Hands-On Lab: Backup and Recovery and File Integrity Monitoring

To accompany Whitman and Mattord, Principles of Information Security, 7th Ed., 2022, ISBN 978-0-357-50643-1; Backup and Recovery and File Integrity Monitoring

Table of Contents

[Introduction 2](#_Toc76727418)

[Objective 2](#_Toc76727419)

[Estimated Completion Time 2](#_Toc76727420)

[Materials Required 2](#_Toc76727421)

[Minimum System Configuration 2](#_Toc76727422)

[Overview of PC Data Backup and Recovery 3](#_Toc76727423)

[Microsoft Windows 10 3](#_Toc76727424)

[Apple macOS 4](#_Toc76727425)

[Linux 4](#_Toc76727426)

[Simple File Integrity Monitoring with PowerShell Get-FileHash 5](#_Toc76727427)

[Accessing PowerShell 5](#_Toc76727428)

[Using PowerShell Get-FileHash Utility 6](#_Toc76727429)

[Using Word to Compare Two PowerShell Get-FileHash Output Files 7](#_Toc76727430)

[Simple File Integrity Monitoring with HashCalc 9](#_Toc76727431)

[Download/Install HashCalc 9](#_Toc76727432)

[Using HashCalc to Calculate Hashes 10](#_Toc76727433)

[Using Maresware Hash64 and Hashcmp to Monitor File Integrity 13](#_Toc76727434)

[Downloading Hash64 and Hashcmp 13](#_Toc76727435)

[Using Hash64 to Calculate File Hashes 14](#_Toc76727436)

[Using Hashcmp to Compare File Hashes 15](#_Toc76727437)

[Self-Reflection and Response 16](#_Toc76727438)

[Instructor’s Response 16](#_Toc76727439)

# Introduction

The most significant action you can take to improve the resilience and survivability of your computer system is to create a robust and tested backup process.

In this lab, you will explore the process of making a backup copy of a computer system. The lab discusses how to set up regular system and data backups for a Windows, Apple, and Linux OS. If you are using a different OS, the process should be relatively similar. You can use a web browser or your OS’s help files to locate the steps for your specific OS.

Next, we will examine the process of determining if a file has changed, whether as a result of a backup and restore or as part of an attack.

## Objective

Upon completion of this activity, you will be able to:

* Describe backup and recovery processes and will be aware of basic backup activities using Windows 10 or another desktop operating system (OS).
* Perform file integrity monitoring using file hash values.

## Estimated Completion Time

If you are prepared, you should be able to complete all tasks in this lab in 15 to 20 minutes. Backup and recovery tasks may take longer if you have a large drive with an extensive amount of data to back up.

## Materials Required

Completion of this lab requires the following software to be installed and configured on your personal computer:

* Microsoft Windows 10, or another operating system version specified by the lab instructor.
* Windows PowerShell enabled on the system.
* HashCalc from <https://www.slavasoft.com/download.htm>, and hash.exe and hashcmp.exe from Maresware downloaded as part of the second set of labs.

To perform and store actual system and data backups, you will need an internal hard drive or external USB drive not currently used on your system.

The lab instructions make use of Microsoft Word.

## Minimum System Configuration

To complete the labs included, it is recommended that you operate them from a computer system (desktop or laptop) that is running Windows 10 and has:

* Intel i5 or better CPU
* 8 GB RAM (minimum) - 16 GB RAM (recommended)
* 1 TB Hard Drive with at least 250 GB free (minimum) - 350 GB free (recommended)
* Microsoft Windows 10 or latest version

[[return to top]](file:///C:\Users\danse\Documents\Cengage\Principles%20of%20IR%20and%20DR%203e\Supplements\Hands-on%20Projects\Final%20versions%20of%20hands-on%20documents\Lab_Mod_01___Overview_of_Information_Security_&amp;_Risk_Management_DE.docx#_top)

# Overview of PC Data Backup and Recovery

Several good tutorials are available online that demonstrate the use of backup and recovery procedures. Rather than replicate their work, this lab simply points to those tutorials. As a future IT or security professional, it is important that you “walk the walk” as well as “talk the talk.” In other words, you must be responsible for providing security for yourself as well as for others. Given the widespread availability of low-cost cloud storage and high-capacity external USB drives, there is no excuse for losing key files.

Quite a few cloud backup services are available:

* Carbonite ([www.carbonite.com](http://www.carbonite.com))
* Dropbox ([www.dropbox.com](http://www.dropbox.com))
* Microsoft (through OneDrive at [www.onedrive.live.com](http://www.onedrive.live.com))
* Apple (through iCloud at [www.apple/icloud](http://www.apple/icloud))
* Google (through Google Drive at [www.google.com/drive/](http://www.google.com/drive/))

Several of these vendors have space they provide at no cost once you establish an account with them. All of the preceding vendors provide additional services and larger storage capacities with a paid subscription.

For example, Dropbox offers 2 GB of storage for free. Dropbox works by allowing you to create file folders in Windows Explorer (for Windows systems). The folders you create and the files in them are stored on your local system and then synchronized with Dropbox cloud storage. You can install Dropbox on multiple computers, allowing synchronization among the computers and multiple users.

## Microsoft Windows 10

For local data backup and recovery, including system backups, Microsoft uses a feature called “Back up using File History.” You can locate it by typing “Backup” in the Windows search bar. This feature allows traditional backups and recovery using local or networked drives. Review each of the documents listed below and perform a test backup using the instructions provided.

* The Windows support page that describes backup and recovery using the Windows 10 File History feature is available at <https://support.microsoft.com/en-us/help/4027408/windows-10-backup-and-restore>.
* A related article on Microsoft’s support site discusses recovery options under Windows 10: <https://support.microsoft.com/en-us/help/12415/windows-10-recovery-options>.
* Windows Central ([www.windowscentral.com](http://www.windowscentral.com)), a popular user help site, provides an article titled “How to use Windows 10 File History to back up data” at [www.windowscentral.com/how-use-file-history-back-your-files](http://www.windowscentral.com/how-use-file-history-back-your-files).

Many more excellent online tutorials discuss Windows backup and recovery. If the preceding links do not provide satisfactory instructions, use a web browser to search for others.

