Checksums

Unable to render {include} The included page could not be found.

Introduction

Fedora provides the capability of computing and storing checksums for Datastreams in a digital object, and later using that checksum to verify that the contents of that object has not been changed. This Datastream checksumming was added to Fedora to aid those in preservation and content security. Using this capability, Fedora repositories can compute a checksum for each Datastream of a digital object, and can later use this checksum to conclusively determine whether the contents of the Datastream has been changed, either through a minor hardware failure (such as a bad disk sector) changing the data stored in the low-level store, or through a user changing the contents of a file in the low-level store either accidentally or maliciously. This is achieved by the Fedora repository computing a checksum for the content whenever the content is added or modified via Fedora API-M functions, and allowing that stored checksum to be compared to one computed for the currently stored version of that content. Note that since valid changes made to Datastreams via the Fedora API-M functions cause the stored checksum to be recomputed, the checksumming feature is only designed to detect changes to the Datastreams of a digital object *outside* of Fedora; changes made to Datastreams within Fedora (via API-M functions) can be tracked through content versioning and through the audit trail records.

Enabling Automatic Checksumming

Since computing checksums for every Datastream when a Datastream is initially added and whenever a Datastream is modified via a Fedora API-M call will cause all such operations to run more slowly, automatic checksumming of Datastreams is *disabled* by default. To enable automatic checksumming, the Fedora Administrator must edit the `fedora.fcfg` file by finding the line in that file specifying: `<param name="autoChecksum" value="false">` and changing the value to "true". The configuration file entry immediately following the "autoChecksum" entry: `<param name="checksumAlgorithm" value="MD5">` specifies which checksumming algorithm is to be used for the automatically generated checksums. The default algorithm is "MD5" and the other valid values for this entry are: "SHA-1" "SHA-256" "SHA-384" and "SHA-512". Entering any other value for the "checksumAlgorithm" parameter will effectively disable automatic checksumming and will generate annoying warning messages.

How Automatic Checksumming Works

When automatic checksumming is enabled, whenever a object is ingested into Fedora, as each Datastream is processed, all of the bytes comprising the content of the Datastream are passed to the appropriate checksumming algorithm. This algorithm will compute and return a digital signature for the content of the Datastream. These checksumming algorithms are designed such that any minor change to the content will produce a wildly different result for the computed checksum. These computed Datastream checksums will then be stored in the XML representation of the digital object. Additionally, whenever a new Datastream is added to an existing object (via addDatastream), and whenever a existing Datastream is modified (via modifyDatastreamByValue or modifyDatastreambyReference) a new checksum will be computed and stored in the object.

Subsequently, someone interested in verifying that the content of a Datastream has been neither damaged nor tampered with will be able to invoke the new API-M function compareDatastreamChecksum. This new function will take the checksum string stored for the specified Datastream and compare it with a newly-computed checksum using the same checksum algorithm as was originally used. If the checksums match, the new API-M function will return a string containing the checksum value. However if the checksums do not match, the function will return a string indicating the error.

Overriding Automatic Checksumming

In some circumstances a user of Fedora will encounter a situation where the checksumming configuration specified for the repository as a whole is not suitable for one or more Datastreams of a given object. Perhaps automatic checksumming is disabled, but there is some Datastream for which you want to have a checksum computed, or, conversly, perhaps automatic checksumming is enabled and some object has a Datastream for which checksumming doesn't make sense (i.e. either a Datastream with dynamic content that changes over time or a truly enormous Datastream for which the checksumming operation would be too time consuming.

It is possible to override the checksumming operation that will be performed and stored for a given Datastream via new parameters that have been added to the API-M functions addDatastream, modifyDatastreamByValue and modifyDatastreamByReference. These functions each have two new parameters, checksumType and checksum. If a value is specified for the parameter checksumType for any of these three functions that is the algorithm that will be used for computing the checksum for that particular Datastream, rather than the global default checksum algorithm specified in the fedora.fcfg file.

The valid values for the checksumType parameter for these three functions are: "MD5" "SHA-1" "SHA-256" "SHA-384" "SHA-512" as above, but also "DISABLED" which will turn off checksumming for that particular Datastream. Additionally for the two modifyDatastream functions the value null specifies that the checksum algorithm in force before the modify operation should continue to be used, whereas the value "DEFAULT" specifies that the checksumming algorithm for that particular Datastream should be changed back to whatever default checksum algorithm has been specified in the fedora. fcfg file.

Another way to override the default checksumming mechanism is via new attributes that have been added to the FOXML and METS specs for ingesting digital objects. The FOXML and METS specifications now allow the checksum algorithm to be used for each Datastream to be specified as attributes on one of the elements defining that Datastream. When a checksum algorithm is specified in the XML for a Datastream of a digital object, this value will be used to compute the checksum for that Datastream rather than the default algorithm specified in the fedora.fcfg file. Note that the syntax for thusly specifying a checksum algorithm for a Datastream in a digital object is different for FOXML and for METS, the specific syntax to use can be found in the schemas for those XML formats.

Verifying Datastream Content

Once checksums are computed and stored for a given Datastream it is possible to verify that the contents of that Datastream has not been changed in any way since the checksum was initially computed. To perform this verification a user simply invokes the new Fedora API function compareDatastreamChecksum, passing in the object id and Datastream id of the Datastream to be verified (plus an optional date string if a version of the Datastream other than the most recent one is to be verified). The API function will read in the digital content of the Datastream, and compute a checksum using the same checksum algorithm stored with the Datastream, and compare the resulting value with the one it previously computed and stored in the digital object. If the two checksums are identical, the function will return the value of the checksum (which could then be stored externally, if desired, as an added measure of security). If the newly computed checksum doesn't match the stored one, the API function will return a message indicating this checksumming failure. The action to take when this situation occurs is left to the user.

Additional Datastream Verification

In some circumstances a user may want further assurances that the content of a Datastream has not been unintentionally changed, perhaps through a faulty network connection or through a "man-in-the-middle" data modification. To provide this capability, there is another new parameter that can be passed in for the API-M functions addDatastream, modifyDatastreamByValue and modifyDatastreamByReference, named checksum. If a value is specified for this parameter, rather than leaving it null it is interpreted as a request to compute the checksum using the provided checksumType, and then compare it with the checksum that was passed in. If the checksums do not match, Fedora will assume that the data that it read for the content of the new of modified Datastream somehow was changed or damaged in transmission and the API-M function will fail and the repository will be rolled back to the state it was in prior to the call. N.B. for inline XML Datastreams, the content is normalized internally during the checksum computation process, which will make devising and passing-in the correct checksum to ensure the integrity of the passed-in content will be somewhat difficult.

Fedora Administrator

The Datastream display panels in the Fedora Administrator display the checksum algorithm and computed checksum. For the current version of a Datastream, the algorithm to be used can be changed via a drop-down list. For previous versions of a Datastream these values are displayed, but are not editable.

Unable to render {include} The included page could not be found.