# FolderHash v1.0

05 July 2011

Çağdaş Çalık
ccalik@metu.edu.tr

This short document describes the hash computation algorithm used in FolderHash tool. The algorithm produces a hash value for a given root folder on a file system. The building blocks are hash function SHA-256 and HMAC algorithms. The main security assumption for the hash calculation of FolderHash tool is that it should not be possible to find two different folders producing the same hash value, provided that the underlying hash function SHA-256 is collision resistant. Here, the difference in folders is considered as either a difference in the folder/file names (except the root folder), file contents or folder structure.

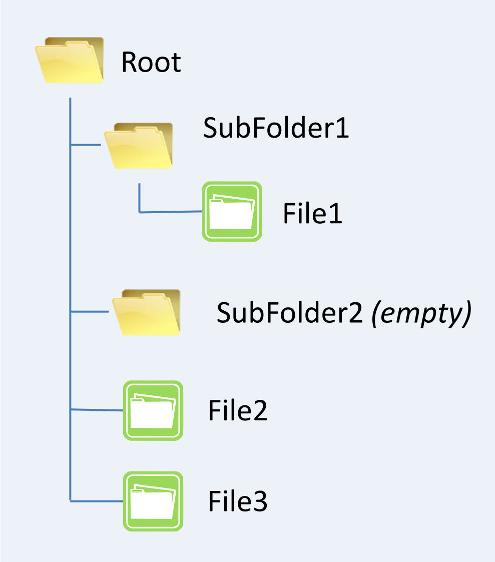
There are two basic procedures: H_File and H_Folder methods. H_File uses SHA-256 to generate the hash of a file. The actual data hashed is concatenation of an optional *Attributes* field, null-terminated name of the file and the file contents.

$$H_{File} = SHA256(Attributes \parallel FileName \parallel FileContent)$$

H_Folder function calculates the hash value of a folder by running the HMAC algorithm on the sum (XOR) of the hashes of files and folders it contains with folder path relative to the root folder as the HMAC key.

$$H_{Folder} = HMAC_{SHA256}(FolderPath, \sum_{file\ in\ Folder} H_{File}(file) + \sum_{folder\ in\ Folder} H_{Folder}(folder))$$

**Example:** For the folder structure depicted on the right, hash calculation for the 'Root' folder is carried out as follows: Hashing of files is obvious, hashing a folder requires the hashing of folders and files it countains. i.e., in order to calculate the hash of a folder, all subfolders and files in it must be processed.



$$H_{Folder}(SubFolder1) = HMAC(\text{"./}SubFolder1\text{"}, H_{File}(File1))$$

$$H_{Folder}(SubFolder2) = HMAC(\text{"./}SubFolder2\text{"}, 0...0)$$

$$H_{Folder}(Root) = HMAC(\text{"."}, H_{Folder}(SubFolder1) + H_{Folder}(SubFolder2) + H_{File}(File2) + H_{File}(File3))$$

**Remarks:**

For the following remarks, it’s assumed that finding collisions in SHA-256 and HMAC using SHA-256 is not *practical*.

- Hash value of a folder does not depend on the root folder name, i.e., the topmost folder for which the actual hash calculation is performed.
- Hash value of two folders (under the root folder) cannot be the same. Whatever the content of the two folders are, hash value of a folder is computed using HMAC, with the folder path as the key, which is unique for all folders.
- Hash value of two files will be the same if and only if they have the same name (hence located in different folders) and same contents. This does not pose a security problem because even if the file hashes are the same, they will be HMAC’ed with different keys. Indeed, it is sufficient to include the relative path of the file in the H_File computation to produce a unique hash value for each file. By not doing this way lets one to see the identical files in a folder tree (provided that their names are the same and the detailed report option is selected which dumps all the hash values in a folder).
- Hash values of files at the same folder level cannot be equal. This is guaranteed by the null-terminated file name string used in the hash computation, which forces the messages to be hashed to differ independent of their contents.
- Processing order of items in a folder is not important since these values are all xor’ed at the end.
- Hash value of an empty folder is HMAC of the all zero string (32 bytes) with the relative folder path as the key. An implication of this is, if someone can force a hash value of a file to be all zero –supposed to be as hard as finding preimages for SHA-256-, he/she has found a collision for the folder hashing algorithm. The two colliding folders are the one with an empty content and the other with the file whose hash value is zero.