## Apple macOS

Like Windows, Apple provides backup and recovery instructions for its users. Between iCloud and the “Time Machine” built-in backup utility, Apple users have a number of options.

* Apple’s support site provides instructions for backing a Mac up to a Time Machine backup: <https://support.apple.com/en-us/HT201250>.
* Apple’s support site also provides instructions for restoring a Mac from a Time Machine backup: <https://support.apple.com/en-us/HT203981>.

## Linux

Linux users may have to conduct some research to learn how their version of Linux performs backups. Some parts of the process will depend on the type of file system that’s installed (for example, Ext3 or Ext4). Web resources like the following link provide instructions for performing command-line backups using external USB drives:

* How-To Geek ([www.howtogeek.com](http://www.howtogeek.com)), a popular user support site, provides the following article that demonstrates how to perform Linux backups using the rsync utility: [www.howtogeek.com/427480/how-to-back-up-your-linux-system/](http://www.howtogeek.com/427480/how-to-back-up-your-linux-system/).

[[return to top]](file:///C:\Users\danse\Documents\Cengage\Principles%20of%20IR%20and%20DR%203e\Supplements\Hands-on%20Projects\Final%20versions%20of%20hands-on%20documents\Lab_Mod_01___Overview_of_Information_Security_&amp;_Risk_Management_DE.docx#_top)

File Integrity Monitoring

What is file integrity monitoring? Simply put, it’s the evaluation of a file to see if it has changed. This is the foundation of backup and recovery file validation and commonly used in host intrusion detection and prevention systems. For backup and recovery, we calculate the hash of a file before we back it up, and then compare that hash to the hash of the file once it’s restored. If the two are the same, the file is valid and unchanged. If the hashes have changed, then something went wrong. Most network systems do something similar when they transmit a file – using something called a checksum. This is the same basic practice.

In host IDPS, the system can calculate the hash of all key files that should not change. It then periodically calculates new hashes and compares the two. If the hash has changed, then the file has changed and the HIDPS notifies an administrator to look to see if an update was performed, or whether an attacker has modified the file.

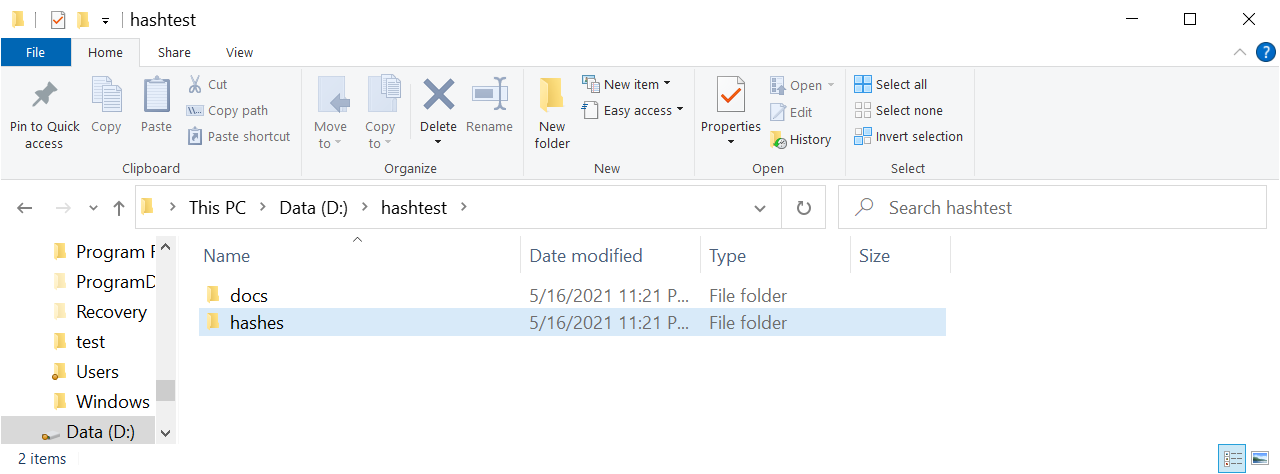
Any changes to the file will result in changed hash values. In this lab, we’ll look at a simple way you can use common Windows tools to perform a simple file hash and comparison.

## Simple File Integrity Monitoring with PowerShell Get-FileHash

In this lab, we will calculate the hash values of a number of files, and then use Microsoft Word to compare a before and after. For the first part of this exercise, we will use the built-in Windows utility PowerShell.

