Segmented Hashes

White paper

Created by

There are two primary use cases for segmented hashes.

1. Better resiliency against data corruption

If your acquired evidence image gets damaged at some point in the future, with regular hashes you will get a hash mismatch upon verification and the entire image becomes useless.

With segmented hashing only a single hash value will become invalid while the rest of the image can still be validated.

2. Hashes for damaged source device

Segmented hashes are great to support multi-pass imaging tools and handling of bad sectors.

Hashes are calculated only for the imaged regions, while all bad sectors are excluded from calculation. This allows to validate a hash even when the source drive is damaged.

How segmented hashing is it different from regular hashing?

With regular hashing, you get a single hash for the entire image.

With segmented hashing, you end up with many hashes of corresponding LBA ranges (chunks) of the image. The sum of these

LBA ranges represents the entire image, just not necessarily in sequential order. By validating all hashes in a set you can still prove that the entire image was not modified.

Example: Segmented hashes when chunk size is 4GB

Segmented hashes file format

All hashes are saved in a CSV file with the following simple format:

```
Hash,start LBA,end LBA
```

Example:

```
75c92419e86ce82734ef3bbb781e6602,0,8388608
e2c7fc5264bae820e46c50b0502236d3,8388609,16777216
42718e48b5adb59563c98727cbce0619,16777217,25165824
```

... And so on until the last LBA.

Free open-source tool

https://github.com/atola-technology/seghash

seghash is a tool that allows to:

- calculate segmented hashes of image
- verify calculated segmented hashes

Example: Segmented hashes calculation

TODO

Example: Segmented hashes verification

TODO