

Data Validation Pack for QuickHash-GUI

Author : Ted Smith, December 2023

Table of Contents

| | |
|--|---|
| Introduction..... | 1 |
| Disclaimer..... | 1 |
| Intended Audiences..... | 2 |
| Ground Truth Data..... | 2 |
| Validation Status..... | 3 |
| Forensic Images..... | 6 |
| Hash Collisions and Deprecation of ‘Cryptographic’ status..... | 7 |
| Suggested Validation Exercise..... | 7 |

Introduction

This document aims to facilitate the validation of deployed instances of QuickHash-GUI in work areas regulated by the Forensic Science Regulator (UK) and international equivalents.

In the UK, any digital evidence provided in the context of criminal prosecution activity falls under the remit of The Forensic Science Regulator and the associated Codes of Practice.

This document aims to detail some basic validation checks using generated Ground Truth Data to ensure that users can see, and demonstrate, that the output from QuickHash-GUI is correct in their environment.

Note that QuickHash-GUI is simply an open-source data hashing tool. It is not, nor has it ever been declared to be, a ‘forensic tool’ like X-Ways Forensics, The Forensic Toolkit, Axiom, NUIX, etc. It is, however, routinely deployed and used within the workflows of digital forensic environments, and for that reason, this data validation pack has been prepared.

Disclaimer

This document has been produced to be helpful to those engaged in criminal prosecution work, but it is merely a starting point and a *suggested* accompaniment of validation. It is not a definitive

legally binding document and it has been written independently of any law enforcement agency or forensic science regulator of any country. All users of QuickHash-GUI are encouraged to use their own methods of validation in accordance with their established validation processes, technical expertise, Quality Assurance teams, and accredited assessment bodies. Finally, the project is open-source, so anyone who wishes to dig deeper into the inner mechanics of the tool is welcome to do so.

At the time of writing, the project website is www.QuickHash-GUI.org. Software downloaded from that location, and only that location, relates to this validation pack. It does not relate to other sources where the tool is hosted without developer consent or endorsement.

That aside, for any other questions about this work, please contact tedsmith@quickhash-gui.org.

Intended Audiences

The intended audiences for this validation pack are those who use QuickHash-GUI as part of digital workflows used in the furtherance of a criminal investigation. It is not intended for academics, private users, civil investigations, hobbyists or anyone who uses QuickHash-GUI outside of a judicial system. Though all users can, and may, wish to benefit from it if they choose to.

Ground Truth Data

This validation pack includes a number of files containing simple data for which the intended hash values of the primarily used algorithms can reasonably and easily be determined by using any data hashing tools.

The character 'a' should always return the following hash value for the corresponding hash algorithms, no matter what data hashing tool is used to compute it :

MD5 :

0cc175b9c0f1b6a831c399e269772661

SHA-1 :

```
86f7e437faa5a7fce15d1ddcb9eaeaea377667b8
```

SHA-256 :

```
ca978112ca1bbdcafac231b39a23dc4da786eff8147c4e72b9807785afee48bb
```

SHA-512 :

```
1f40fc92da241694750979ee6cf582f2d5d7d28e18335de05abc54d0560e0f5302
```

```
860c652bf08d560252aa5e74210546f369fbbbce8c12cfc7957b2652fe9a75
```

The remaining hash algorithms are very fast and are used in a myriad of circumstances globally. But less commonly in digital forensic scenarios, so they are not documented further here. They can instead be validated by the user in the same way as documented here but under their own direction.

A 1 byte source file called ‘a.txt’ (without a carriage return) is provided to facilitate this check for the readers convenience. Note that text files in Linux inherently contain a line feed making a 1 character file 2 bytes (not the case with Windows). The reader could modify their own text file in Linux as to remove line endings (or carriage returns) using awk. For example :

```
awk '{printf("%s", $0)}' input_file.txt > output_file.txt
```

Validation Status

All new releases of QuickHash-GUI are checked when new hash algorithms are added. They are not checked subsequently unless a development feature or fix is applied that has a direct impact on the computation of an algorithm result. That said, all users are encouraged to use this document as a means to cross reference their own deployment of the tool into their given IT infrastructure. For example, if the tool is deployed onto a new Windows operating system, that differed from a previous version, they may wish to check that QuickHash-GUI still computes ‘ca978112ca1bbdcafac231b39a23dc4da786eff8147c4e72b9807785afee48bb’ for SHA-256 for the single lower case character ‘a’.