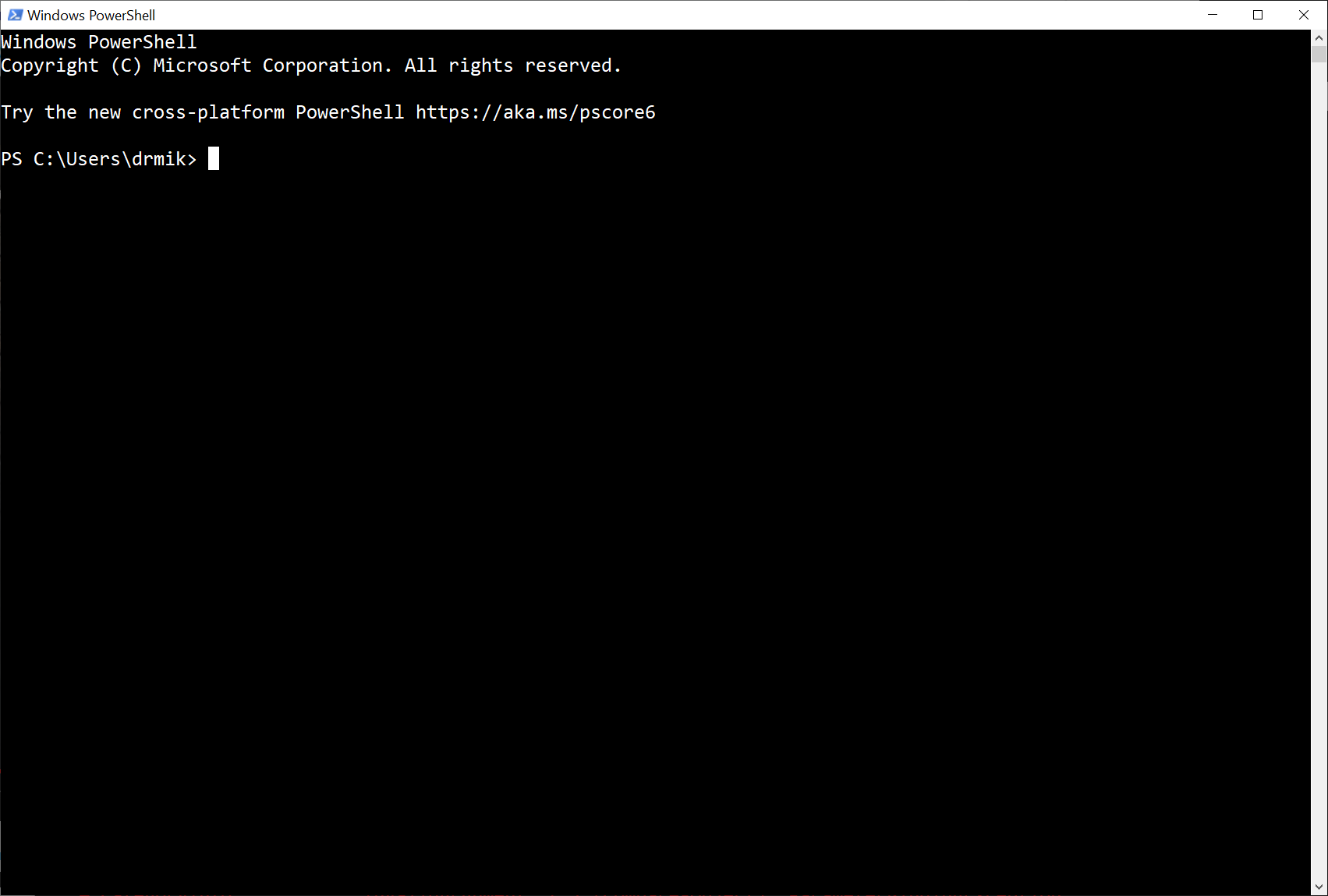
### Accessing PowerShell

1. Before we start this lab, on a USB drive or student directory you can access during the lab, create a folder called hashtest. n that folder create two more subfolders called docs and hashes, as shown in Figure L05-1.



**Figure L05-1** hashtest folders

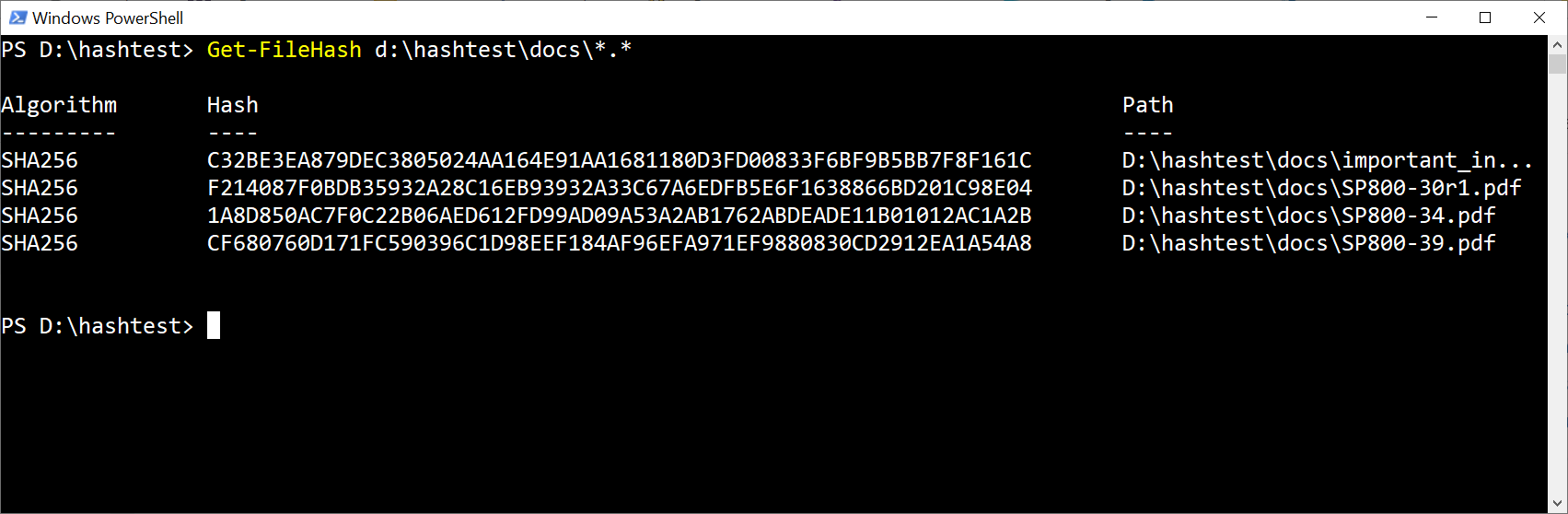
1. Next, find and save a few files to the docs folder to use in the exercise. It doesn’t matter what they are as long as you can edit at least one of them. In my example, I’ve downloaded several NIST Special publication PDFs and a .txt file I can edit called important\_information.txt.
2. Next start PowerShell by right clicking the Windows **Start** button, then selecting **Windows PowerShell**. You should see the Windows PowerShell window open as shown in Figure L05-2.



**Figure L05-2** Windows PowerShell

### Using PowerShell Get-FileHash Utility

1. The PowerShell Utility Get-FileHash (described here: <https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/get-filehash?view=powershell-7.1>) provides a way to calculate the Hash value using a number of different algorithms. Our example will use the default SHA256. Other options and parameters are described in the above Microsoft document.
2. Navigate to the hashtest folder you created. Then type the following command:  
   **Get-FileHash *d:\*hashtest\docs\\*.\*** (replacing ***d:*** with the path to your hashtest folder) and hit **Enter**. You should see a list of the files contained in your docs directory and their hash values. This information is only placed on the PowerShell windows right now.



