Standards Change Request

File Checksums **SCR3-1034.v4**
Elizabeth D. Rye **March 20, 2006**

Provenance:

Date: 2006-03-06, revision 3.0
Working Group: E. Rye (lead), T. King, M. McAuley
Title: File Checksums (SCR3-1034.v3)

Date: 2005-11-14, revision 2.0
Working Group: T. King (lead), M. McAuley
Title: MD5 Checksums (SCR3-1034.v2)

Date: 2004-11-22, revision 1.0
Working Group: J. Wilf (lead), T. King, M. McAuley
Title: MD5 Checksums for Files (SCR3-1034.v1)

Problem:

Provide a means for testing the integrity of PDS data.

As an entity responsible for maintaining data, it is critical that the PDS be able to ascertain the integrity of its archive. This includes verifying the integrity of data stored on various types of external physical media (all of which have finite life spans), detecting errors introduced during transfer of data to newer media, and detecting errors that occur during the transmission of data from data providers to the PDS, between PDS nodes, from the PDS to the NSSDC, and from the PDS to end users.

Proposed Solution:

A simple method for detecting these types of errors is to create and maintain a list of checksum values for every file contained on a PDS archive volume (logical or physical).

(Alternatively, checksums may be contained within data product labels, although this is only practical for storing the checksums of products with detached labels and provides no test for the integrity of the label files themselves. When used this way, the checksum keyword should be positioned within the object description of the object it is calculated for. Thus, in the case of a data product with a single detached label describing two data

objects in two separate files, the checksum for each file would be stored within each of the explicit FILE object definitions.)

There are various types of checksums available that can be used for this purpose. The PDS has traditionally supported a simple checksum consisting of a 32-bit integer sum of all bytes in a data file. The one currently best suited to this purpose utilizes the 5th generation Manifest Digest (MD5) algorithm. However, the proposals outlined in this SCR are not limited to any particular type of checksum and should allow for the easy implementation of new and better checksum algorithms as they become available.

The proposed changes outlined in this SCR are:

1. Establish a reserved file, "CHECKSUM.TAB" (or "axxCHKSM.TAB" for multiple files), which contains checksum values for all files on an archive volume, to be included in the INDEX directory of the archive.
2. Create a new keyword, CHECKSUM_TYPE, for use in the CHECKSUM.LBL (or axxCHKSM.LBL) file, to provide flexibility in permitting various types of checksums to be used.
3. Update the element definition for the MD5_CHECKSUM keyword, updating the STATUS_TYPE to APPROVED.

Requested Changes:

Changes to the Standards Reference

The following changes to the PDS Standards Reference are required to support this SCR:

Add to section 10.2.2 Reserved File Names, "CHECKSUM.TAB".

Add to Chapter 19.3.2.3 INDEX Subdirectory, after INDEX.TAB:

CHECKSUM.LBL **Required**

This is the PDS label for the CHECKSUM.TAB file. If the type of checksum is not specified as a column in the CHECKSUM.TAB file, the column object description for the CHECKSUM column should contain the CHECKSUM_TYPE keyword.

Although CHECKSUM.LBL is the preferred name for this file, the name axxCHKSM.LBL may also be used, with axx replaced by an appropriate mnemonic.

CHECKSUM.TAB **Required**

This file contains a checksum for every file on the volume except itself and its label. The format is to be a PDS ASCII tabular format, with one column (named CHECKSUM) providing the checksum values and another column (named FILE_SPECIFICATION_NAME) containing the path and name of each file in the archive relative to the root directory of the volume. An optional third column (named CHECKSUM_TYPE) may be used to indicate varying types of checksums calculated for files on the volume.

Although CHECKSUM.TAB is the preferred name for this file, the name axxCHKSM.TAB may also be used, with axx replaced by an appropriate mnemonic. For an example of the CHECKSUM.TAB and CHECKSUM.LBL files, see Appendix D, section D.2.

Each figure in Chapter 19. Volume Organization and Naming, will need to be updated to include a “CHECKSUM.TAB” and a "CHECKSUM.LBL" file in the INDEX directory.

Appendix D, section D.2 add the sample CHECKSUM.TAB and CHECKSUM.LBL files as shown in the attachment. (The following sections of Appendix D will need to be re-numbered.)