For completeness, the screenshots below illustrate QuickHash-GUI v3.3.4 on Linux Mint 21 producing the hash of the letter ‘a’, cross referenced using ‘<https://emn178.github.io/online-tools/>’. Forensic tools could be used as well, of course (see below in subsequent pages).

The validation step can be repeated using the File tab as well, and pointing QuickHash-GUI to 'a.txt' should produce the same hashes as documented above on both Linux, Windows and Apple OSX.

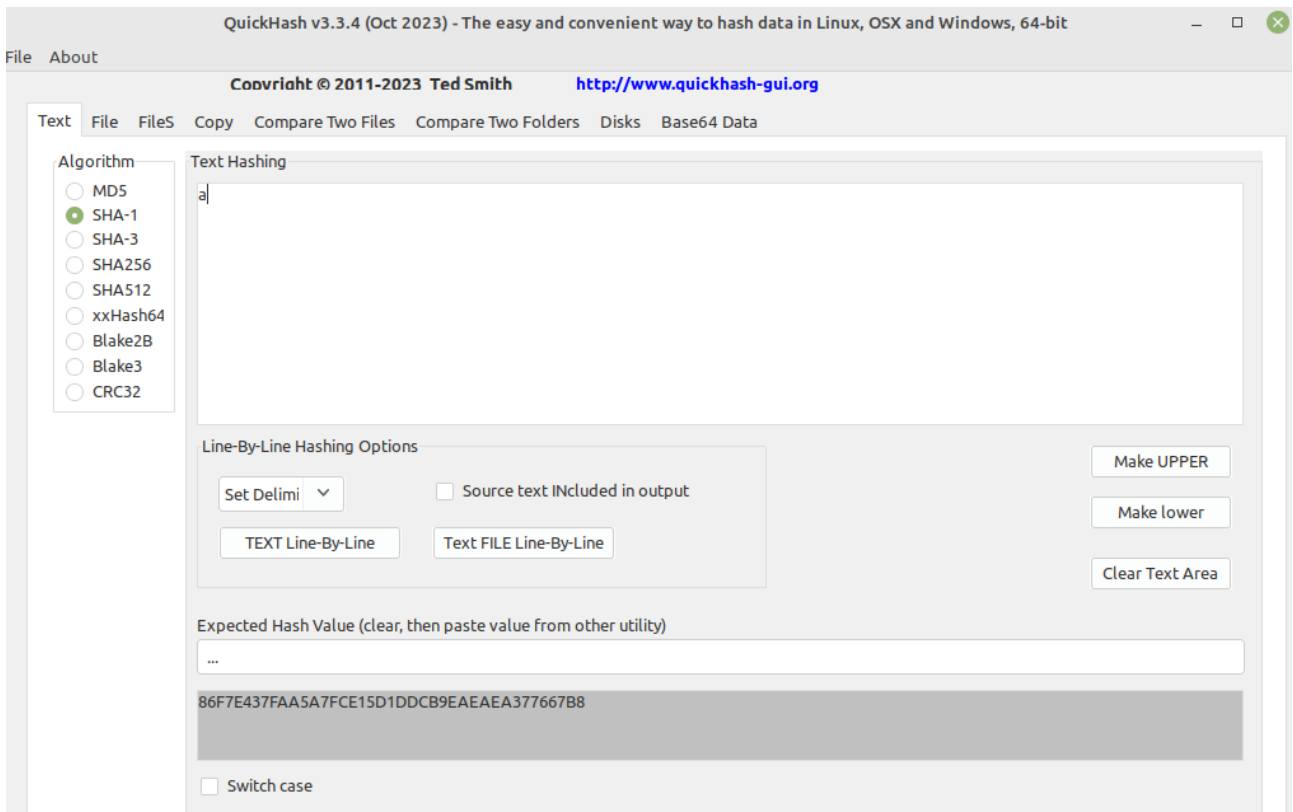


Figure 1: The SHA-1 hash value of the 'a' character

Online Tools Hash Encoding Misc

SHA1

This SHA1 online tool helps you calculate hash from string or binary. You can input UTF-8, UTF-16, Hex to SHA1. It also supports HMAC.