**Figure L05- 3** Get-FileHash Results

1. To copy the results to a file, type the same command with the following modifications:  
   **Get-FileHash *d:\*hashtest\docs\\*.\*** **> *d:*\hashtest\hashes\hash1.txt** (replacing ***d:*** with the path to your hashtest folder) and hit **Enter**.
2. You won’t see much happen as the system redirects the output to the text file you specify. Using Windows Explorer, navigate to your hashtest/hashes directory and look. You should see a text file: *hash1.txt*. Double click to open it. It should be identical to the results you saw on your screen.

### Using Word to Compare Two PowerShell Get-FileHash Output Files

1. Our next step is to change one of the files and then use Get-FileHash to calculate another hash text file, and then use Word to compare the two to see if we can detect the changes. Open one of the files you saved and make a few changes. It only takes a single change to end up with a different hash. Save your changes and then run the previous PowerShell command sending the output to a different text file as follows:  
   **Get-FileHash *d:\*hashtest\docs\\*.\*** **> *d:*\hashtest\hashes\hash2.txt** (replacing ***d:*** with the path to your hashtest folder) and hit **Enter**. (As a shortcut in PowerShell, you can use the up arrow to scroll through your previous commands and then change them).
2. Look in your *hashtest/hashes* folder. You should see two text files now: *hash1.txt* and *hash2.txt*.
3. Open Microsoft Word, and then open both text files in Word. If Word prompts you with a File Conversion window, just select **Unicode**, and click **OK**.
4. Click the **Review** menu tab at the top of one of the text files in Word, then, click the **Compare** drop-down button. First set up the output windows by selecting **Show Source Documents** and then selecting **Show Both**.
5. Next click the **Compare** button again and select the **Compare…** command in the menu to open the *Compare Documents* dialog box.
6. Under *Original document*, select **hash1.txt**. It should be available since you have the file opened. If not, navigate to the *hashtest/hashes* folder and select it.
7. Next, under *Revised document*, select **hash2.txt**, and click **OK**.
8. Word will open a new document with any differences flagged as shown in Figure L05-4. In your document, you should see a red bar beside the file you changed.



**Figure L05-4** Word Comparison of hash text files

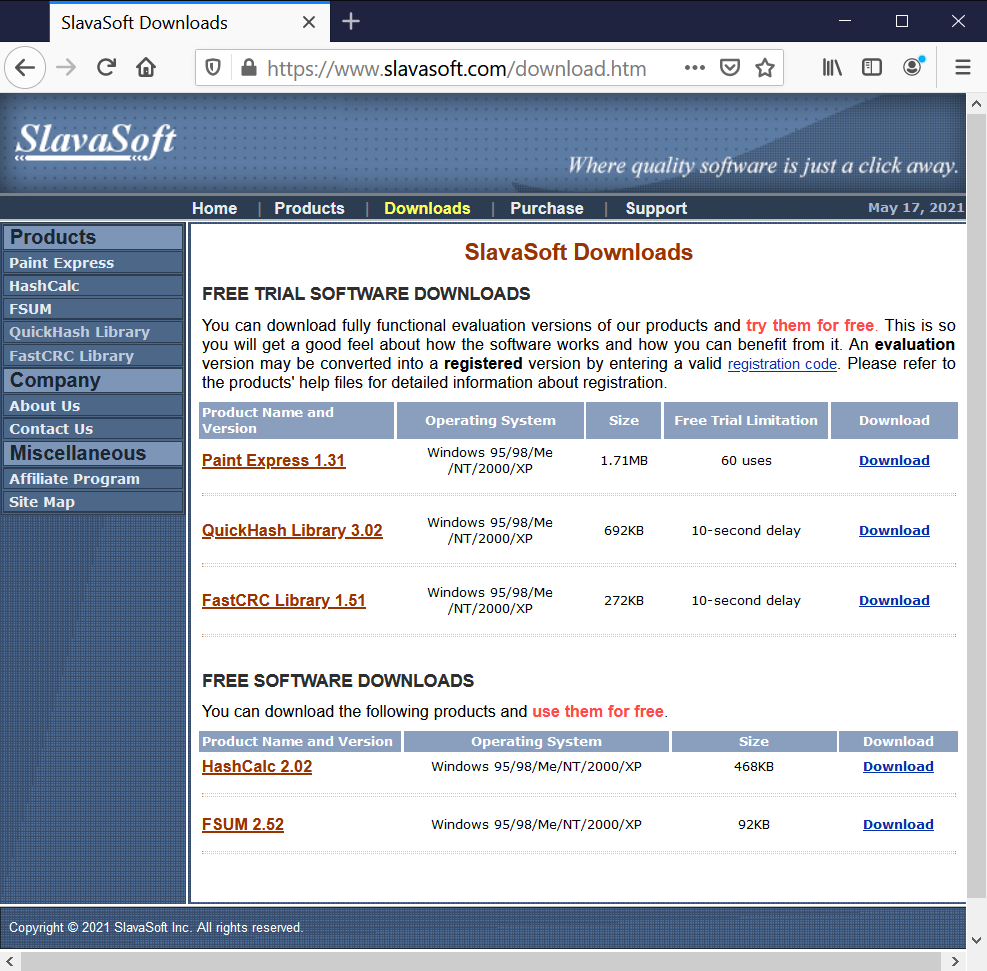
1. The original document and revised document appear at the right side of the screen. The compared document appears in the center of the screen. Any revisions appear in the “Revisions” pane at the left side of the screen. Here you see one entry for “author deleted” and one for “author inserted.” This means the two hash values in the two documents are different. You can do this with hundreds of file hash values and the process will not only detect changed hash values but also new or missing files.

## Simple File Integrity Monitoring with HashCalc

In this lab, students will calculate multiple hash values using a freely available program from SlavaSoft.