Changes to the Data Dictionary

Modify the description of the MD5_CHECKSUM keyword as shown in the attached element definition template.

Add the new keyword, CHECKSUM_TYPE, as shown in the attached element definition template.

Changes to the PDS Tool Suite

There are no changes needed to any PDS tool, since a number of utilities are already widely available which can be used to produce and read the CHECKSUM.TAB file.

Impact Assessment:

In addition to the above described changes, at some point checksum files will have to be generated for all archive volumes generated prior to the approval of this Standards Change Request. Furthermore, each node will be responsible for developing tools to periodically validate the integrity of their archive holdings using the checksum files generated in response to this SCR. Mechanisms may also be eventually generated (presumably associated with the product servers) to provide checksum values to users who download individual PDS data product files.

```
PDS_VERSION_ID                    = PDS3
LABEL_REVISION_NOTE               = "2004-04-06, CN: BAM;
2004-10-14, PPI: S. Joy; 2006-03-06, EN: EDR"

OBJECT                            = ELEMENT_DEFINITION
  ELEMENT_NAME                    = "md5_checksum"
  BL_NAME                         = "md5checksum"
  DESCRIPTION                     = "

   The MD5 algorithm takes as input a file (message) of arbitrary length
   and produces as output a 128-bit 'fingerprint' or 'message digest' of
   the input. It is conjectured that it is computationally infeasible to
   produce two messages having the same message digest, or to produce
   any message having a given prespecified target message digest. The
   MD5 algorithm is intended for digital signature applications.

   Most standard MD5 checksum calculators return a 32 character
   hexadecimal value containing lower case letters.  In order to accommodate
   this existing standard, the PDS requires that the value assigned to the
   MD5_CHECKSUM keyword be a value composed of lowercase letters (a-f) and
   numbers (0-9).  In order to comply with other standards relating to the
   use of lowercase letters in strings, the value must be quoted using double
   quotes.


   Example:  MD5_CHECKSUM = '0ff0a5dd0f3ea4e104b0eae98c87f36c'

   The MD5 algorithm was described by its inventor, Ron Rivest of
   RSA Data Security, Inc., in an Internet Request For Comments
   document, RFC1321 (document available from the PDS).

   References
   ==========
   [1] Rivest, R., The MD4 Message Digest Algorithm, RFC 1320, MIT and
       RSA Data Security, Inc., April 1992.

   [2] Rivest, R., The MD4 message digest algorithm, in A.J.  Menezes
       and S.A. Vanstone, editors, Advances in Cryptology - CRYPTO '90
       Proceedings, pages 303-311, Springer-Verlag, 1991.

   [3] CCITT Recommendation X.509 (1988), The Directory -
       Authentication Framework.  (Note: In the X.509 type
       AlgorithmIdentifier, the parameters for MD5 should have type NULL.)"

  GENERAL_DATA_TYPE               = "CHARACTER"
  MAXIMUM                         = ""
  MINIMUM                         = ""
  MAXIMUM_LENGTH                  = "32"
  MINIMUM_LENGTH                  = "32"
  STANDARD_VALUE_TYPE             = "DEFINITION"
  STANDARD_VALUE_SET_DESC         = "N/A"
  KEYWORD_DEFAULT_VALUE           = "N/A"
  UNIT_ID                         = "NONE"
  SOURCE_NAME                     = "PDS CN/B. SWORD"
  FORMATION_RULE_DESC             = "N/A"
  SYSTEM_CLASSIFICATION_ID        = "COMMON"
  GENERAL_CLASSIFICATION_TYPE     = "N/A"
  CHANGE_DATE                     = "2006-03-20"
```

```
  STATUS_TYPE                     = "APPROVED"
  STANDARD_VALUE_OUTPUT_FLAG      = "N"
  TEXT_FLAG                       = "N"
  TERSE_NAME                      = "md5checksum"
  SQL_FORMAT                      = "CHAR(32)"
  BL_SQL_FORMAT                   = "char(32)"
  DISPLAY_FORMAT                  = "JUSTLEFT"
  AVAILABLE_VALUE_TYPE            = "N/A"
END_OBJECT                        = ELEMENT_DEFINITION
END
```

```
PDS_VERSION_ID                    = PDS3
LABEL_REVISION_NOTE               = "2006-03-06, EN: EDR"

OBJECT                            = ELEMENT_DEFINITION
  ELEMENT_NAME                    = "checksum_type"
  BL_NAME                         = "checksumtype"
  DESCRIPTION                     = "
   The CHECKSUM_TYPE keyword is used to specify the type of checksum
   algorithm used to calculate a checksum for a file or data object."


  GENERAL_DATA_TYPE               = "IDENTIFIER"
  MAXIMUM                         = "N/A"
  MINIMUM                         = "N/A"
  MAXIMUM_LENGTH                  = "12"
  MINIMUM_LENGTH                  = "1"
  STANDARD_VALUE_TYPE             = "DYNAMIC"
  STANDARD_VALUE_SET              = {"MD5", "SHA-1"}
  STANDARD_VALUE_SET_DESC         = "N/A"
  KEYWORD_DEFAULT_VALUE           = "N/A"
  UNIT_ID                         = "N/A"
  SOURCE_NAME                     = "PDS EN/E. RYE"
  FORMATION_RULE_DESC             = "N/A"
  SYSTEM_CLASSIFICATION_ID        = "COMMON"
  GENERAL_CLASSIFICATION_TYPE     = "N/A"
  CHANGE_DATE                     = "2006-03-20"
  STATUS_TYPE                     = "APPROVED"
  STANDARD_VALUE_OUTPUT_FLAG      = "Y"
  TEXT_FLAG                       = "N"
  TERSE_NAME                      = "checksumtype"
  SQL_FORMAT                      = "CHAR(12)"
  BL_SQL_FORMAT                   = "char(12)"
  DISPLAY_FORMAT                  = "JUSTLEFT"
  AVAILABLE_VALUE_TYPE            = "N/A"
END_OBJECT                        = ELEMENT_DEFINITION
END
```