Input Type UTF-8

a

☐ Remember Input☐ Enable HMAC

Hash ☒ Auto Update

86f7e437faa5a7fce15d1ddcb9eaeaea377667b8

Figure 2: The SHA-1 hash value of the 'a' character as computed online

The same file in a Windows environment saved to a 10Mb virtually attached and mounted 10Mb disk generates the same value as shown below with X-Ways Forensics :

| | | |
|------------------|--|-------------------------|
| Secure (3) | | existing, already viewe |
| UpCase (1) | | existing |
| \$Volume | | existing, already viewe |
| a.txt | 86F7E437FAA5A7FCE15D1DDCB9EAEAEA377667B8 | existing |
| Free space (net) | | virtual (for examinatio |
| Idle space | | virtual (for examinatio |

Figure 3: The SHA-1 hash value of the file 'a.txt', storing the character 'a', computed by X-Ways Forensics

And the same file on the main operating system drive can be checked with QuickHash-GUI and shows the same computed hash :

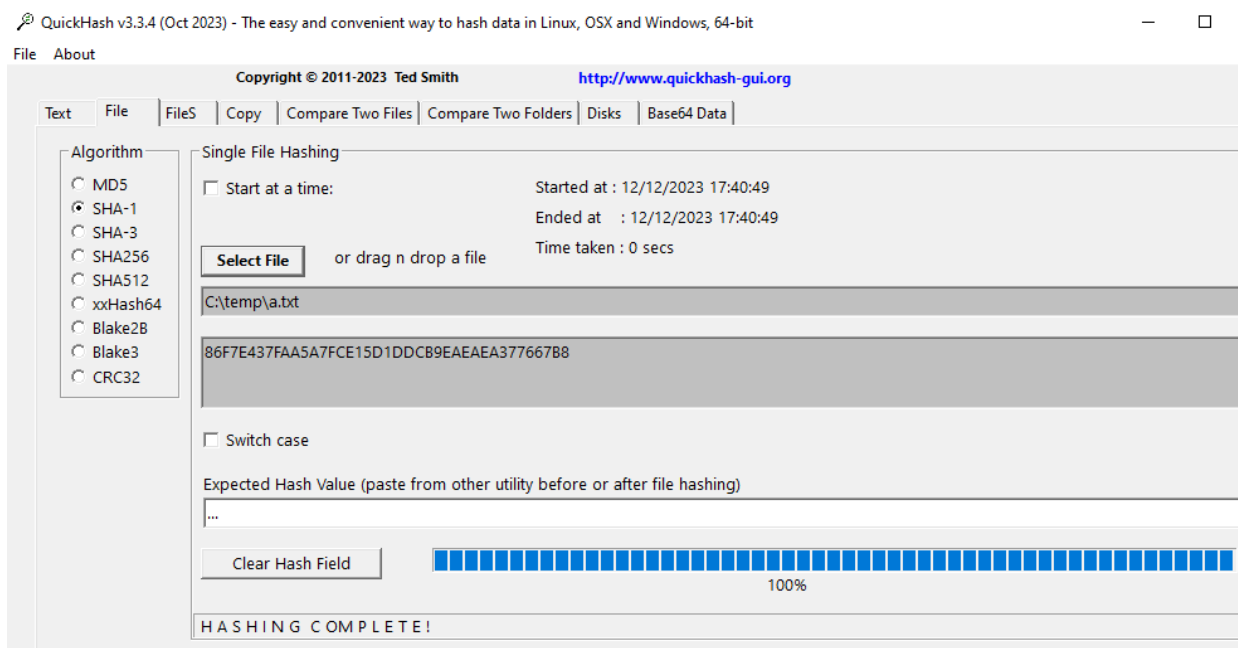


Figure 4: The SHA-1 hash value of the file 'a.txt', storing the character 'a', computed by QuickHash-GUI

Forensic Images

A forensic image created and saved as an E01 series of files can be read and verified by QuickHash-GUI by the incorporation of the libewf open-source library. To validate this, a 10Mb virtual hard disk was created and formatted with NTFS. The file 'a.txt' containing the single character 'a' was saved to it. The virtual hard disk was then added to X-Ways Forensics and forensically imaged. This forensic image is provided as part of the data validation pack.