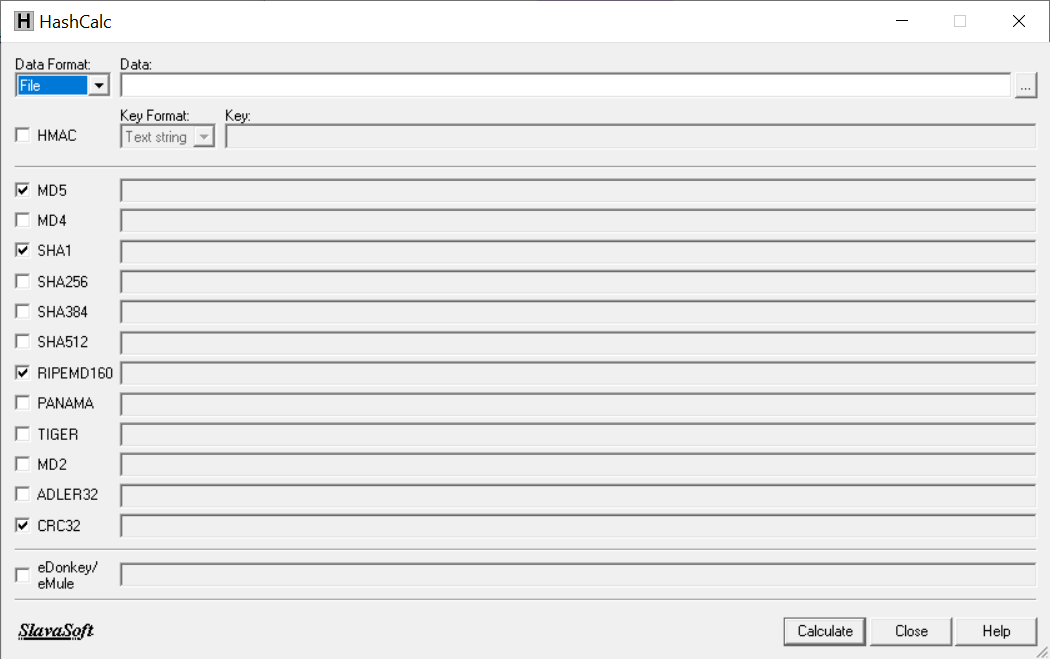
### Download/Install HashCalc

1. Using a web browser, go to <https://www.slavasoft.com/> and select the Downloads option in the top menu at the bottom of this page, shown in Figure L05-5.



**Figure L05-5** lavaSoft Downloads Page

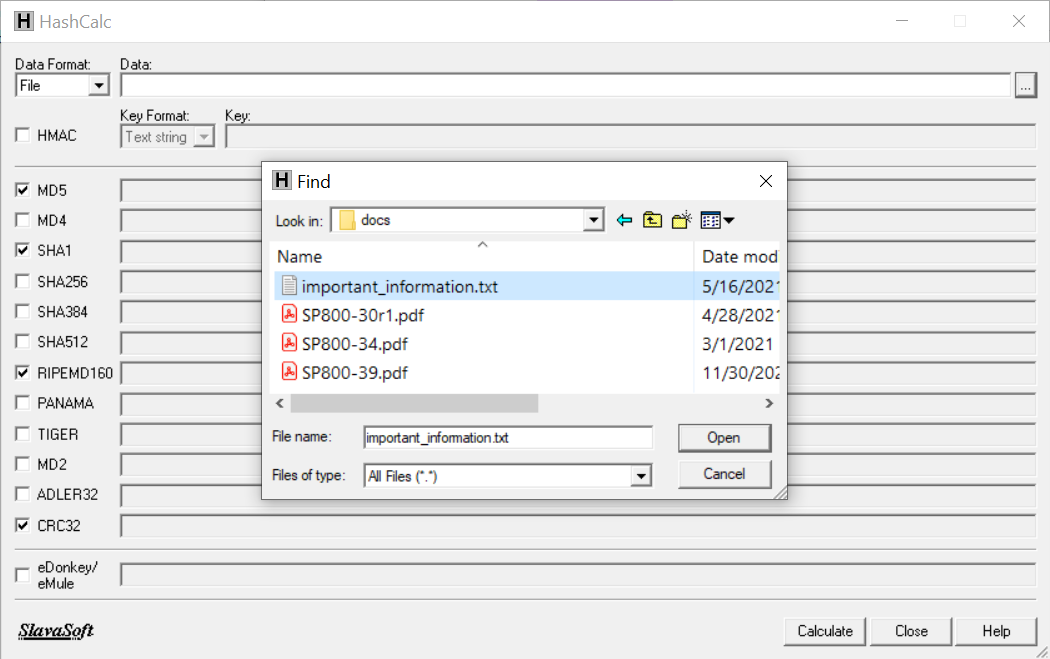
1. At the bottom of the page, click the download links for HashCalc 2.02.
2. Save the downloaded zip file to your hashtest directory where you can execute it. Once you extract the file from its zip containers, you will need to install the HashCalc.
3. Double click the HashCalc **setup.exe**. When prompted accept the agreement. Install it to your hashtest directory, using the options provided. When the install finishes, it should start HashCalc automatically, as shown below.



