	FileName	Value
3/21/2022 18:06	MD5	case 1-2021.7z.001	d1aac3be8533b706835870aa1adc07b3

