An approach to determine the identity and integrity of an ESEF reporting package or a stand-alone document

Royal Netherlands Institute of Chartered Accountants

NBA

Version 1.2022.1.0

Content

Introduction

In a permission letter for the client, it must be stated clearly what the audited object is, and it must be certain that this object has not been modified. In the paper world this is done so by authenticating each separate page of the audit object and to link the auditor's opinion and audit object to each other. The reason for doing this, is that in case of doubt, it can be determined what the original control object was.

In this case the audit object is for example an ESEF-filing. This a Reporting Package, saved as a ZIP-file, which contains, based on a prescribed structure, multiple files. These files represent the issuer's XBRL Inline document and the issuer's taxonomy. Since an ESEF-filing is digital, authenticating the audit object is no longer possible in the old way. The same situation also applies to stand-alone XHTML documents which are used by issuers who only needs to prepare non-consolidated financial statements.

Authenticating digital files is done by the method of creating hashes. Creating a hash is a mathematical operation on a file which results in a unique string of characters. Every time the same mathematical operation is conducted on the file, the result remains the same. If something changes in the file, the result will be different. By adding the hash to the auditor's opinion, the audit object can be authenticated, and it can be determined whether the audit object has changed or not.

Objectives

The conditions for the use of a hash on ESEF-filings or stand-alone document[1] are:

1. Must be reliable.
2. To apply platform independently.
3. The basis for the mathematical operation must be open source.
4. Must be flexible.

Approach

The approach consists of a Proof-of-Concept which meets the objectives.

Reliable

A hash should not be created for an entire Reporting Package (ZIP-file). The date of creation (due to the recreation of a Reporting Package or file) or the order of the files in the ZIP-file have an impact on the digital signature of the ZIP-file. Different digital signatures will result in different hashes. Even if the same Reporting Package is zipped with a different program the digital signature is different, thus resulting in a different hash.

The only reliable way to create a hash is on the content of a file. If needed the Proof-of-Concept extracts all files from the ZIP-file to calculate for every (relevant) file a hash.

Platform independent

To create a hash which is on every platform (Windows, macOS, Unix etc.) the same, canonicalization of the content must take place. Canonicalization is a process to ensure that each file is digital interpreted in the same way.

[1] This Proof-of-Concept also allows the generation of hashes for non-XML based files, such as images and Microsoft Office files.

The Proof-of-Concept canonicalizes all XML-based files before the hash is calculated.

Open source

To ensure a broad and future-proof use, it is necessary that the basis for calculating a hash is available for each platform and can be used at no additional cost.

The Proof-of-Concept uses SHA256 as a standard for creating a hash. The patent for this standard is released under a royalty-free license.

Flexible

The solution for creating hashes must be easy to adjust so it can be used under different and changing circumstances.

Global operation of the Proof-of-Concept

1. Select a Reporting Package (ZIP-file) or stand-alone document
2. If needed all files in the Reporting Package are being extracted.
3. From all relevant files (in the case of this Proof-of-Concept all files excluding the auditor's opinion in PDF-format[2]) a hash is being generated (calculated).
    a. Before generating the hash of the XML-based files a canonicalization takes place.
    b. For images (JPG/JPEG, PNG, SVG and GIF) the hashes are being generated without canonicalization.
4. All obtained hashes are being sorted. By this the location of the files in the ZIP-file have no influence on the generated hash.
5. All hashes are being truncated to the last 16 characters and concatenated in alphabetical order to one string (in case of a Reporting Package).
6. From this string, an overall hash is being generated (in case of a Reporting Package).
7. The overall hash is being truncated to the last 16 characters. This string is the reference to the audit object (e.g. the Reporting Package).
8. In the result not only, the reference (overall hash) is presented, but also the individual truncated hashes of all the files in scope. With this it is easy to detect which file has been modified.

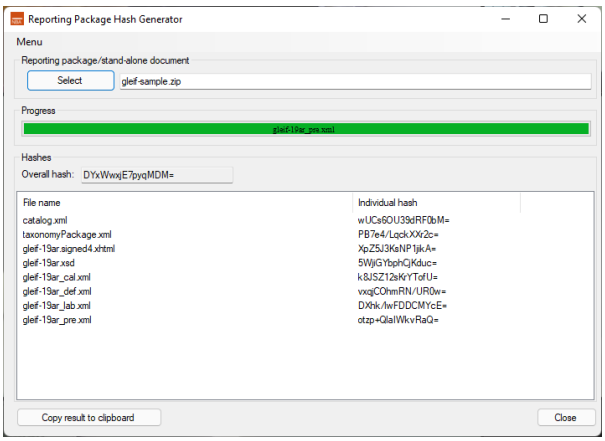

Figure 1 Sample output of the Proof-of-Concept.

[2] If the auditor's opinion would be a part of the hash that is being added to the auditor's opinion as a reference to the audit object, the content of the Reporting Package would change. Thus, resulting in a different hash.

The source code (C# and VB.Net) of the Proof-of-Concept is freely available. Also, a compiled program is available at https://www.nba.nl/reporting-package-hash-generator.

Agreements for creating the hash

To ensure that the calculated hash of an identical (audit) object is always the same on different platforms, agreements must be made. For the Proof-of-Concept, these agreements are:

1. All XML-based files MUST be canonicalized according to the C14N XML canonicalization specification of the World Wide Web Consortium (W3C). This specification can be found at https://www.w3.org/TR/2001/REC-xml-c14n-20010315.
2. The canonicalization method MUST be inclusive and with comments.
3. Whitespaces in the XML files MUST be preserved.
4. Sorting of the truncated hashes MUST be case insensitive.

Before sorting	After sorting
9oL76gt5Ad590Uo=	3KYp2zz3V5Gyznk=
HcqfOQj1NRC2SfQ=	3nyYXQ3b0IIOzvA=
3nyYXQ3b0IIOzvA=	6EH8mh9szTTTrlg=
6EH8mh9szTTTrlg=	9oL76gt5Ad590Uo=
broTPRv83pDvUxo=	9rIa6+TUN8MDLbM=
9rIa6+TUN8MDLbM=	broTPRv83pDvUxo=
dFdNpHXwRwn2Eb0=	dFdNpHXwRwn2Eb0=
3KYp2zz3V5Gyznk=	HcqfOQj1NRC2SfQ=

Table 1 An example of case insensitive sorting

5. Hashes are calculated according to SHA-256. The standard can be found at https://web.archive.org/web/20161126003357/http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf.

Warning: The .NET Framework does not fully comply with the C14N XML canonicalization specification. The standard XML namespace is not being omitted when creating the canonicalization result. Because of this, in the Proof-of-Concept all standard XML namespaces are being removed from the files before canonicalization.

The NBA's membership comprises a broad, diverse occupational group of over 23,000 professionals working in public accountancy practice, at government agencies, as internal accountants or in organisational management. Integrity, objectivity, professional competence and due care, confidentiality and professional behaviour are fundamental principles for every accountant. The NBA assists accountants to fulfil their crucial role in society, now and in the future.

Further information
If you have any questions, please send an e-mail to:
esef@nba.nl