**Figure L05-6** SlavaSoft HashCalc

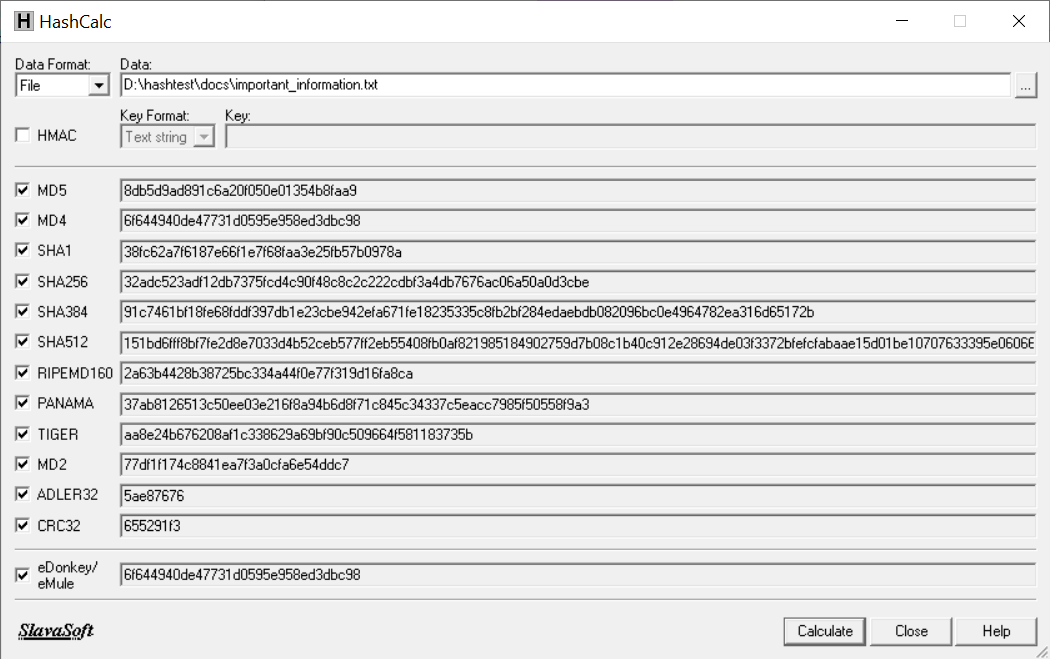
### Using HashCalc to Calculate Hashes

1. HashCalc is a tool that can calculate the value of a file or a data string using multiple algorithms simultaneously. Begin by selecting the **…** button to the right of the data field and navigating up to your docs folder. Select a document you can edit and click **Open**, as shown in Figure L05-7 below.



**Figure L05-7** SlavaSoft HashCalc selecting file

1. Next check all the boxes to on the left of HashCalc, and then click the Calculate button at the bottom as shown here in Figure L05-8. HashCalc creates the hash values using all these different algorithms.



**Figure L05-8** SlavaSoft HashCalc results

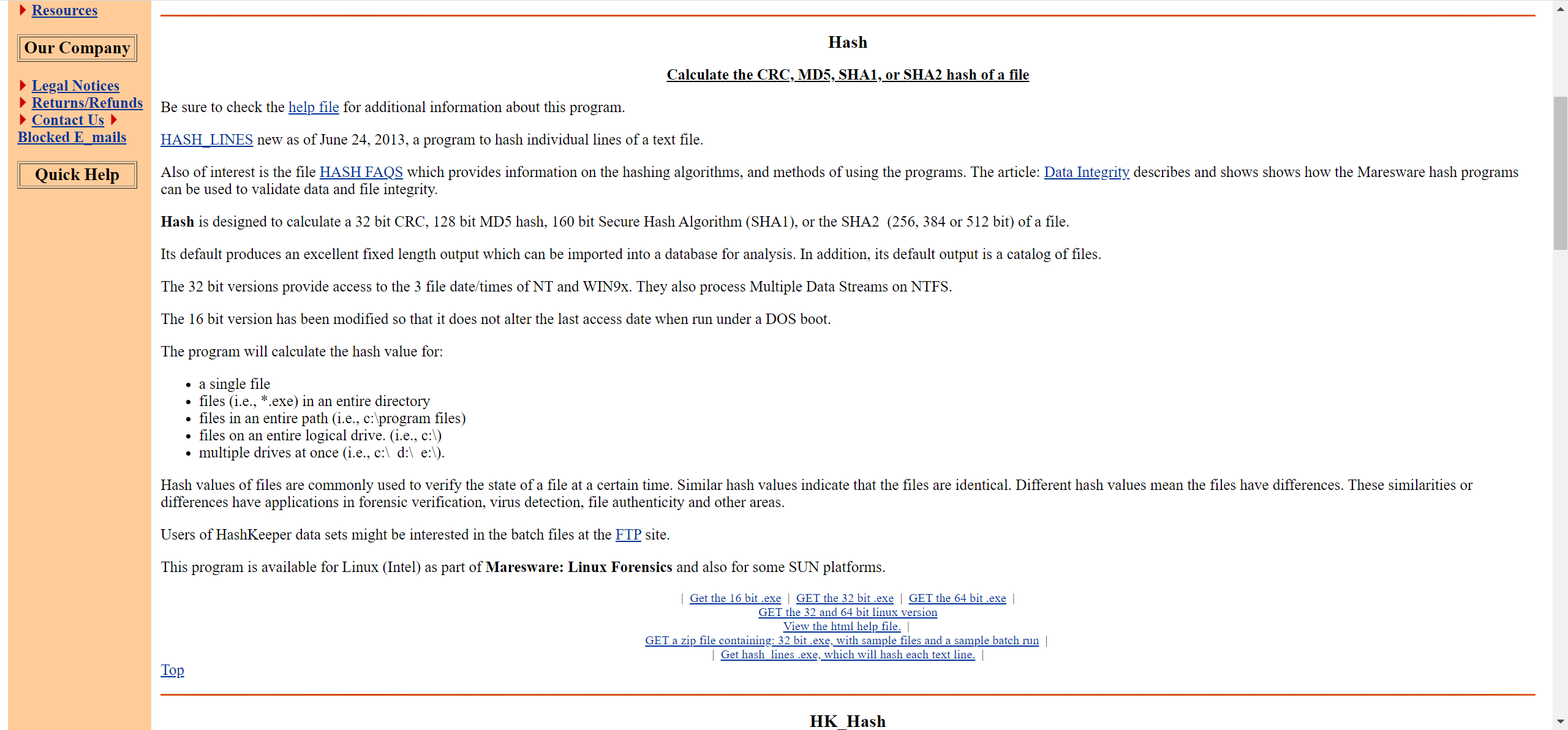
1. Close the HashCalc utility when finished.

## Using Maresware Hash64 and Hashcmp to Monitor File Integrity

While HashCalc does a great job of calculating single hash values, we need more functionality to do a file integrity check, especially with multiple files. We can use two utilities from Dan Mare’s “MaresWare.” Dan Mares is a long renowned expert in Computer Forensics and has recently made some of his utilities free to download.