D.2 CHECKSUM.TAB and CHECKSUM.LBL

Each PDS archive volume must include a "CHECKSUM.TAB" file in the INDEX subdirectory. This file must be accompanied by a detached PDS label. (Note that in the case of multiple checksum files in the same directory of an archive volume, the files may be named axxCHKSM.TAB and axxCHKSM.LBL respectively, where axx is replaced by an appropriate mnemonic.) The CHECKSUM.TAB file contains a checksum for every file contained on the archive volume (or in the entire archive, if stored as a virtual volume online), with the exception of the checksum file itself and its label.

D.2.1 Example of CHECKSUM.TAB

```
1e8d45f622e09b9e2998af1a6d67a296 aareadme.txt
7dcfa51691ddd149a5a091ebe87b9bb1 errata.txt
f8dd7758cb5231c9e7817c4710d00b6e browse/mars/c1246xxx/i862934l.img
d8b83365f5e117b9665181944889da3d browse/mars/c1246xxx/i862934r.img
.
.
.
```

D.2.2 Example of CHECKSUM.LBL

```
PDS_VERSION_ID               = PDS3

RECORD_TYPE                  = FIXED_LENGTH
RECORD_BYTES                 = 71
FILE_RECORDS                 = 3623

DESCRIPTION                  = "CHECKSUM.TAB provides a checksum for all
                                files included on this archive volume, with
                                the exception of the checksum file itself
                                and its label."

^CHECKSUM_TABLE              = "CHECKSUM.TAB"

OBJECT                       = CHECKSUM_TABLE
  INTERCHANGE_FORMAT         = ASCII
  ROW_BYTES                  = 71
  ROWS                       = 3623
  COLUMNS                    = 2

  OBJECT                     = COLUMN
    NAME                     = CHECKSUM
    DESCRIPTION              = "The checksum of the indicated file."
    CHECKSUM_TYPE            = MD5
    DATA_TYPE                = CHARACTER
    START_BYTE               = 1
    BYTES                    = 32
```

```
  END_OBJECT                  = COLUMN

  OBJECT                      = COLUMN
    NAME                      = FILE_SPECIFICATION_NAME
    DESCRIPTION               = "Identifies the file for which the checksum
                                was calculated."
    DATA_TYPE                 = CHARACTER
    START_BYTE                = 34
    BYTES                     = 36
  END_OBJECT                  = COLUMN

END_OBJECT                    = CHECKSUM_TABLE
END
```