The hash of the forensic image 'QH-DataValidation-Image.E01' as computed by X-Ways Forensics is :

SHA-1 :

3643F99AB6C9A8C3233274D4FE716FEE381CFD11

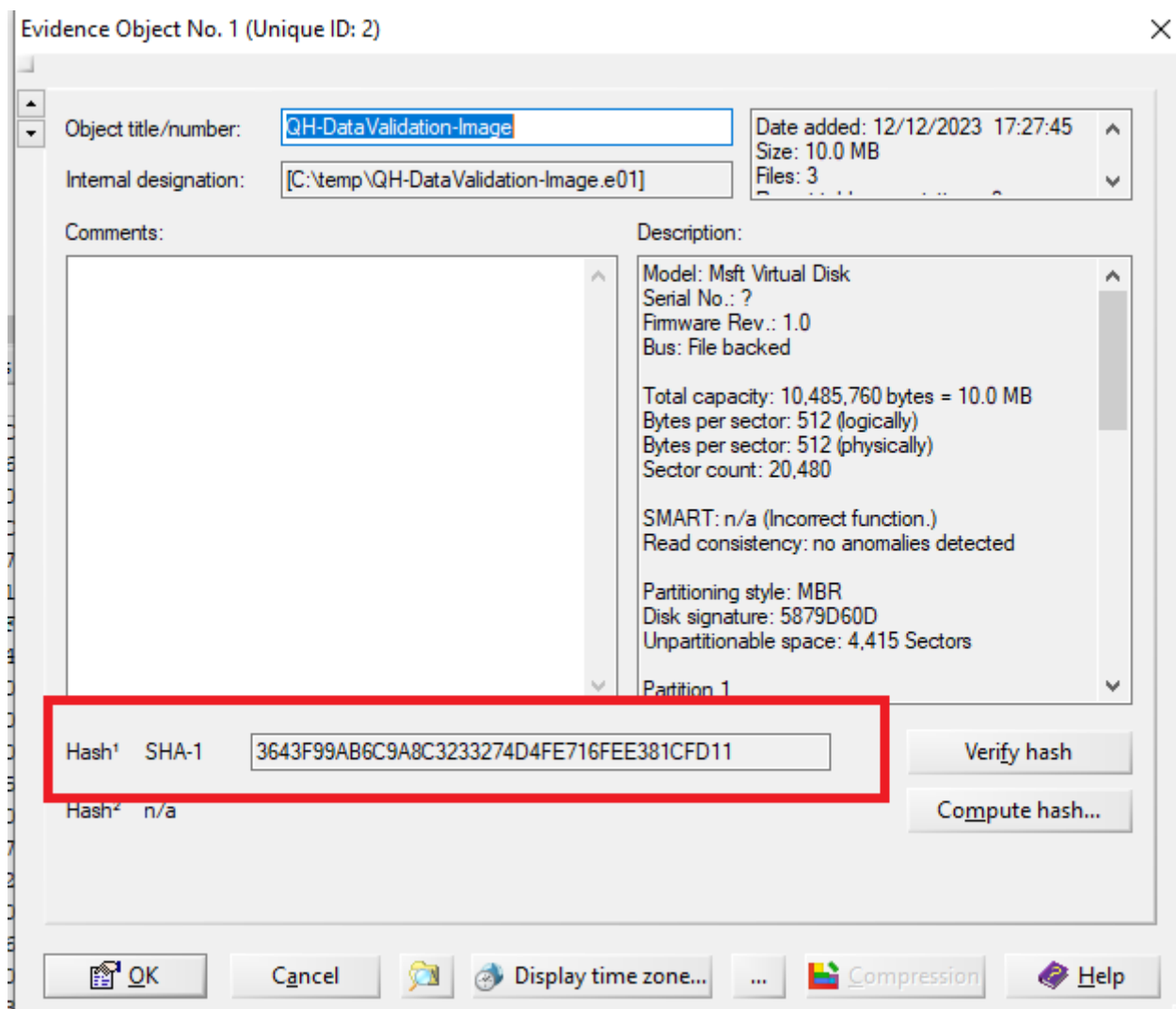


Figure 5: The SHA-1 hash of the provided validation pack forensic image shown by X-Ways Forensics

The hash computed for the same image by QuickHash-GUI should be the same, and it should successfully retrieve the expected hash from within the E01, as shown by the screenshot below :