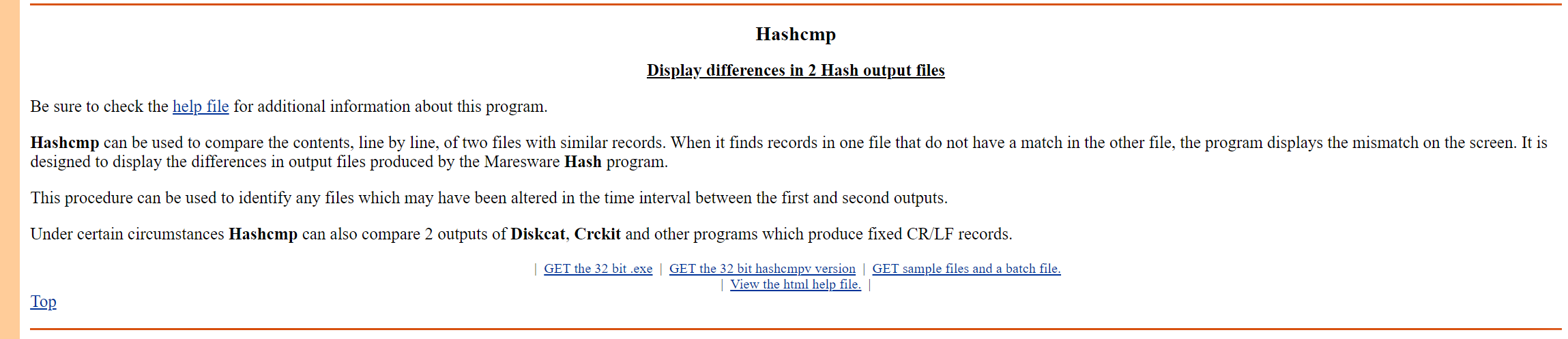
### Downloading Hash64 and Hashcmp

1. Using a web browser, go to <http://www.dmares.com/maresware/gk.htm>. Scroll down to *Hash* and click the **Get the 64-bit .exe** link at the bottom, as shown in Figure L05-9. *(Do NOT select Get the 32 bit .exe)*. Save the file to your *hashtest* directory. You can view help for Hash here: <http://www.dmares.com/maresware/html/hash.htm>.



**Figure L05-9** MaresWare Hash utility download

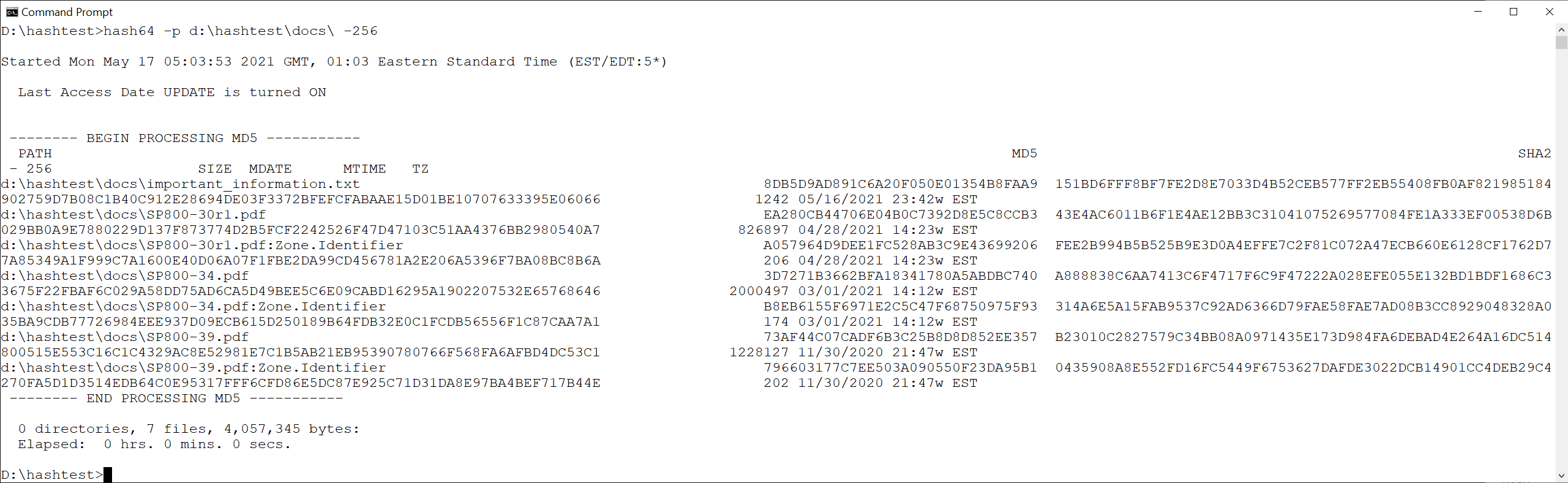
1. Scroll down to *Hashcmp* and click the **Get the 32 bit .exe** link at the bottom, as shown in Figure L05-10. Save the file to your *hashtest* directory. You can view help for Hashcmp here: <http://www.dmares.com/maresware/html/hashcmp.htm>.



**Figure L05-10** MaresWare Hashcmp utility download

### Using Hash64 to Calculate File Hashes

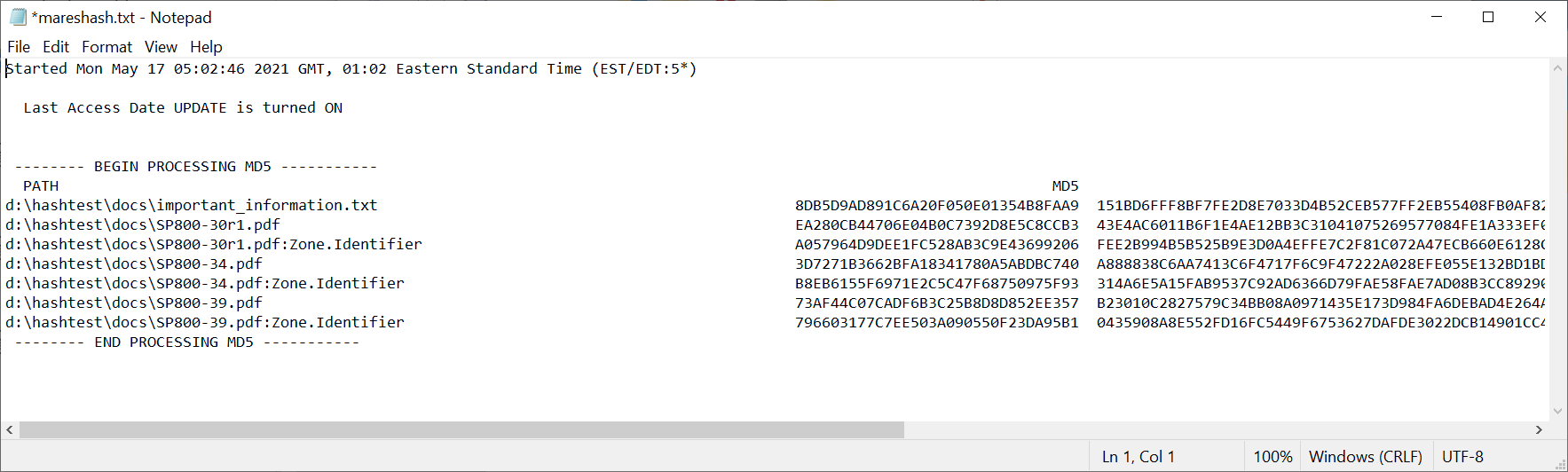
1. Neither of these are installed utilities, as they both run directly from the .exe. We’ll begin by using the hash command line tool to calculate the hash of one of our files. Hash.exe can do the same things as the PowerShell version, calculating the hash of a single file or entire folder of files. You could calculate the hash values of your entire hard drive if you chose to do so. Open a command window by typing cmd in the Windows task bar search field. Navigate to your hashtest directory and type the following command: **hash64 -p d:\hashtest\docs\ -256**. This will give us both the MD5 and SHA-256 hashes for all files in our docs directory, as shown below in Figure L05-11.



**Figure L05-11** Mareware hash64 results

1. We can redirect this output to a file for future comparison by adding the Windows redirect: **hash64 -p d:\hashtest\docs\ -256 > d:\hashtest\hashes\mareshash.txt**

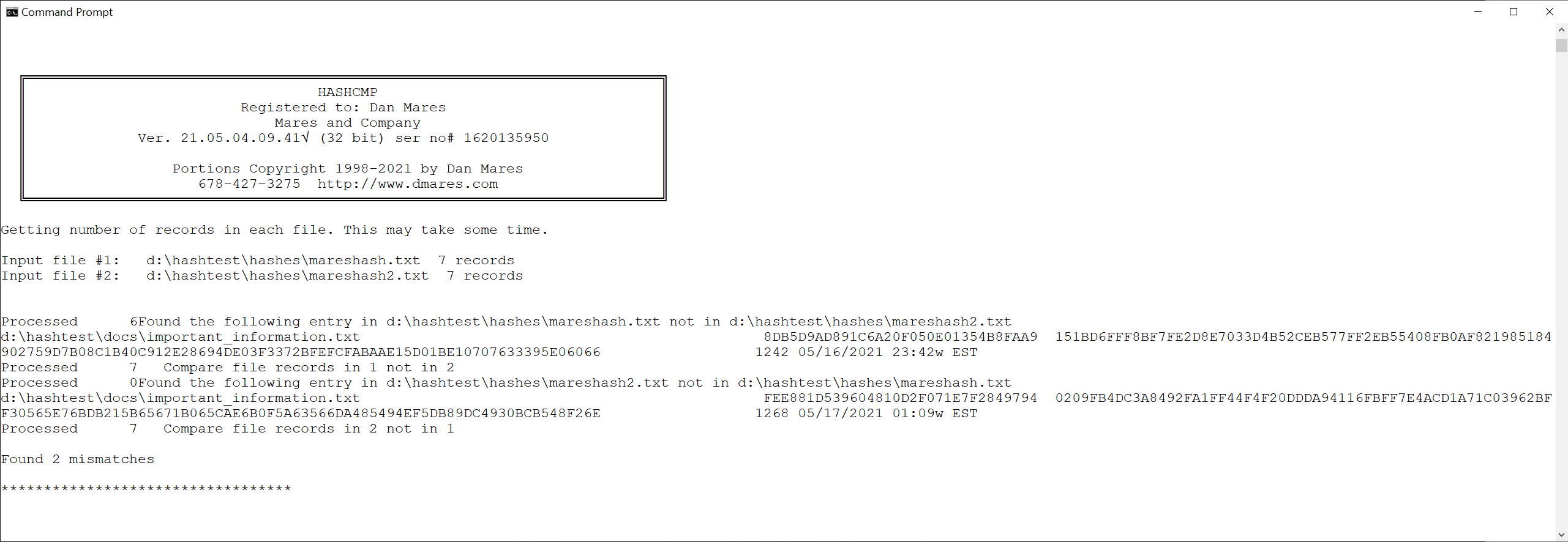
(substituting your folder locations for d: here). You should be able to verify the output file in your hashes folder as shown below in Figure L05-12.



**Figure L05-12** Mareware Hash64 text output

### Using Hashcmp to Compare File Hashes

1. Now it’s time for the second part, the comparison. First open one of the files in your docs directory and change it. Then run the previous command again, directing your output to a second file like this: **hash64 -p d:\hashtest\docs\ -256 > d:\hashtest\hashes\mareshash2.txt** (substituting your folder locations for d: here).
2. Now type the following command: **hashcmp d:\hashtest\hashes\mareshash.txt d:\hashtest\hashes\mareshash2.txt** (substitute your directory information for d. as before). You should see a message that there was a difference found between the files in the two hash output files as shown in Figure L05-13.



**Figure L05-13**

1. This is a clear indication that the two files are different. Why they are different is a different task entirely. You could automate the entire process in a batch file accomplishing the following:

* First you would run a hash to output on a specified folder of files.
* Then you would develop a .bat (batch) program to run a hash on the same folder directing the output to a new file, and then run hashcmp to compare the new hash output file to the old.

This would essentially be a host intrusion detection process, where you detect changed files in the target directory.

# Self-Reflection and Response

Have you chosen to make a backup copy of your computer system? In the space below, explain why or why not. What steps did you take (or will you take in the future) to research and implement your method?

|  |
| --- |
|  |

Can you think of another reason, not mentioned in the lab, for using the File Integrity monitor featers found in PowerShell? Describe how you might use it.

|  |
| --- |
|  |

Were you able to install and use the hashing tools from MaresWare? What was your experience using these tools?

|  |
| --- |
|  |

## Instructor’s Response

|  |
| --- |
|  |