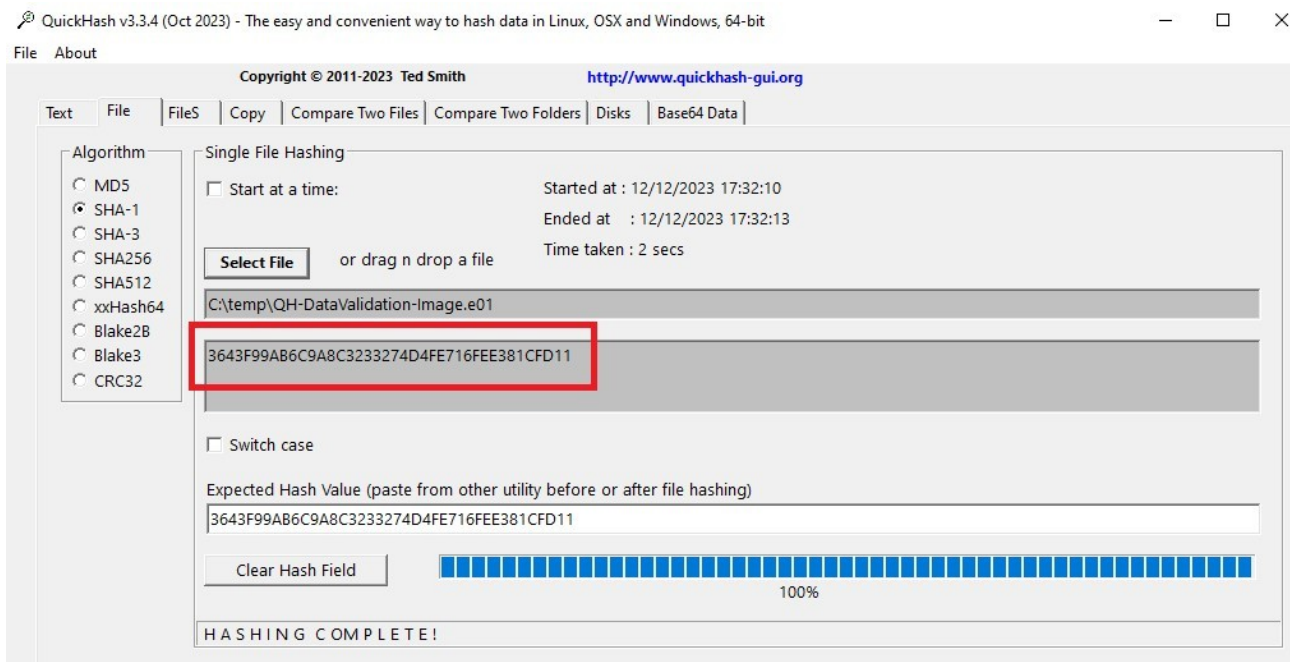


Figure 6: The SHA-1 hash of the provided validation pack forensic image shown by QuickHash-GUI

Hash Collisions and Deprecation of ‘Cryptographic’ status

The engineered hash collisions of MD5 and, more recently, [SHA-1 in 2017](#) have resulted in those algorithms no longer being considered secure enough for security measures, like password hash storage for banking purposes. As such, they are less commonly referred to as “cryptographic hash algorithms” now, despite being declared as such originally. That said, both algorithms are still used extensively across the world, especially for the purposes of routine data validation in non-cryptographic settings, due to their speed and relative reliance in the context of disk hashing, forensic image hashing etc for which no collisions have yet been demonstrated to the knowledge of the author. They continue to be used intensively across the world in the field of digital forensics.

Suggested Validation Exercise

- 1) Extract the content of the data validation zip file to a local folder.
- 2) Launch the version of QuickHash-GUI within the IT estate
- 3) Use the “Text” tab and type the letter ‘a’ in the text area.
- 4) Check the computed hash values match one, or all, of the following :

MD5 :

0cc175b9c0f1b6a831c399e269772661

SHA-1 :

86f7e437faa5a7fce15d1ddcb9eaeaea377667b8

SHA-256

:

ca978112ca1bbdcafac231b39a23dc4da786eff8147c4e72b9807785afee48bb

SHA-512

:

1f40fc92da241694750979ee6cf582f2d5d7d28e18335de05abc54d0560e0f5302

860c652bf08d560252aa5e74210546f369fbbbce8c12cfc7957b2652fe9a75

5) Then use the “File” tab, and navigate to the file ‘a.txt’ and check it computes the same values as above.

6) If your organisation uses QuickHash-GUI to verify forensic images, remain in the “File” tab, select the E01 file segment from the data validation pack (or the reader could generate their own, if they would prefer) and choose to hash the full E01 when prompted to do so. The resulting SHA-1 hash should be :

SHA-1

3643F99AB6C9A8C3233274D4FE716FEE381CFD11